

Paul Clough Colum Foley Cathal Gurrin  
Gareth J.F. Jones Wessel Kraaij  
Hyowon Lee Vanessa Murdoch (Eds.)

LNCS 6611

# Advances in Information Retrieval

33rd European Conference on IR Research, ECIR 2011  
Dublin, Ireland, April 2011  
Proceedings



*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

*Lancaster University, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Alfred Kobsa

*University of California, Irvine, CA, USA*

Friedemann Mattern

*ETH Zurich, Switzerland*

John C. Mitchell

*Stanford University, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz

*University of Bern, Switzerland*

C. Pandu Rangan

*Indian Institute of Technology, Madras, India*

Bernhard Steffen

*TU Dortmund University, Germany*

Madhu Sudan

*Microsoft Research, Cambridge, MA, USA*

Demetri Terzopoulos

*University of California, Los Angeles, CA, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Gerhard Weikum

*Max Planck Institute for Informatics, Saarbruecken, Germany*

Paul Clough Colum Foley Cathal Gurrin  
Gareth J.F. Jones Wessel Kraaij  
Hyowon Lee Vanessa Murdoch (Eds.)

# Advances in Information Retrieval

33rd European Conference on IR Research, ECIR 2011  
Dublin, Ireland, April 18-21, 2011  
Proceedings

Volume Editors

Paul Clough  
University of Sheffield, Information School  
Regent Court, 211 Portobello Street, Sheffield, S1 4DP, UK  
E-mail: p.d.clough@sheffield.ac.uk

Colum Foley  
Cathal Gurrin  
Hyowon Lee  
Dublin City University, CLARITY: Centre for Sensor Web Technologies  
School of Computing, Glasnevin, Dublin 9, Ireland  
E-mail: {cfoley, cgurrin, hlee}@computing.dcu.ie

Gareth J.F. Jones  
Dublin City University, Centre for Next Generation Localisation  
School of Computing, Glasnevin, Dublin 9, Ireland  
E-mail: gjones@computing.dcu.ie

Wessel Kraaij  
TNO Human Factors, Brassersplein 2, 2612 CT Delft, The Netherlands  
E-mail: wessel.kraaij@tno.nl

Vanessa Murdoch  
Yahoo! Research, 177 Diagonal, 08018 Barcelona, Spain  
E-mail: vmurdoch@yahoo-inc.com

ISSN 0302-9743 e-ISSN 1611-3349  
ISBN 978-3-642-20160-8 e-ISBN 978-3-642-20161-5  
DOI 10.1007/978-3-642-20161-5  
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011924468

CR Subject Classification (1998): H.3, H.2, I.2, H.4, H.2.8, I.7, H.5

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

© Springer-Verlag Berlin Heidelberg 2011

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

*Typesetting:* Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

# Preface

These proceedings contain the papers presented at ECIR 2011, the 33rd European Conference on Information Retrieval. The conference was organized by Dublin City University in cooperation with the University of Sheffield, and was supported by the Information Retrieval Specialist Group at the British Computer Society (BCS-IRSG) and the Special Interest Group on Information Retrieval (ACM SIGIR). It was held during April 18–21, 2011 in Dublin, Ireland.

We greeted the attendees at ECIR 2011 with the following address: “*Táimid an-bhrodúil fáilte a chur romhaibh chuig Baile Átha Cliath agus chuig an trocha triú Comhdháil Europeach ar Aisghabháil Faisnéise. Tá súil againn go mbeidh am iontach agaibh anseo in Éirinn agus go mbeidh bhur gcuairt taitneamhnach agus sásúil. Táimid an-bhrodúil go háirithe fáilte a chur roimh na daoine ón oiread sin tíortha difriúla agus na daoine a tháinig as i bhfad i gcéin. Tá an oiread sin páipéar curtha isteach chuigh an chomhdháil seo go bhfuil caighdeán na bpáipéar, na bpóstaer agus na léiríú an-ard ar fad agus táimid ag súil go mór le hócaid iontach.* We are delighted to welcome you to Dublin for the 33rd annual European Conference on Information Retrieval. We hope that the attendees have a wonderful stay in Ireland and that their visits are both enjoyable and rewarding. We are very proud to welcome visitors from both Ireland and abroad and we are delighted to be able to include in the proceedings such high-quality papers, posters and demonstrations.”

ECIR 2011 received a total 355 submissions across four categories; 223 full-paper submissions, 48 short paper submissions, 52 poster submissions and 12 demonstration submissions. Of these submissions, 217 were from Europe (65%), 65 from Asia (19%), 44 from North and South America (13%) and 9 were from the Middle East and Africa (3%). All submissions were reviewed by at least three members of the 168-person International Program Committee, to whom we owe a debt of gratitude for providing their valuable time to ECIR 2011. Of the 223 full papers submitted, 45 were selected for oral presentation, which equates to a 20% acceptance rate. For short papers, a total of 17 were accepted for oral presentation. In addition, 24 posters and 6 demonstrations were accepted for ECIR 2011. We are pleased to note that ECIR maintained its student focus with 62% of the accepted publications having a student as the primary author. The accepted contributions represent the state of the art in information retrieval and cover a diverse range of topics including aggregated, distributed, enterprise, cross-lingual, domain-specific, interactive, multimedia and Web search, along with evaluation issues, search for social networks, the theory of IR, Q/A & NLP, recommendation and text categorization. One notable feature of 2011 was the growth of research into search technologies for social networks, which is reflected in its prominence in the ECIR 2011 program.

As in recent years, the conference was preceded by a day of workshops and tutorials. This format has proven successful at ECIR in recent years. At ECIR 2011, we hosted five workshops and four tutorials, based on the recommendation of the Workshop and Tutorial Committees.

The workshops covered a range of topical issues in information retrieval:

- Evaluating Personal Search Workshop (EPS 2011)
- Diversity in Document Retrieval (DDR 2011)
- Workshop Information Access for Personal Media Archives (IAPMA 2011)
- Information Management and Retrieval in Mobile Ad Hoc Societies Workshop (MASoc 2011)
- Information Retrieval over Query Sessions (IRQS 2011)

The four tutorials covered topics of advertising, users and risk management:

- Online Advertising: An Information Scientist's Perspective
- Web Search: The Role of Users
- Designing Effective Search and Discovery Experiences
- Risk Management in Information Retrieval

Following the success of previous BCS events, an IRSG Industry Day was organized that provided a topical selection of talks, panels and case studies from the world of search and information access with a focus on issues pertaining to IR practitioners. The Industry Day was collocated with ECIR 2011 and took place on the final day of the conference. Our thanks go to Udo Kruschwitz and Tony Russell-Rose for coordinating the ECIR 2011 Industry Day.

We would like to thank our invited keynote speakers, Kalervo Järvelin and Thorsten Joachims, for their stimulating contributions to the conference. We also thank the 2010 BCS/BCS-IRSG Karen Spärck Jones Award winner for his inspiring keynote. The recipient of the 2010 Karen Spärck Jones Award was announced at the conference; the award was presented to Evgeniy Gabrilovich in recognition of his contributions to advancing our understanding of IR and NLP through experimentation. The British Computer Society Information Retrieval Specialist Group (BCS IRSG) in conjunction with the BCS created the award to commemorate the achievements of Karen Spärck Jones. It is given annually to encourage and promote talented researchers who have endeavored to advance our understanding of information retrieval and/or natural language processing.

Special thanks go to the Workshops Chair, Leif Azzopardi, and the Tutorials Chair, Evangelos Kanoulas, and the members of their committees for their effort in selecting the workshops and tutorials for ECIR 2011. We thank the Student Mentor Chair, Nicola Stokes, for her superb efforts. We also acknowledge the commitment of our local organization team including Hyowon Lee (Proceedings and Advertising Chair), our website team of David Scott and Daragh Byrne and our local arrangements team of Colum Foley and Peter Wilkins. Special mention goes to Alan Smeaton for his constant availability to provide support and advice and to two previous General Chairs, Stefan Rieger and Mohand Boughanem, who provided valuable input at various points in the process.

In addition, we wish to thank all authors who spent their time and effort to submit their work to ECIR 2011, and all of the participants and student volunteers for their contributions and valuable support. Our gratitude also goes to the ECIR 2011 Program Committee members, the Award Committee members and other invited reviewers for more than 1,000 reviews which were required for ECIR 2011.

We are grateful to the sponsors for generously providing financial support for the conference, including Dublin City University, Science Foundation Ireland, Fáilte Ireland, Google, Microsoft Research, Bing, Yahoo! Research, CLARITY and CNGL. We would also like to thank the School of Computing at Dublin City University, in particular Stephen Blott and Mike Scott for their support as the heads of the School of Computing. Our special thanks go to Ann Marie Sweeney, Deirdre Sheridan, Krisztina Ligeti and all the CLARITY admin/support team, Anne Troy from the DCU Finance Department, Ana Terres from the DCU Office of the Vice President for Research, Harald Weinreich from ConfTool and our contacts in the Radisson Blu and the Guinness Storehouse.

Finally, we thank all of the attendees at ECIR 2011 who made the trip to Dublin.

April 2011

Cathal Gurrin  
Paul Clough  
Gareth J.F. Jones  
Wessel Kraaij  
Vanessa Murdoch

# Organization

ECIR 2011 was organized by Dublin City University, Ireland.

## Organizing Committee

General Chair	Cathal Gurrin, Dublin City University, Ireland
Program Co-chairs	Paul Clough, University of Sheffield, UK Gareth J.F. Jones, Dublin City University, Ireland
Posters Chair	Wessel Kraaij, TNO, The Netherlands
Student Mentor Chair	Nicola Stokes, University College Dublin, Ireland
Workshops Chair	Leif Azzopardi, University of Glasgow, UK
Tutorials Chair	Evangelos Kanoulas, University of Sheffield, UK
Demonstrations Chair	Vanessa Murdock, Yahoo! Research, Spain
Industry Co-chairs	Tony Russell-Rose, Endeca Technologies, UK Udo Kruschwitz, University of Essex, UK
Local Organizing Co-chairs	Peter Wilkins, Google, Ireland Colum Foley, Dublin City University, Ireland
Advertising/Proceedings Chair	Hyowon Lee, Dublin City University, Ireland
Website	Daragh Byrne, Dublin City University, Ireland David Scott, Dublin City University, Ireland

## Program Committee

Maristella Agosti	University of Padua, Italy
Ahmet Aker	University of Sheffield, UK
Giambattista Amati	Fondazione Ugo Bordoni, Italy
Massih-Reza Amini	CNRC, Canada
Leif Azzopardi	University of Glasgow, UK
Ricardo Baeza-Yates	Yahoo! Research, Spain
Alex Bailey	Google Switzerland GmbH, Switzerland
Alvaro Barreiro	University of A Coruña, Spain
Roberto Basili	University of Rome, Tor Vergata, Italy
Michel Beigbeder	Ecole des Mines de Saint- Etienne, France
Nicholas Belkin	Rutgers University, USA
Bettina Berendt	Katholieke Universiteit Leuven, Belgium
Gloria Bordogna	CNR, DALMINE, Italy
Mohand Boughanem	Université Paul Sabatier, France



Peter Bruza	Queensland University of Technology, Australia
Wray Buntine	NICTA, Canberra, Australia
Daragh Byrne	Dublin City University, Ireland
Fidel Cacheda	University of A Coruña, Spain
Jamie Callan	Carnegie Mellon University, USA
Claudio Carpineto	Fondazione Ugo Bordoni, Italy
Ben Carterette	University of Delaware, USA
Carlos Castillo Ocaranza	Yahoo! Spain
Max Chevalier	IRIT, France
Paul-Alexandru Chirita	Adobe Systems Inc., Romania
Charles Clarke	University of Waterloo, Canada
Fabio Crestani	University of Lugano, Switzerland
Bruce Croft	University of Massachusetts, USA
Alfredo Cuzzocrea	ICAR-CNR and University of Calabria, Italy
Franciska de Jong	University of Twente, The Netherlands
Pablo de la Fuente	Universidad de Valladolid, Spain
Maarten de Rijke	University of Amsterdam, The Netherlands
Arjen P. de Vries	Centrum Wiskunde & Informatica, The Netherlands
Aiden Doherty	Dublin City University, Ireland
Efthimis N. Efthimiadis	University of Washington, USA
David Craig Elswiler	University of Erlangen, Germany
Paul Ferguson	Dublin City University, Ireland
Juan M. Fernandez-Luna	University of Granada, Spain
Nicola Ferro	University of Padua, Italy
Colum Foley	Dublin City University, Ireland
Edward Fox	Virginia Tech, USA
Norbert Fuhr	University of Duisburg-Essen, Germany
Patrick Gallinari	University Pierre et Marie Curie, France
Eric Gaussier	Laboratoire d'Informatique de Grenoble/Université, France
Ayse Goker	City University London, UK
Gregory Grefenstette	EXALEAD, France
David Adam Grossman	IIT, USA
Antonio Gulli	Microsoft Bing, UK
Cathal Gurrin	Dublin City University, Ireland
Allan Hanbury	Information Retrieval Facility, Austria
Preben Hansen	Swedish Institute of Computer Science, Sweden
Donna Harman	NIST, USA
Claudia Hauff	University of Twente, The Netherlands
Jer Hayes	IBM Smarter Cities Technology Centre, Ireland
Ben He	Graduate University of Chinese Academy of Sciences, China
Yulan He	Open University, UK
Djoerd Hiemstra	University of Twente, The Netherlands

Andreas Hotho	University of Würzburg, Germany
Gilles Hubert	IRIT - University of Toulouse, France
Theo Huibers	University of Twente, The Netherlands
David A. Hull	Google, USA
Bernard Jansen	Penn State University, USA
Frances Johnson	Manchester Metropolitan University, UK
Hideo Joho	University of Tsukuba, Japan
Kristiina Jokinen	University of Helsinki, Finland
Joemon M. Jose	University of Glasgow, UK
Jaap Kamps	University of Amsterdam, The Netherlands
Evangelos Kanoulas	University of Sheffield, UK
Jussi Karlgren	SICS, Kista, Sweden
Gabriella Kazai	Microsoft Research, UK
Jaana Kekalainen	University of Tampere, Finland
Manolis Koubarakis	National and Kapodistrian University of Athens, Greece
Wessel Kraaij	TNO, The Netherlands
Udo Kruschwitz	University of Essex, UK
Mounia Lalmas	University of Glasgow, UK
James Lanagan	Dublin City University, Ireland
Monica Angela Landoni	USI University of Lugano, Switzerland
Birger Larsen	Royal School of Library and Information Science, Denmark
Martha Larson	Delft University of Technology, The Netherlands
Matt Lease	University of Texas at Austin, USA
Hyowon Lee	Dublin City University, Ireland
Johannes Leveling	Dublin City University, Ireland
Xuelong Li	Birkbeck College, University of London, UK
David E. Losada	University of Santiago de Compostela, Spain
Mihai Lupu	Information Retrieval Facility, Austria
Craig Macdonald	University of Glasgow, UK
Andrew MacFarlane	City University London, UK
Marco Maggini	University of Siena, Italy
Thomas Mandl	University of Hildesheim, Germany
Massimo Melucci	University of Padua, Italy
Alessandro Micarelli	Roma Tre University, Italy
Dunja Mladenic	J. Stefan Institute, Ljubljana, Slovenia
Marie-Francine Moens	Katholieke Universiteit Leuven, Belgium
Alistair Moffat	University of Melbourne, Australia
Henning Müller	University of Applied Sciences Western Switzerland, Switzerland
Vanessa Murdock	Yahoo! Research, Spain
G. Craig Murray	University of Maryland, College Park, USA
Eamonn Newman	Dublin Institute of Technology, Ireland

Jian-Yun Nie	DIRO, Université Montréal, Canada
Neil O'Hare	Dublin City University, Ireland
Michael O'Mahony	University College Dublin, Ireland
Michael Philip Oakes	University of Sunderland, UK
Iadh Ounis	University of Glasgow, UK
Gabriella Pasi	University of Milano-Bicocca, Italy
Jan O. Pedersen	Microsoft, USA
Ari Pirkola	University of Tampere, Finland
Andreas Rauber	Vienna University of Technology, Austria
Thomas Roelleke	Queen Mary, University of London, UK
Dmitri Roussinov	University of Strathclyde, UK
Stefan Rürger	The Open University, UK
Ian Ruthven	University of Strathclyde, UK
Mark Sanderson	RMIT University, Australia
Rodrygo Santos	University of Glasgow, UK
Tamas Sarlos	Yahoo!, USA
Ralf Schenkel	Saarland University and Max-Planck-Institut für Informatik, Germany
Falk Scholer	RMIT University, Australia
Fabrizio Sebastiani	Consiglio Nazionale delle Ricerche, Italy
Florence Sedes	Université Paul Sabatier, France
Giovanni Semeraro	University of Bari, Italy
Milad Shokouhi	Microsoft Research Cambridge, UK
Mario J. Silva	University of Lisbon, Portugal
Fabrizio Silvestri	Italian National Research Council, Italy
Alan Smeaton	Dublin City University, Ireland
Mark D. Smucker	University of Waterloo, Canada
Barry Smyth	University College Dublin, Ireland
Vaclav Snasel	VSB Technical University of Ostrava, Czech Republic
Thomas Sødning	Oslo University College, Norway
Dawei Song	The Robert Gordon University, UK
Ruihua Song	Microsoft Research Asia, China
Eero Sormunen	University of Tampere, Finland
Amanda Spink	Loughborough University, UK
Nicola Stokes	University College Dublin, Ireland
Tomek Strzalkowski	SUNY Albany, USA
L Venkata Subramaniam	IBM India Research Lab, India
Hanna Suominen	NICTA, Canberra Research Laboratory and Australian National University, Australia
Lynda Tamine-Lechani	IRIT, France
Martin Theobald	Max-Planck-Institut für Informatik, Germany
Ulrich Thiel	Fraunhofer, Germany
Paul Thomas	CSIRO, Australia
Anastasios Tombros	Queen Mary University of London, UK

Theodora Tsirikla	Centrum Wiskunde & Informatica, The Netherlands
Pertti Vakkari	University of Tampere, Finland
Olga Vechtomova	University of Waterloo, Canada
Sumithra Velupillai	Stockholm University, Sweden
Jun Wang	University College London, UK
Nick Webb	SUNY Albany, USA
Ryen White	Microsoft Research, USA
Peter Wilkins	Google, Ireland
Tao Yang	University of California at Santa Barbara, USA
Roman Yangarber	University of Helsinki, Finland
Dell Zhang	Birkbeck, University of London, UK
Dong Zhou	Trinity College Dublin, Ireland
Jianhan Zhu	University College London, UK

### Additional Reviewers

M-Dyaa Albakour	University of Essex, UK
Robin Aly	University of Twente, The Netherlands
Jaime Arguello	Carnegie Mellon University, USA
Abhishek Arun	Bing Europe, Microsoft, UK
Pierpaolo Basile	University of Bari, Italy
David Batista	University of Lisbon, Portugal
Adam Bermingham	Dublin City University, Ireland
Claudio Biancalana	Roma Tre University, Italy
Roi Blanco	Yahoo! Research, Spain
Janez Brank	J. Stefan Institute, Slovenia
Beate Navarro Bullock	University of Würzburg, Germany
B. Barla Cambazoglu	Yahoo! Research, Spain
Annalina Caputo	University of Bari, Italy
Nuno Cardoso	University of Lisbon, Portugal
Paula Carvalho	University of Lisbon, Portugal
Karen Church	Telefonica, Barcelona, Spain
Miguel Costa	FCCN and University of Lisbon, Portugal
Ronan Cummins	University of Glasgow, UK
Lorand Dali	J. Stefan Institute, Slovenia
Marco de Gemmis	University of Bari, Italy
Marcus Duyzend	Bing Europe, Microsoft, UK
Carsten Eickhoff	Delft University of Technology, The Netherlands
Jill Freyne	CSIRO, Australia
Jagadeesh Gorla	University College London, UK
Katja Hofmann	University of Amsterdam, The Netherlands
Diego Fernández Iglesias	University of A Coruña, Spain
Tamas Jambor	University College London, UK
Richard Johansson	University of Trento, Italy

Leszek Kaliciak	The Robert Gordon University, UK
Kyriakos Karenos	Bing Europe, Microsoft, UK
Liadh Kelly	Dublin City University, Ireland
Phil Kelly	Dublin City University, Ireland
Christoph Kofler	Delft University of Technology, The Netherlands
Maheedhar Kolla	University of Waterloo, Canada
Bevan Koopman	Queensland University of Technology, Australia
Anagha Kulkarni	Carnegie Mellon University, USA
Leo Iaquinta	University of Bari, Italy
Chenghua Lin	University of Exeter, UK
Christina Lioma	University of Stuttgart, Germany
Elena Lloret	Universidad de Alicante, Spain
Vreixo Formoso López	University of A Coruña, Spain
Pasquale Lops	University of Bari, Italy
Rudolf Mayer	Vienna University of Technology, Austria
Kevin McCarthy	University College Dublin, Ireland
Kevin McGuinness	Dublin City University, Ireland
Cataldo Musto	University of Bari, Italy
Francesco Nidito	Bing Europe, Microsoft, UK
Igor Nitto	Bing Europe, Microsoft, UK
Monica Lestari Paramita	University of Sheffield, UK
Javier Parapa	University of A Coruña, Spain
Virgil Pavlu	Northeastern University, USA
Benjamin Piwowarski	University of Glasgow, UK
Stevan Rudinac	Delft University of Technology, The Netherlands
Delia Rusu	J. Stefan Institute, Slovenia
Hohyon Ryu	University of Texas at Austin, USA
Dave Sadlier	Dublin City University, Ireland
Giuseppe Sansonetti	Roma Tre University, Italy
Pavel Serdyukov	Delft University of Technology, The Netherlands
Yue Shi	Delft University of Technology, The Netherlands
Laurianne Sitbon	Queensland University of Technology, Australia
Ilija Subasic	Katholieke Universiteit Leuven, Belgium
Mike Symonds	Queensland University of Technology, Australia
Maria Alessandra Torsello	University of Bari, Italy
Mitja Trampung	J. Stefan Institute, Slovenia
Dolf Trieschnigg	University of Twente, The Netherlands
Frans van der Sluis	University of Twente, The Netherlands
Mathias Verbeke	Katholieke Universiteit Leuven, Belgium
Suzan Verberne	University of Nijmegen, The Netherlands
Jun Wang	The Robert Gordon University, UK

Lei Wang	The Robert Gordon University, UK
Robert H Warren	University of Waterloo, Canada
Fang Xu	Saarland University, Germany
Eunho Yang	University of Texas at Austin, USA
Emine Yilmaz	Microsoft Research Cambridge, UK
Yuan Yuan	Aston University, UK
Le Zhao	Carnegie Mellon University, USA
Peng Zhang	The Robert Gordon University, UK

## Tutorials Program Committee

Avi Arampatzis	Democritus University of Thrace, Greece
James Allan	University of Massachusetts, Amherst, USA
Ben Carterette	University of Delaware, USA
Djoerd Hiemstra	University of Twente, The Netherlands
Kalervo Järvelin	University of Tampere, Finland
Rosie Jones	Akamai, USA
Alistair Moffat	University of Melbourne, Australia
Maarten de Rijke	University of Amsterdam, The Netherlands
Emine Yilmaz	Microsoft Research Cambridge, UK

## Workshops Program Committee

Omar Alonso	Microsoft / Bing.com, USA
Giambattista Amati	Fondazione Ugo Bordoni, Italy
Fabio Crestani	University of Lugano, Switzerland
Bruce Croft	University of Massachusetts, USA
Udo Kruschwitz	University of Essex, UK
David Losada	University of Santiago de Compostella, Spain
Andrew MacFarlane	City University London, UK
Josiane Mothe	Institut de Recherche en Informatique de Toulouse, France
Milad Shokouhi	Microsoft Research Cambridge, UK
Paul Thomas	CSIRO, Australia

## Award Committee

- Best paper  
Charlie Clarke, University of Waterloo, Canada  
Norbert Fuhr, University of Duisburg-Essen,  
Germany  
Matt Lease, University of Texas at Austin, USA  
Alan Smeaton, Dublin City University, Ireland
- Best student paper  
Arjen P. de Vries, Centrum Wiskunde &  
Informatica, The Netherlands  
Gregory Grefenstette, EXALEAD, France  
Jim Jansen, Pennsylvania State University,  
USA
- Best poster  
Bettina Berendt, Katholieke Universiteit  
Leuven, Belgium  
Stefan Ruger, The Open University, UK  
Eero Sormunen, University of Tampere,  
Finland

## Previous Venues of ECIR

ECIR began life as the BCS-IRSG Annual Colloquium on Information Retrieval Research and was held in the UK until 1998 when Grenoble, France, hosted the first non-UK-based colloquium. This reflected the increasingly international nature and greater size of the colloquium. The annual event also changed its name in 2001 from the Annual Colloquium on Information Retrieval to the European Annual Colloquium on Information Retrieval. This was followed by a change to become ECIR - The European Conference on Information Retrieval in 2003, which reflected the fact that ECIR had grown to become a full-blown international conference with a European focus. In its history, ECIR has taken place in the following locations:

2010	Milton Keynes, UK
2009	Toulouse, France
2008	Glasgow, UK
2007	Rome, Italy
2006	London, UK
2005	Santiago de Compostela, Spain
2004	Sunderland, UK
2003	Pisa, Italy
2002	Glasgow, UK
2001	Darmstadt, Germany
2000	Cambridge, UK
1999	Glasgow, UK
1998	Grenoble, France
1997	Aberdeen, UK
1996	Manchester, UK
1995	Crewe, UK
1994	Drymen, UK
1993	Glasgow, UK
1992	Lancaster, UK
1991	Lancaster, UK
1990	Huddersfield, UK
1989	Huddersfield, UK
1988	Huddersfield, UK
1987	Glasgow, UK
1986	Glasgow, UK
1985	Bradford, UK
1984	Bradford, UK
1983	Sheffield, UK
1982	Sheffield, UK
1981	Birmingham, UK
1980	Leeds, UK
1979	Leeds, UK



## Sponsoring Institutions



The  
University  
Of  
Sheffield.

## In co-operation with



# Table of Contents

## Keynote Talks

IR Research: Systems, Interaction, Evaluation and Theories . . . . .	1
<i>Kalervo Järvelin</i>	
Ad Retrieval Systems <i>in vitro</i> and <i>in vivo</i> : Knowledge-Based Approaches to Computational Advertising . . . . .	4
<i>Evgeniy Gabrilovich</i>	
The Value of User Feedback . . . . .	6
<i>Thorsten Joachims</i>	

## Text Categorisation (I)

Text Classification for a Large-Scale Taxonomy Using Dynamically Mixed Local and Global Models for a Node . . . . .	7
<i>Heung-Seon Oh, Yoonjung Choi, and Sung-Hyon Myaeng</i>	
User-Related Tag Expansion for Web Document Clustering . . . . .	19
<i>Peng Li, Bin Wang, Wei Jin, and Yachao Cui</i>	
A Comparative Experimental Assessment of a Threshold Selection Algorithm in Hierarchical Text Categorization . . . . .	32
<i>Andrea Addis, Giuliano Armano, and Eloisa Vargiu</i>	

## Recommender Systems

Improving Tag-Based Recommendation by Topic Diversification . . . . .	43
<i>Christian Wartena and Martin Wubbels</i>	
A Joint Model of Feature Mining and Sentiment Analysis for Product Review Rating . . . . .	55
<i>Jorge Carrillo de Albornoz, Laura Plaza, Pablo Gervás, and Alberto Díaz</i>	
Modeling Answerer Behavior in Collaborative Question Answering Systems . . . . .	67
<i>Qiaoling Liu and Eugene Agichtein</i>	

**Web IR (I)**

Clash of the Typings: Finding Controversies and Children's Topics Within Queries .....	80
<i>Karl Gyllstrom and Marie-Francine Moens</i>	
Are Semantically Related Links More Effective for Retrieval? .....	92
<i>Marijn Koolen and Jaap Kamps</i>	
Caching for Realtime Search.....	104
<i>Edward Bortnikov, Ronny Lempel, and Kolman Vornovitsky</i>	
Enhancing Deniability against Query-Logs .....	117
<i>Avi Arampatzis, Pavlos S. Efraimidis, and George Drosatos</i>	

**IR Evaluation**

On the Contributions of Topics to System Evaluation.....	129
<i>Stephen Robertson</i>	
A Methodology for Evaluating Aggregated Search Results .....	141
<i>Jaime Arguello, Fernando Diaz, Jamie Callan, and Ben Carterette</i>	
Design and Implementation of Relevance Assessments Using Crowdsourcing.....	153
<i>Omar Alonso and Ricardo Baeza-Yates</i>	
In Search of Quality in Crowdsourcing for Search Engine Evaluation....	165
<i>Gabriella Kazai</i>	

**IR for Social Networks (I)**

Summarizing a Document Stream .....	177
<i>Hiroya Takamura, Hikaru Yokono, and Manabu Okumura</i>	
A Link Prediction Approach to Recommendations in Large-Scale User-Generated Content Systems .....	189
<i>Nitin Chiluka, Nazareno Andrade, and Johan Pouwelse</i>	
Topic Classification in Social Media Using Metadata from Hyperlinked Objects .....	201
<i>Sheila Kinsella, Alexandre Passant, and John G. Breslin</i>	
Peddling or Creating? Investigating the Role of Twitter in News Reporting .....	207
<i>Ilija Subašić and Bettina Berendt</i>	

**Cross-Language IR**

Latent Sentiment Model for Weakly-Supervised Cross-Lingual Sentiment Classification .....	214
<i>Yulan He</i>	
Fractional Similarity: Cross-Lingual Feature Selection for Search .....	226
<i>Jagadeesh Jagarlamudi and Paul N. Bennett</i>	
Is a Query Worth Translating: Ask the Users! .....	238
<i>Ahmed Hefny, Kareem Darwish, and Ali Alkakhky</i>	

**IR Theory (I)**

Balancing Exploration and Exploitation in Learning to Rank Online....	251
<i>Katja Hofmann, Shimon Whiteson, and Maarten de Rijke</i>	
reFER: Effective Relevance Feedback for Entity Ranking .....	264
<i>Tereza Iofciu, Gianluca Demartini, Nick Craswell, and Arjen P. de Vries</i>	
The Limits of Retrieval Effectiveness .....	277
<i>Ronan Cummins, Mounia Lalmas, and Colm O’Riordan</i>	
Learning Conditional Random Fields from Unaligned Data for Natural Language Understanding .....	283
<i>Deyu Zhou and Yulan He</i>	

**Text Categorisation (II)**

Subspace Tracking for Latent Semantic Analysis .....	289
<i>Radim Řehůřek</i>	
Text Retrieval Methods for Item Ranking in Collaborative Filtering ....	301
<i>Alejandro Bellogín, Jun Wang, and Pablo Castells</i>	
Classifying with Co-stems: A New Representation for Information Filtering .....	307
<i>Nedim Lipka and Benno Stein</i>	

**Multimedia IR**

Interactive Trademark Image Retrieval by Fusing Semantic and Visual Content .....	314
<i>Marçal Rusiñol, David Aldavert, Dimosthenis Karatzas, Ricardo Toledo, and Josep Lladós</i>	

Dynamic Two-Stage Image Retrieval from Large Multimodal Databases . . . . .	326
<i>Avi Arampatzis, Konstantinos Zagoris, and Savvas A. Chatzichristofis</i>	

## IR for Social Networks (II)

Comparing Twitter and Traditional Media Using Topic Models . . . . .	338
<i>Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li</i>	
Exploiting Thread Structures to Improve Smoothing of Language Models for Forum Post Retrieval . . . . .	350
<i>Huizhong Duan and Chengxiang Zhai</i>	
Incorporating Query Expansion and Quality Indicators in Searching Microblog Posts . . . . .	362
<i>Kamran Massoudi, Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp</i>	
Discovering Fine-Grained Sentiment with Latent Variable Structured Prediction Models . . . . .	368
<i>Oscar Täckström and Ryan McDonald</i>	

## IR Applications

Combining Global and Local Semantic Contexts for Improving Biomedical Information Retrieval . . . . .	375
<i>Duy Dinh and Lynda Tamine</i>	
Smoothing Click Counts for Aggregated Vertical Search . . . . .	387
<i>Jangwon Seo, W. Bruce Croft, Kwang Hyun Kim, and Joon Ho Lee</i>	
Automatic People Tagging for Expertise Profiling in the Enterprise . . . . .	399
<i>Pavel Serdyukov, Mike Taylor, Vishwa Vinay, Matthew Richardson, and Ryen W. White</i>	

## Text Categorisation (III)

Text Classification: A Sequential Reading Approach . . . . .	411
<i>Gabriel Dulac-Arnold, Ludovic Denoyer, and Patrick Gallinari</i>	
Domain Adaptation for Text Categorization by Feature Labeling . . . . .	424
<i>Cristina Kadar and José Iria</i>	

**IR for Social Networks (III)**

TEMPER : A Temporal Relevance Feedback Method . . . . .	436
<i>Mostafa Keikha, Shima Gerani, and Fabio Crestani</i>	
Terms of a Feather: Content-Based News Recommendation and Discovery Using Twitter . . . . .	448
<i>Owen Phelan, Kevin McCarthy, Mike Bennett, and Barry Smyth</i>	
Topical and Structural Linkage in Wikipedia . . . . .	460
<i>Kelly Y. Itakura, Charles L.A. Clarke, Shlomo Geva, Andrew Trotman, and Wei Chi Huang</i>	

**Web IR (II)**

An Analysis of Time-Instability in Web Search Results . . . . .	466
<i>Jinyoung Kim and Vitor R. Carvalho</i>	
Rules of Thumb for Information Acquisition from Large and Redundant Data . . . . .	479
<i>Wolfgang Gatterbauer</i>	
Bringing Why-QA to Web Search . . . . .	491
<i>Suzan Verberne, Lou Boves, and Wessel Kraaij</i>	
The Power of Peers . . . . .	497
<i>Nick Craswell, Dennis Fetterly, and Marc Najork</i>	
Introducing the User-over-Ranking Hypothesis . . . . .	503
<i>Benno Stein and Matthias Hagen</i>	
Second Chance: A Hybrid Approach for Dynamic Result Caching in Search Engines . . . . .	510
<i>I. Sengor Altıngövdü, Rifat Özcan, B. Barla Cambazoglu, and Özgür Ulusoy</i>	

**IR Theory (II)**

Learning Models for Ranking Aggregates . . . . .	517
<i>Craig Macdonald and Iadh Ounis</i>	
Efficient Compressed Inverted Index Skipping for Disjunctive Text-Queries . . . . .	530
<i>Simon Jonassen and Svein Erik Bratsberg</i>	
Within-Document Term-Based Index Pruning with Statistical Hypothesis Testing . . . . .	543
<i>Sree Lekha Thota and Ben Carterette</i>	

SkipBlock: Self-indexing for Block-Based Inverted List . . . . .	555
<i>Stéphane Campinas, Renaud Delbru, and Giovanni Tummarello</i>	
Weight-Based Boosting Model for Cross-Domain Relevance Ranking Adaptation . . . . .	562
<i>Peng Cai, Wei Gao, Kam-Fai Wong, and Aoying Zhou</i>	

## Interactive IR

What Makes Re-finding Information Difficult? A Study of Email Re-finding . . . . .	568
<i>David Elsweiler, Mark Baillie, and Ian Ruthven</i>	
A User-Oriented Model for Expert Finding . . . . .	580
<i>Elena Smirnova and Krisztian Balog</i>	
Simulating Simple and Fallible Relevance Feedback . . . . .	593
<i>Feza Baskaya, Heikki Keskustalo, and Kalervo Järvelin</i>	
AutoEval: An Evaluation Methodology for Evaluating Query Suggestions Using Query Logs . . . . .	605
<i>M-Dyaa Albakour, Udo Kruschwitz, Nikolaos Nanas, Yunhyong Kim, Dawei Song, Maria Fasli, and Anne De Roeck</i>	
To Seek, Perchance to Fail: Expressions of User Needs in Internet Video Search . . . . .	611
<i>Christoph Kofler, Martha Larson, and Alan Hanjalic</i>	

## Question Answering / NLP

Passage Reranking for Question Answering Using Syntactic Structures and Answer Types . . . . .	617
<i>Elif Aktolga, James Allan, and David A. Smith</i>	
An Iterative Approach to Text Segmentation . . . . .	629
<i>Fei Song, William M. Darling, Adnan Duric, and Fred W. Kroon</i>	
Improving Query Focused Summarization Using Look-Ahead Strategy . . . . .	641
<i>Rama Badrinath, Suresh Venkatasubramanian, and C.E. Veni Madhavan</i>	
A Generalized Method for Word Sense Disambiguation Based on Wikipedia . . . . .	653
<i>Chenliang Li, Aixin Sun, and Anwitaman Datta</i>	

## Posters

Representing Document Lengths with Identifiers . . . . .	665
<i>Raffaele Perego, Fabrizio Silvestri, and Nicola Tonellotto</i>	
Free-Text Search versus Complex Web Forms . . . . .	670
<i>Kien Tjin-Kam-Jet, Dolf Trieschnigg, and Djoerd Hiemstra</i>	
Multilingual Log Analysis: LogCLEF . . . . .	675
<i>Giorgio Maria Di Nunzio, Johannes Leveling, and Thomas Mandl</i>	
A Large-Scale System Evaluation on Component-Level . . . . .	679
<i>Jens Kürsten and Maximilian Eibl</i>	
Should MT Systems Be Used as Black Boxes in CLIR? . . . . .	683
<i>Walid Magdy and Gareth J.F. Jones</i>	
Video Retrieval Based on Words-of-Interest Selection . . . . .	687
<i>Lei Wang, Dawei Song, and Eyad Elyan</i>	
Classic Children's Literature – Difficult to Read? . . . . .	691
<i>Dolf Trieschnigg and Claudia Hauff</i>	
Applying Machine Learning Diversity Metrics to Data Fusion in Information Retrieval . . . . .	695
<i>David Leonard, David Lillis, Lusheng Zhang, Fergus Toolan, Rem W. Collier, and John Dunnion</i>	
Reranking Collaborative Filtering with Multiple Self-contained Modalities . . . . .	699
<i>Yue Shi, Martha Larson, and Alan Hanjalic</i>	
How Far Are We in Trust-Aware Recommendation? . . . . .	704
<i>Yue Shi, Martha Larson, and Alan Hanjalic</i>	
Re-ranking for Multimedia Indexing and Retrieval . . . . .	708
<i>Bahjat Safadi and Georges Quénot</i>	
Combining Query Translation Techniques to Improve Cross-Language Information Retrieval . . . . .	712
<i>Benjamin Herbert, György Szarvas, and Iryna Gurevych</i>	
Back to the Roots: Mean-Variance Analysis of Relevance Estimations . . .	716
<i>Guido Zuccon, Leif Azzopardi, and Keith van Rijsbergen</i>	
A Novel Re-ranking Approach Inspired by Quantum Measurement . . . . .	721
<i>Xiaozhao Zhao, Peng Zhang, Dawei Song, and Yuexian Hou</i>	
Simple vs. Sophisticated Approaches for Patent Prior-Art Search . . . . .	725
<i>Walid Magdy, Patrice Lopez, and Gareth J.F. Jones</i>	



Towards Quantum-Based DB+IR Processing Based on the Principle of Polyrepresentation . . . . .	729
<i>David Zellhöfer, Ingo Frommholz, Ingo Schmitt, Mounia Lalmas, and Keith van Rijsbergen</i>	
ATTention: Understanding Authors and Topics in Context of Temporal Evolution . . . . .	733
<i>Nasir Naveed, Sergej Sizov, and Steffen Staab</i>	
Role of Emotional Features in Collaborative Recommendation . . . . .	738
<i>Yashar Moshfeghi and Joemon M. Jose</i>	
The Importance of the Depth for Text-Image Selection Strategy in Learning-To-Rank . . . . .	743
<i>David Buffoni, Sabrina Tollari, and Patrick Gallinari</i>	
Personal Blog Retrieval Using Opinion Features . . . . .	747
<i>Shima Gerani, Mostafa Keikha, Mark Carman, and Fabio Crestani</i>	
Processing Queries in Session in a Quantum-Inspired IR Framework . . . .	751
<i>Ingo Frommholz, Benjamin Piwowarski, Mounia Lalmas, and Keith van Rijsbergen</i>	
Towards Predicting Relevance Using a Quantum-Like Framework . . . . .	755
<i>Emanuele Di Buccio, Massimo Melucci, and Dawei Song</i>	
Fusion vs. Two-Stage for Multimodal Retrieval . . . . .	759
<i>Avi Arampatzis, Konstantinos Zagoris, and Savvas A. Chatzichristofis</i>	
Combination of Feature Selection Methods for Text Categorisation . . . . .	763
<i>Robert Neumayer, Rudolf Mayer, and Kjetil Nørkvåg</i>	

## Demonstrations

Time-Surfer: Time-Based Graphical Access to Document Content . . . . .	767
<i>Hector Llorens, Estela Saquete, Borja Navarro, and Robert Gaizauskas</i>	
ARES: A Retrieval Engine Based on Sentiments: Sentiment-Based Search Result Annotation and Diversification . . . . .	772
<i>Gianluca Demartini</i>	
Web Search Query Assistance Functionality for Young Audiences . . . . .	776
<i>Carsten Eickhoff, Tamara Polajnar, Karl Gyllstrom, Sergio Duarte Torres, and Richard Glassey</i>	

Conversation Retrieval from Twitter . . . . .	780
<i>Matteo Magnani, Danilo Montesi, Gabriele Nunziante, and Luca Rossi</i>	
Finding Useful Users on Twitter: Twittomender the Followee Recommender . . . . .	784
<i>John Hannon, Kevin McCarthy, and Barry Smyth</i>	
Visual Exploration of Health Information for Children . . . . .	788
<i>Frans van der Sluis, Sergio Duarte Torres, Djoerd Hiemstra, Betsy van Dijk, and Frea Kruisinga</i>	
<b>Author Index</b> . . . . .	793

# IR Research: Systems, Interaction, Evaluation and Theories

Kalervo Järvelin

School of Information Sciences  
University of Tampere, Finland  
kalervo.jarvelin@uta.fi

**Abstract.** The ultimate goal of information retrieval (IR) research is to create ways to support humans to better access information in order to better carry out their (work) tasks. Because of this, IR research has a primarily technological interest in knowledge creation – how to find information (better)? IR research therefore has a constructive aspect (to create novel systems) and an evaluative aspect (are they any good?). Evaluation is sometimes referred to as a hallmark and distinctive feature of IR research. No claim on IR system performance is granted any merit unless proven through evaluation. Technological innovation alone is not sufficient. In fact, much research in IR deals with IR evaluation and its methodology.

Evaluation, in general, is the systematic determination of merit and significance of something using criteria against some standards. Evaluation therefore requires some object that is evaluated and some goal that should be achieved or served. In IR, both can be set in many ways. The object usually is an IR system – but what is an IR system? The goal is typically the quality of the retrieved result – but what is the retrieved result and how does one measure quality? These questions can be answered in alternative ways leading to different kinds of IR evaluation.

Practical life with all its variability is difficult and expensive to investigate. Therefore surrogate and more easily measurable goals are employed in IR evaluation, typically the quality of the ranked result list instead of the work task result. The task performance process may also be cut down from a work task to a search task and down to running an individual query in a test collection. This simplification has led to standardization of research designs and tremendous success in IR research. However, as the goals and systems drift farther away from the practical life condition, one needs to ask, whether the findings still best serve the initial goal of evaluation (supporting human performance)? If means (outputs) replace ends (outcomes), one runs the risk of sub-optimization.

It is important to evaluate all subsystems of information retrieval processes, in addition to the search engines. Through a wider perspective one may be able to put the subsystems and their contributions in relation with each other. We will discuss nested IR evaluation frameworks ranging from IR system centered evaluation to work-task based evaluation. We will also point to the Pandora's box of problems that the enlargement of the scope of research entails. Is science at risk here?

The contributions of a research area, in addition to constructive and evaluative contributions, may be generally empirical, theoretical and

methodological. Why should anyone in IR care about anything beyond IR experimentation (i.e. evaluation) using test collections? The Cranfield model seeks to relate texts (documents), queries, their representations and matching to topical relevance in ranked output. Who relates this, and a range of possible other contributing factors, to outcomes in search task performance or work task performance? The talk will outline some possibilities for descriptive, explanatory and theoretical research in IR. As an example of descriptive research, we will discuss information access in task processes. Regarding explanatory and theoretical research, we look at unit theories that connect work task stages and properties to information need properties, information sources, and searching. Such studies do not solve a technical problem, nor evaluate any particular technique, and may therefore be considered unpractical. However, they may identify mechanisms that mediate between IR processes and task outcomes and position factors in the processes of information access into a balanced perspective. Therefore they may help focus research efforts on technical problems or evaluation.

## Short Bio

Kal Järvelin (<http://www.uta.fi/~likaja>) is Professor and Vice Dean at the School of Information Sciences, University of Tampere, Finland. He holds a PhD in Information Studies (1987) from the same university. He was Academy Professor, Academy of Finland, in 2004–2009.

Järvelin's research covers information seeking and retrieval, linguistic and conceptual methods in IR, IR evaluation, and database management. He has coauthored over 250 scholarly publications and supervised sixteen doctoral dissertations. He has been a principal investigator of several research projects funded by the Academy of Finland in particular, by EU and industry.

Järvelin has frequently served the ACM SIGIR Conferences as a program committee member (1992-2009), Conference Chair (2002) and Program Co-Chair (2004, 2006); and the ECIR, the ACM CIKM and many other conferences as pc member. He is an Associate Editor of *Information Processing and Management* (USA).

Järvelin received the Finnish Computer Science Dissertation Award 1986; the ACM SIGIR 2000 Best Paper Award<sup>1</sup> for the seminal paper on the discounted cumulated gain evaluation metric; the ECIR 2008 Best Paper Award<sup>1</sup> for session-based IR evaluation; the IiX 2010 Best Paper Award<sup>1</sup> for a study on task-based information access; and the Tony Kent Strix Award 2008 in recognition of contributions to the field of information retrieval.

## References

1. Byström, K., Järvelin, K.: Task Complexity Affects Information Seeking and Use. *Information Processing & Management* 31(2), 191–213 (1995)
2. Ingwersen, P., Järvelin, K.: *The Turn: Integration of Information Seeking and Retrieval in Context*. Springer, Dordrecht (2005)

---

<sup>1</sup> All with co-authors.

3. Järvelin, K.: An Analysis of Two Approaches in Information Retrieval: From Frameworks to Study Designs. *Journal of the American Society for Information Science and Technology (JASIST)* 58(7), 971–986 (2007)
4. Järvelin, K.: Explaining User Performance in Information Retrieval: Challenges to IR evaluation. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) *ICTIR 2009. LNCS*, vol. 5766, pp. 289–296. Springer, Heidelberg (2009)
5. Kumpulainen, S., Järvelin, K.: Information Interaction in Molecular Medicine: Integrated Use of Multiple Channels. In: Belkin, N., et al. (eds.) *Proceedings of the Information Interaction in Context Conference (IiX 2010)*, pp. 95–104. ACM, New York (2010)
6. Vakkari, P.: A theory of the task-based information retrieval process: A summary and generalization of a longitudinal study. *Journal of Documentation* 57(1), 44–60 (2001)

# Ad Retrieval Systems *in vitro* and *in vivo*: Knowledge-Based Approaches to Computational Advertising

Evgeniy Gabrilovich

Yahoo! Research, California, USA  
gabr@yahoo-inc.com

**Abstract.** Over the past decade, online advertising became the principal economic force behind many an Internet service, from major search engines to globe-spanning social networks to blogs. There is often a tension between online advertising and user experience, but on the other hand, advertising revenue enables a myriad of free Web services to the public and fosters a great deal of innovation. Matching the advertisers' message to a receptive and interested audience benefits both sides; indeed, literally hundreds of millions of users occasionally click on the ads, hence they should be considered relevant to the users' information needs by current IR evaluation principles. The utility of ads can be better explained by considering advertising as a medium of information [2,3]. Similarly to aggregated search [1], which enhances users' Web search experience with relevant news, local results, user-generated content, or multimedia, online advertising provides another rich source of content. This source, however, is in a complexity class of its own, due to the brevity of bid phrases, ad text being optimized for presentation rather than indexing, and multiple, possibly contradictory utility functions.

A new scientific sub-discipline—Computational Advertising—has recently emerged, which strives to make online advertising integral to the user experience and relevant to the users' information needs, as well as economically worthwhile to the advertiser and the publisher. In this talk we discuss the unique algorithmic challenges posed by searching the ad corpus, and report on empirical evaluation of large-scale advertising systems *in vivo*. At first approximation, finding user-relevant ads is akin to ad hoc information retrieval, where the user context is distilled into a query executed against an index of ads. However, the elaborate structure of ad campaigns, along with the cornucopia of pertinent non-textual information, makes ad retrieval substantially and interestingly different. We show how to adapt standard IR methods for ad retrieval, by developing structure-aware indexing techniques and by augmenting the ad selection process with exogenous knowledge. Computational advertising also employs a host of NLP techniques, from text summarization for just-in-time ad matching, to machine translation for cross-language ad retrieval, to natural language generation for automatic construction of advertising campaigns. Last but not least, we study the interplay between the algorithmic and sponsored search results, as well as formulate and explore context transfer, which characterizes the user's transition from

Web search to the context of the landing page following an ad-click. These studies offer deep insights into how users interact with ads, and facilitate better understanding of the much broader notion of relevance in ad retrieval compared to Web search.

**Biography:** Evgeniy Gabrilovich is a Senior Research Scientist and Manager of the NLP & IR Group at Yahoo! Research, where he works on advancing the state of the art in information retrieval and natural language processing at Web scale. He has filed over 20 patents, and has over 40 publications in top international venues such as ACM TOIS, JMLR, JAIR, WWW, and WSDM. Evgeniy served as a Senior PC member or Area Chair at SIGIR, AACL, IJCAI, EMNLP, and ICWSM, and also served on the program committees of virtually every major conference in the field.

Evgeniy earned his MSc and PhD degrees in Computer Science from the Technion—Israel Institute of Technology. In his PhD thesis, he developed a methodology for using large scale repositories of world knowledge (e.g., all the knowledge available in Wikipedia) to enhance text representation beyond the bag of words. Subsequently, he also worked on combining heterogeneous knowledge sources, such as Web search results, classification with respect to an external taxonomy, and users' actions, for improving the accuracy of information retrieval. Evgeniy researched how to interpret information in context, and devised methods for analyzing the context of user queries. He developed one of the most commonly used datasets for computing semantic relatedness of words, as well as formulated a methodology for parameterized generation of labeled datasets for text categorization. Evgeniy taught multiple tutorials on computational advertising and organized a number of workshops at top conferences, including a workshop on feature generation and selection for information retrieval at SIGIR, and workshops on the synergy between user-contributed knowledge and research in AI at IJCAI and AACL.

## References

1. Arguello, J., Diaz, F., Callan, J., Crespo, J.-F.: Sources of evidence for vertical selection. In: Proceedings of the 31st Annual International ACM SIGIR Conference, pp. 315–322 (2009)
2. Nelson, P.: Advertising as information. *Journal of Political Economy* 82(4), 729–754 (1974)
3. Ogilvy, D.: *Ogilvy on Advertising*. Vintage Books, London (1985)

# The Value of User Feedback

Thorsten Joachims

Cornell University,  
Department of Computer Science,  
Ithaca, NY, USA  
tj@cs.cornell.edu,  
www.joachims.org

**Abstract.** Information retrieval systems and their users engage in a dialogue. While the main flow of information is from the system to the user, feedback from the user to the system provides many opportunities for short-term and long-term learning. In this talk, I will explore two interrelated questions that are central to the effective use of feedback. First, how can user feedback be collected so that it does not lay a burden on the user? I will argue that the mechanisms for collecting feedback have to be integrated into the design of the retrieval process, so that the user's short-term goals are well-aligned with the system's goal of collecting feedback. Second, if this integration succeeds, how valuable is the information that user feedback provides? For the tasks of retrieval evaluation and query disambiguation, the talk will quantify by how much user feedback can save human annotator effort and improve retrieval quality respectively.

**Bio.** Thorsten Joachims is an Associate Professor in the Department of Computer Science at Cornell University. In 2001, he finished his dissertation with the title "The Maximum-Margin Approach to Learning Text Classifiers: Methods, Theory, and Algorithms", advised by Prof. Katharina Morik at the University of Dortmund. From there he also received his Diplom in Computer Science in 1997 with a thesis on WebWatcher, a browsing assistant for the Web. From 1994 to 1996 he was a visiting scientist at Carnegie Mellon University with Prof. Tom Mitchell. His research interests center on a synthesis of theory and system building in the field of machine learning, with a focus on support vector machines, structured output prediction, and learning algorithms for information retrieval.



# Text Classification for a Large-Scale Taxonomy Using Dynamically Mixed Local and Global Models for a Node

Heung-Seon Oh, Yoonjung Choi, and Sung-Hyon Myaeng

Department of Computer Science,  
Korea Advanced Institute of Science and Technology  
{ohs,choiyj35,myaeng}@kaist.ac.kr

**Abstract.** Hierarchical text classification for a large-scale Web taxonomy is challenging because the number of categories hierarchically organized is large and the training data for deep categories are usually sparse. It's been shown that a narrow-down approach involving a search of the taxonomical tree is an effective method for the problem. A recent study showed that both local and global information for a node is useful for further improvement. This paper introduces two methods for mixing local and global models dynamically for individual nodes and shows they improve classification effectiveness by 5% and 30%, respectively, over and above the state-of-art method.

**Keywords:** Web Taxonomy, Hierarchical Text Classification, ODP.

## 1 Introduction

Web taxonomy is a large-scale hierarchy of categories where each category is represented by a set of web documents. Utilizing web taxonomies has been shown useful for several tasks such as rare query classification, contextual advertising, and web search improvement [1,2,3]. For example, query classification to a web taxonomy can help detecting a specific topic of interest, which can lead to more relevant online ads than those searched by the query directly. Despite its usefulness, text classification for a large-scale web taxonomy is challenging because the number of categories organized in a big hierarchy is large and the training data for deep categories are usually sparse. As such, the traditional text classification methods relying on statistical machine learning alone are not satisfactory especially when the classification task involves web taxonomies such as ODP and Yahoo! Directories that have hundreds of thousands of categories and millions of documents distributed unevenly.

Many methods have been developed and studied to deal with the hierarchical text classification problem [4,5,6,7,8,9,10,11,12,13,14,15]. Compared to the previous research focusing on machine learning algorithm alone, a narrow-down approach was proposed by introducing a *deep classification* algorithm in [8]. This algorithm consists of two stages: search and classification. In the search stage, several candidate categories that are highly related to the input document are retrieved from the entire hierarchy using a search technique. Then training data are collected for each candidate by considering the documents associated with the node and those with its ancestor category nodes. Finally classification is performed after training a classifier for the candidate

categories using the training documents. By focusing on highly relevant categories from the entire hierarchy, this approach can alleviate error propagation and time complexity problems of a pure machine-learning based algorithm.

Despite the improved performance, the afore mentioned approach suffers from a relatively small number of training documents that are local to a category in a huge hierarchy. As a remedy, an enhanced algorithm was proposed, which makes use of the path from the candidate category node to the root of the hierarchy. Not only the documents associated with the candidate category (local information) but also the documents associated with the top-level categories (global information) connected to the candidate node are used to enrich the training data. Experimental results showed a remarkable performance improvement [15].

In this work, a fixed mixture weight was applied universally to each category node inmodulating the relative contributions of local and global models. However, we note the role of global information may vary depending on the richness of local information. Suppose there are two candidate categories  $D$  and  $D'$  that are highly similar to each other based on the associated documents but have entirely different paths:  $A/B/C/D$  and  $E/F/G/D'$ , where  $A$  and  $E$  are the roots of two sub-trees in the hierarchy, respectively. If a fixed mixture weight is applied to combine information associated with  $A$  (global) and  $D$  (local) and  $E$  (global) and  $D'$ (local), it ignores relative importance of global and local information in each case. This paper proposes efficient and effectivemethods for determining the mixture weight automatically in this context, which avoid EM-like time-consuming updates.

The rest of this paper is organized as follows. Section 2 delivers relevant work briefly. The details of the proposed methods are explained in Section 3 and 4. Section 5 presents the experimental results and error analysis of the results. Finally we conclude this paper with discussion and future work in Section 6.

## 2 Related Work

Four types of approaches have been proposed to deal with hierarchical text classification: big-bang, shrinkage, top-down and narrow-down approaches. In big-bang approaches, a single classifier is trained for the entire set of categories in a hierarchy. To build a classifier, various algorithms have been utilized, including SVM [10], Centroid-based [11], rule-based [12], and association rule-based [13] approaches. It's been shown that it takes much more time in a big-bang approach than in a top-down one even when a hierarchy contains thousands of categories [14]. With hundreds of thousands of categories, moreover, building a single classifier for large-scale hierarchy is intractable [6].

In a top-down approach, a classifier is trained for each level of a category hierarchy. Several studies adapted a top-down approach with various algorithms such as multiple Bayesian[16] and SVM [6,7]. In [6], it reports that a top-down based SVM classifier results in big performance drops on Yahoo! Directory as it goes to deeper levels. This is caused by propagations of errors at early stages involving the ancestors in classifications. In [7], two refinement methods using cross-validation and meta-features which are the predication of children of a target node are introduced to deal with error propagation and nonlinearity of general concepts in high level.

In [9], shrinkage approach was proposed for hierarchical classification. Probabilities of words for a class corresponding to a leaf node are calculated for the parents up to the root and combined with a set of parameters. This method of dealing with the data sparseness problem is similar to our method in that it uses the data on the ancestor nodes. However, our method stops gathering training data right before a shared ancestor while the shrinkage approach uses all the parent data up to the root of a hierarchy. The shrinkage method requires heavy computation not only because of the need to consider all the data on the path to the root but also because of the time for parameter estimations with the EM algorithm. By focusing on the top- and leaf-level information, our method reduces time complexity considerably.

A recent study proposed the narrow-down approach[8]. The algorithm consists of search and classification stages. In classification, the algorithm just focused on local information by utilizing trigram language model classifier. In [15], an enhanced algorithm of deep classification is proposed by introducing neighbor-assistant training data selection and the idea of combining local and global information based on naïve Bayes classifier. While it showed the possibility of augmenting local with global information, it used a fixed mixture weight for all candidate categories.

### 3 Search Stage

The *enhanced deep classification* approach consists of search and classification stages. For a document to be classified, top-k candidate categories are retrieved at the search stage. In order to select training data, a local model is constructed with the documents associated with each candidate category. Similarly, a global model is constructed for a top-level category connected to the candidate node in the given hierarchy. Finally, classification is performed using the classifier trained by the local and global models constructed based on a dynamically computed mixture weight.

The aim of the search stage is to narrow down the entire hierarchy to several highly related categories by using search technique. Two strategies are proposed in [8]: document based and category based search strategies. In the document based strategy, the retrieval unit is a document. Simply we index each document and obtain top-k relevant documents based on relevance scores. Then, the corresponding categories of the retrieved documents are taken as candidate categories. In the category based strategy, we first make a mega document by merging the documents belonging to a category and build an inverted index with a set of mega documents. This ensures a one to one relation between categories and mega documents. Then the search is performed with respect to the given document and produces top-k relevant mega documents. The corresponding categories are taken as candidates since a mega document only links to a unique category. We chose Lucene<sup>1</sup> among several publicly available search engines with the category based strategy which outperforms the document based strategy[8].

### 4 Classification Stage

At the classification stage, a final category among the candidate categories is assigned to the given document. Since a classifier is trained for candidate categories, a set of

---

<sup>1</sup> <http://lucene.apache.org>

training data must be collected from the hierarchy. This section explains how we dynamically collect the training data, i.e., the documents somehow associated with the candidate categories.

#### 4.1 Training Data Selection

For the narrow-down approach, two training data selection strategies have been proposed in [8] and [15]: ancestor-assistant and neighbor-assistant strategies. The common purpose of these strategies is to alleviate data sparseness in deep categories. The ancestor-assistant strategy utilizes the structure of the hierarchy. For each candidate, it borrows the documents of its ancestors all the way up until a common ancestor shared by other candidate node appear in the hierarchy. The neighbor-assistant strategy, an extension of the ancestor-assistant strategy, borrows documents not only from the ancestors but also from the children of the ancestors. The neighbor-assistant strategy produces slightly better performance than the ancestor-assistant strategy but requires more time [15]. We opted for the ancestor-assistant strategy in the paper for the sake of reduced time complexity.

#### 4.2 Classifier

Given that training must be done for each document to be classified, a lightweight classifier is preferred to those that require longer training time. For this reason, we opted for a naïve Bayes classifier (NBC) in this work. NBC estimates the posterior probability of a test or input document as follows:

$$P(c_i | d) = \frac{P(d | c_i)P(c_i)}{P(d)} \propto P(c_i) \prod_{j=1}^N P(t_j | c_i)^{v_j} \quad (1)$$

where  $c_i$  is a category  $i$ ,  $d$  is a test document,  $N$  is the vocabulary size,  $t_j$  is a term  $j$  in  $d$  and  $v_j$  is term frequency of  $t_j$ .

NBC assigns a category to a document as follows:

$$c^* = \arg \max_{c_i \in C} \{ P(c_i) \prod_{j=1}^N P(t_j | c_i)^{v_j} \} \quad (2)$$

In general, NBC estimates  $P(c_i)$  and  $P(t_j | c_i)$  from the entire collection  $D$  of training data. In deep classification, however,  $D$  denotes the training data from candidate categories. Since both local and global documents are used for training, the probabilities are computed separately and then combined with a mixture weight as follows.

$$P(t_j | c_i) = (1 - \lambda_i)P(t_j | c_i^{global}) + \lambda_i P(t_j | c_i^{local}) \quad (3)$$

$$P(c_i) = (1 - \lambda_i)P(c_i^{global}) + \lambda_i P(c_i^{local}) \quad (4)$$

where  $c_i^{global}$  is the top-level category of  $c_i$  and  $c_i^{local}$  is the same as  $c_i$  but rephrased for explicit explanation, i.e.  $c_i^{global} = A$  and  $c_i^{local} = A/B/C$  for  $c_i = A/B/C$ .

### 4.3 Global and Local Models

The aim of the global model is to utilize information in the top-level categories under the root node. Because of the topical diversity of the categories in a hierarchy, the vocabulary size is very large although the number of training data is usually small. To alleviate this problem in representing top-level categories, 15,000 terms are selected using the chi-square feature selection method, which is known for the best performing one in text classification [17].

The prior probability is estimated as follows:

$$P(c_i^{global}) = \frac{|D_i|}{|D|} \quad (5)$$

where  $D$  is the entire document collection and  $D_i$  is a sub-collection in  $c_i^{global}$ . The conditional probability is estimated by mixture of  $P(t_j | c_i^{global})$  and  $P(t_j)$  to avoid zero probabilities.

$$P(t_j | c_i^{global}) = (1 - \alpha) P_{global}(t_j | c_i^{global}) + \alpha P_{global}(t_j) \quad (6)$$

where  $\alpha$  is a mixture weight ( $0 \leq \alpha \leq 1$ ). Two parameters are estimated as follows:

$$P_{global}(t_j | c_i^{global}) = \frac{\sum_{d_k \in c_i^{global}} tf_{jk}}{\sum_{t_u \in V^{global}} \sum_{d_k \in c_i^{global}} tf_{uk}} \quad (7)$$

$$P_{global}(t_j) = \frac{\sum_{d_i \in D} tf_{jk}}{\sum_{t_u \in V^{global}} \sum_{d_i \in D} tf_{uk}} \quad (8)$$

where  $tf_{ji}$  is the term frequency of term  $t_j$  in top-level category  $c_i^{global}$  and  $V^{global}$  is a set of terms selected by the chi-square feature selection method over the entire document collection.

The role of our local model is to estimate the parameters of candidate categories directly from the set of documents associated with them. Unlike the global model we don't limit the feature space using a feature selection method. The time required for feature selection is not tolerable given that generating a local model cannot be done off-line. The terms selected for the global model cannot be used for the local model either because a limited number of terms concentrated in a semantic category is used in deep categories. Parameter estimation is similar to that of the global model except that it is done for selected training data corresponding to candidate categories.

### 4.4 Dynamic Determination of a Mixture Weight

The main step in classification is to properly combine local and global models for each candidate category node by determining a proper mixture weight. The shrinkage approach proposed in the past finds a mixture weight using a simple form of expectation maximization (EM) algorithm [9]. Unfortunately, this method is not applicable to a large-scale web taxonomy. Too many parameters are involved in

estimating a mixture weight because a word is generated conditioned on different categories and levels. For example, ODP has hundreds of thousands of terms, millions of categories, and 15 levels of depth. As a result, heavy computation is required if the EM algorithm is used for training that must be done for individual documents. Even though only a small number of candidate categories are chosen, the cost is prohibitive as we should dynamically compute mixture weights for different input documents.

As an alternative to the approach, we propose two methods that determine the mixture weights dynamically for individual documents to be classified. They are used to optimize the weights for individual documents, instead of using a “one-for-all” type of weight.

**Content-Based Optimization (CBO).** The main idea is to utilize the difference between local and global models in terms of their semantic contents that can be estimated based on word frequencies. The similarity can be computed based on the probability distributions of words in the two models. Given a candidate category, we don’t need to give a high weight to the global model if it is similar to the local model. For example, when a classification decision is to be made between  $D$  and  $D'$  having paths  $A/B/C/D$  and  $E/F/G/D'$ , respectively, there is no point of giving a high weight to the top level categories,  $A$  and  $E$ , if  $D$  and  $D'$  are similar to them, respectively. In this case, it is reasonable to focus on local information for classification. We capture this intuition with the following equation.

$$\lambda_i = 1 - \frac{\sum_j P(t_j | c_i^{local}) \cdot P(t_j | c_i^{global})}{\sqrt{\sum_j P(t_j | c_i^{local})^2} \sqrt{\sum_j P(t_j | c_i^{global})^2}} \quad (9)$$

**Relevance-Based Optimization (RBO).** The main idea here is to utilize the relevance scores of candidate categories obtainable from the search stage. The higher the relevance score of a category is in comparison with others, the higher weight it is given. Note that relevance scores are calculated with no regard to the hierarchy information since all the documents under a category is treated as a mega document regardless of its position in the hierarchy. In this work, we use cosine similarity between the input document to be classified and the set of documents belonging to the candidate category at hand. Based on this interpretation, a mixture weight is calculated as follows:

$$\lambda_i = \frac{RelScore(c_i^{local})}{\sum_k RelScore(c_k^{local})} \quad (10)$$

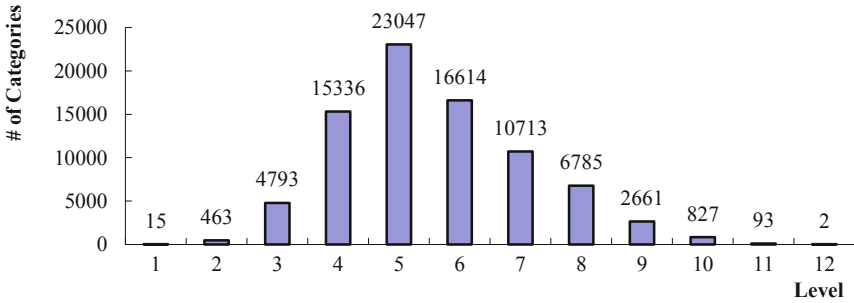
where  $k$  is the number of candidate categories for the initial search result.

## 5 Experiments

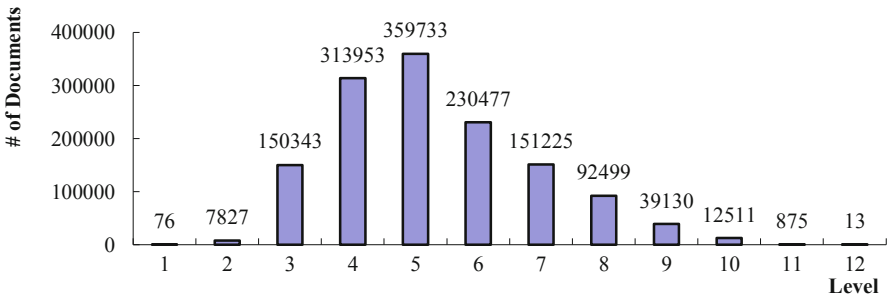
### 5.1 Experimental Set-Up

**Data.** ODP has about 700K categories and 4.8M documents with 17 top-level categories: *Arts, Business, Computer, Games, Health, Home, Kids\_and\_Teens, News,*

*Recreation, Reference, Regional, Science, Shopping, Society, Sports, World, and Adult*. Among them, we filtered out *World* and *Regional* categories since documents in the *Regional* category exist in other categories and those in the *World* category are not written in English [15]. In addition, the categories having non-English documents were also discarded. As a result, the total number of top-level categories is 15 in our experiments. For leaf categories, when the names are just enumeration of the alphabets such as “A”, “B”, ..., and “Z”, we merged them to their parent category. For example, documents that belong to *.../Fan\_Works/Fan\_Art/Mcategory* are merged to *.../Fan\_Works/Fan\_Art* category. Finally, the categories with less than two documents were discarded.



**Fig. 1.** Category Distributions at Different Levels in the Filtered ODP



**Fig. 2.** Document Distributions at Different Levels in Filtered ODP

Figures 1 and 2 show the category and document distributions, respectively, at different levels after filtering. ODP contains 81,349 categories and 1,358,649 documents. About 89% of the documents belong to the categories at the top 7 levels, and about 67% of them belong to the categories at the 4<sup>th</sup>, 5<sup>th</sup>, and 6<sup>th</sup> levels. Most of the documents (96.46%) belong to only one category. In this experiment, 20,000 documents that belong to a unique category are used for testing, and the rests are used for training. In selecting the test data, we considered the proportions of the numbers of documents and categories at each level to those in the entire hierarchy as in [15].

**Evaluation Criteria.** To show the efficacy of our methods, evaluations were conducted for different levels and categories. For level-based evaluation, we tested sensitivity of the methods to different abstraction levels as in [8,15] by looking at the micro-averaged F1 values at different levels. For example, when a document is classified to *Science/Biology/Ecology/Ecosystems*, we first check whether *Science* matches the first level of the correct category. If it passes the first test, it is tested for the second level by checking whether *Science/Biology* matches the correct category, and so on. This progressive tests stop at level 9 because the number of documents at a deeper level is very small. For category-based evaluation, we tested whether the methods depend on different topic categories. The categories at the top level were only used.

## 5.2 Evaluation

We compared our methods with the deep classification (DC) in [8] and the enhanced deep classification (EDC) [15] as summarized in Table 1. DC employs the category-based search, ancestor-assistant training data selection, and trigram language model classifier. EDC also utilizes the same search strategy but a different method for training data selection and a different classifier.

**Table 1.** The summary of algorithm settings

	Deep Classification (DC)	EnhancedDC (EDC)
Search	Category-based	Category-based
Training	Ancestor-assistant	Neighbor-assistant
Classifier	Trigram Language Model	Naïve Bayes Combination of Local and Global Information
Smoothing	No-smoothing	Add-One in Global Model

We implemented these methods and experimented with our dataset. Top-5 categories were considered as the candidates for classification. For the implementation of EDC, ancestor-assistant strategy was chosen to avoid the time complexity of neighbor-assistant strategy.

Our basic model is the same as EDC but different in two different aspects: the smoothing and mixture weight computation methods. Compared to the add-one smoothing applied only to the global model, we utilize interpolation for both local and global models. More importantly, the two optimization methods were used to determine the appropriate mixture weights dynamically in the proposed approach whereas the mixture weight between local and global models was set empirically.

**Overall Performance.** Figure 3 shows that our proposed deep classification with CBO (PDC-CBO) and RBO (PDC-RBO) outperforms both DC and EDC at all levels with only one exception (PDC-CBO slightly worse than EDC at level 1. PDC-RBO attains 0.8302 at the top-level, for example, while DC and EDC reach 0.6448 and 0.7868 respectively. Overall, PDC-CBO and PDC-RBO obtained 5% and 30% improvements respectively over EDC, and 77% and 119% over DC.



Since the global model provides information about the top-level category, it is most useful when two or more sub-categories (i.e. local models) contain similar information. With PDC-CBO, for example, *Kids\_and\_Teens/Arts/Dance* and *Arts/Performing\_Arts/Dance* may contain similar document contents at the *Dance* level. In this case, the top-level category information can provide a guide to the right path. One possible drawback of CBO is that the information content at the top level may be overly general. On the other hand, RBO uses relative information, i.e., relevance scores that effectively reflect the local information. It attempts to compute the degree to which the local information can be trusted and then fill in the rest with the global information.

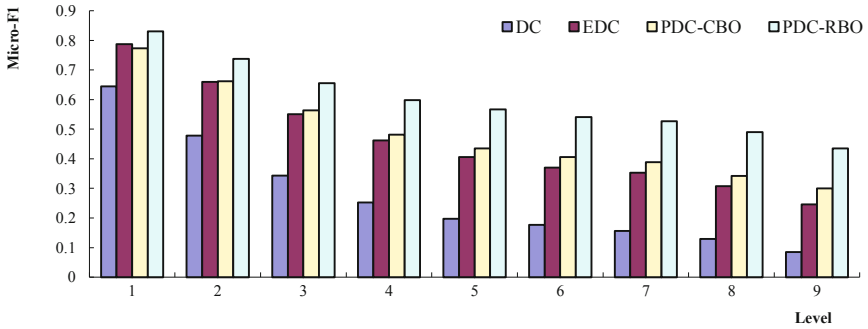


Fig. 3. Comparisons among Four Different Methods

**Role of Optimization.** We conducted experiments to investigate the role of optimization methods. The EDC method with no optimization and add-one smoothing method in the global model is regarded as the baseline to which each proposed optimization method is applied. As shown in Figure 4, the RBO method achieved 23.4% improvement over the baseline. On the other hand, the CBO method makes the performance somewhat decrease even though it gains some improvement at the 9th level. While the main thrust of the CBO method is to utilize the difference between the local and global models, the add-one smoothing applied to the global model makes the advantage disappear.

**Effects of Interpolation.** Figure 5 and 6 show the effects of the modified classifier with interpolation aimed at avoiding zero probabilities. The alpha is set to 0.7 since it achieved the best performance. As shown in Figure 5, CBO with the modified classifier obtains better performance than the EDC algorithm although CBO with add-one smoothing results in lower performance than EDC. The improvement becomes larger with deeper levels. Moreover, the performance improves 7% over the CBO method with the add-one smoothing method. As shown in Figure 6, RBO with the modified classifier also shows 5% improvement compared to RBO with add-one smoothing. The interpolation method is valuable in estimating the global and local models.

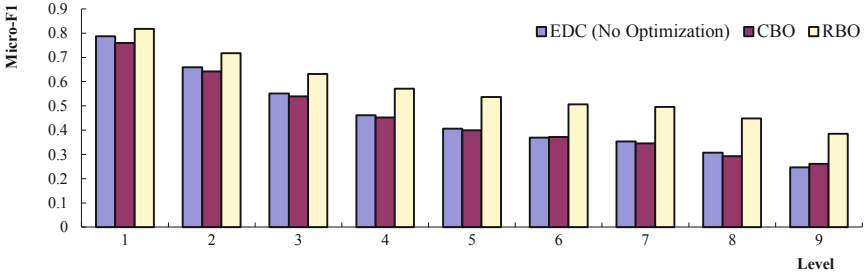


Fig. 4. Comparisons among Different Optimization Methods

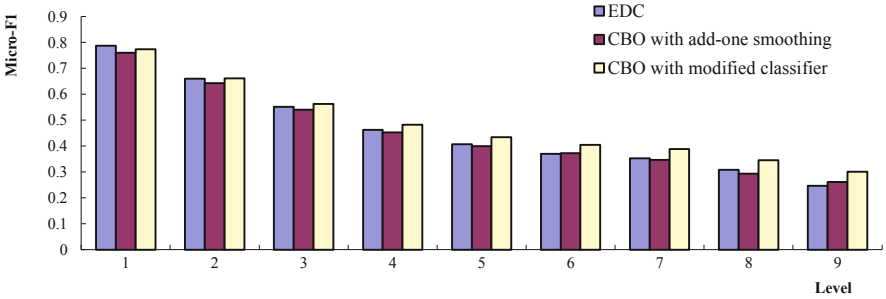


Fig. 5. Comparisons among EDC, CBO with Add-One Smoothing, and CBO with Interpolation

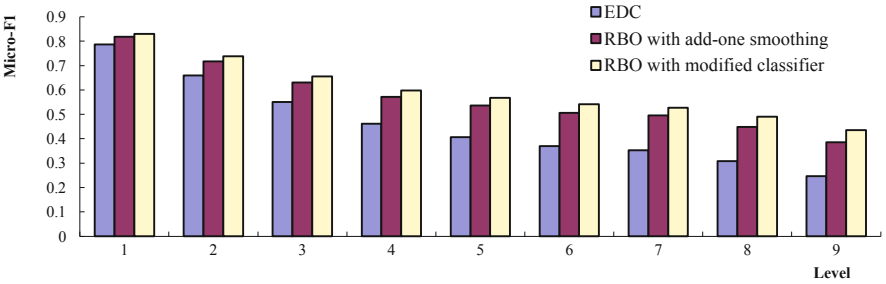


Fig. 6. Comparisons among EDC and RBO with Add-One Smoothing, and RBO with Interpolation

**Performance with Different Categories.** With the limited space and resources required to run the experiments for all the categories in the hierarchy, we show the performance for the top-level categories only. As shown in Table 2, PDC-CBO and PDC-RBO show remarkable improvements over DC and EDC across different categories. The macro-F1 scores of PDC-CBO and PDC-RBO show 25.05% and 77.83% improvements, respectively, compared to EDC. Our method with RBO shows the best performance on *Adult* category where micro-F1 is 0.4428. Other categories, except for *Kids\_and\_Teens* and *Reference*, also achieved over 0.35% improvements in micro-F1.

**Table 2.** Performance on Top-Level Categories

Category	DC		EDC		PDC-CBO		PDC-RBO	
	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1
<i>Adult</i>	0.0148	0.0183	0.1819	0.2012	<b>0.2224</b>	<b>0.2407</b>	<b>0.4010</b>	<b>0.4428</b>
<i>Arts</i>	0.0202	0.0194	0.1884	0.1895	<b>0.2307</b>	<b>0.2204</b>	<b>0.4315</b>	<b>0.4036</b>
<i>Society</i>	0.0108	0.0152	0.1652	0.1951	<b>0.2187</b>	<b>0.2291</b>	<b>0.3799</b>	<b>0.3963</b>
<i>Recreation</i>	0.0072	0.0097	0.1757	0.2007	<b>0.2453</b>	<b>0.2648</b>	<b>0.4009</b>	<b>0.4279</b>
<i>Home</i>	0.0077	0.0143	0.1711	0.1972	<b>0.2386</b>	<b>0.2350</b>	<b>0.4104</b>	<b>0.4222</b>
<i>Kids_and_Teens</i>	0.0121	0.0115	0.1015	0.1368	<b>0.1007</b>	<b>0.1265</b>	<b>0.1861</b>	<b>0.2170</b>
<i>Reference</i>	0.0031	0.0063	0.1123	0.1316	<b>0.1268</b>	<b>0.1638</b>	<b>0.2248</b>	<b>0.2602</b>
<i>Computers</i>	0.0073	0.0064	0.1555	0.1914	<b>0.2062</b>	<b>0.2201</b>	<b>0.3267</b>	<b>0.3582</b>
<i>Business</i>	0.0062	0.0069	0.1601	0.2003	<b>0.1968</b>	<b>0.2287</b>	<b>0.3556</b>	<b>0.4036</b>
<i>Games</i>	0.0085	0.0142	0.1339	0.1483	<b>0.1721</b>	<b>0.1878</b>	<b>0.2818</b>	<b>0.3045</b>
<i>Sports</i>	0.0137	0.0185	0.1580	0.1844	<b>0.2306</b>	<b>0.2411</b>	<b>0.3855</b>	<b>0.4177</b>
<i>Shopping</i>	0.0079	0.0076	0.1737	0.2110	<b>0.2091</b>	<b>0.2394</b>	<b>0.3678</b>	<b>0.4073</b>
<i>Health</i>	0.0064	0.0077	0.1613	0.1853	<b>0.2199</b>	<b>0.2416</b>	<b>0.3884</b>	<b>0.4221</b>
<i>Science</i>	0.0143	0.0170	0.1974	0.2229	<b>0.2438</b>	<b>0.2569</b>	<b>0.4015</b>	<b>0.4202</b>
<i>News</i>	0.0031	0.0074	0.1401	0.1849	<b>0.1096</b>	<b>0.1357</b>	<b>0.3420</b>	<b>0.3563</b>
Average	0.0096	0.0120	0.1584	0.1854	<b>0.1981</b>	<b>0.2154</b>	<b>0.3523</b>	<b>0.3773</b>
Improvement					<b>25.05%</b>	<b>16.22%</b>	<b>77.83%</b>	<b>75.14%</b>

## 6 Conclusion

In this paper, we propose a modified version of deep classification, which introduces two optimization methods for the purpose of properly combining local and global models. The experimental results show that our approach with the two optimization methods achieves 5% and 30% improvements in micro-F1, respectively, over the state-of-art methods in terms of level-oriented evaluations. For category-oriented evaluations, we also achieved 77.83% and 75.14% improvements in micro-F1 and macro-F1.

Even though our optimization methods perform quite well, there are some remaining issues that need to be investigated further. One is that RBO may be very sensitive to the number of candidate categories, and therefore there should be further work in investigating ways to make the method invariant. The other one is that what needs to be one when the difference between local and global models is significantly large. These issues will be studied with more extensive experiments.

## Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0028079), and Global RFP program funded by Microsoft Research.

## References

1. Broder, A.Z., Fontoura, M., Gabrilovich, E., Joshi, A., Josifovski, V., Zhang, T.: Robust classification of rare queries using web knowledge. In: 30th ACM SIGIR, pp. 231–238 (2007)
2. Broder, A., Fontoura, M., Josifovski, V., Riedel, L.: A semantic approach to contextual advertising. In: 30th ACM SIGIR, pp. 559–566 (2007)
3. Zhang, B., Li, H., Liu, Y., Ji, L., Xi, W., Fan, W., Chen, Z., Ma, W.Y.: Improving web search results using affinity graph. In: 28th ACM SIGIR, pp. 504–511 (2005)
4. Kosmopoulos, A., Gaussier, E., Paliouras, G., Aseervatham, S.: The ECIR 2010 large scale hierarchical classification workshop. SIGIR Forum. 44, 23–32 (2010)
5. Sun, A., Lim, E.P.: Hierarchical text classification and evaluation. In: IEEE ICDM, pp. 521–528 (2001)
6. Liu, T.Y., Yang, Y., Wan, H., Zeng, H.J., Chen, Z., Ma, W.Y.: Support vector machines classification with a very large-scale taxonomy. ACM SIGKDD Explorations Newsletter 7, 36–43 (2005)
7. Bennett, P.N., Nguyen, N.: Refined experts: improving classification in large taxonomies. In: 32nd ACM SIGIR, pp. 11–18 (2009)
8. Xue, G.R., Xing, D., Yang, Q., Yu, Y.: Deep classification in large-scale text hierarchies. In: 31st ACM SIGIR, pp. 619–626 (2008)
9. McCallum, A., Rosenfeld, R., Mitchell, T., Ng, A.Y.: Improving text classification by shrinkage in a hierarchy of classes. In: 15th ICML, pp. 359–367 (1998)
10. Cai, L., Hofmann, T.: Hierarchical document categorization with support vector machines. In: 13th ACM CIKM, pp. 78–87 (2004)
11. Labrou, Y., Finin, T.: Yahoo! as an ontology: using Yahoo! categories to describe documents. In: 8th ACM CIKM, pp. 180–187 (1999)
12. Sasaki, M., Kita, K.: Rule-based text categorization using hierarchical categories. In: IEEE International Conference on Systems, Man, and Cybernetics, vol. 3, pp. 2827–2830 (1998)
13. Wang, K., Zhou, S., He, Y.: Hierarchical classification of real life documents. In: 1st (SIAM) International Conference on Data Mining, pp. 1–16 (2001)
14. Yang, Y., Zhang, J., Kisiel, B.: A scalability analysis of classifiers in text categorization. In: 26th ACM SIGIR, pp. 96–103 (2003)
15. Oh, H.S., Choi, Y., Myaeng, S.H.: Combining global and local information for enhanced deep classification. In: 2010 ACM SAC, pp. 1760–1767 (2010)
16. Koller, D., Sahami, M.: Hierarchically classifying documents using very few words. In: International Conference on Machine Learning, pp. 170–178 (1997)
17. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: International Conference on Machine Learning, pp. 412–420 (1997)

# User-Related Tag Expansion for Web Document Clustering

Peng Li<sup>1,2</sup>, Bin Wang<sup>1</sup>, Wei Jin<sup>3</sup>, and Yachao Cui<sup>1,2</sup>

<sup>1</sup> Institute of Computing Technology, Chinese Academy of Sciences, China

<sup>2</sup> Graduate School of the Chinese Academy of Sciences, Beijing, China

<sup>3</sup> Department of Computer Science, North Dakota State University, USA  
{lipeng01,wangbin,cuiyachao}@ict.ac.cn, Wei.Jin@ndsu.edu

**Abstract.** As high quality descriptors of web page semantics, social annotations or tags have been used for web document clustering and achieved promising results. However, most web pages have few tags (less than 10). This sparsity seriously limits the usage of tags on clustering. In this work, we propose a user-related tag expansion method to overcome the problem, which incorporates additional useful tags into the original tag document by utilizing user tagging as background knowledge. Unfortunately, simply adding tags may cause topic drift, i.e., the dominant topic(s) of the original document may be changed. This problem is addressed in this research by designing a novel generative model called Folk-LDA, which jointly models original and expanded tags as independent observations. Experimental results show that (1) Our user-related tag expansion method can be effectively applied to over 90% tagged web documents; (2) Folk-LDA can alleviate the topic drift in expansion, especially for those topic-specific documents; (3) Compared to word-based clustering, our approach using only tags achieves a statistically significant increase of 39% on F1 score while reducing 76% terms involved in computation at best.

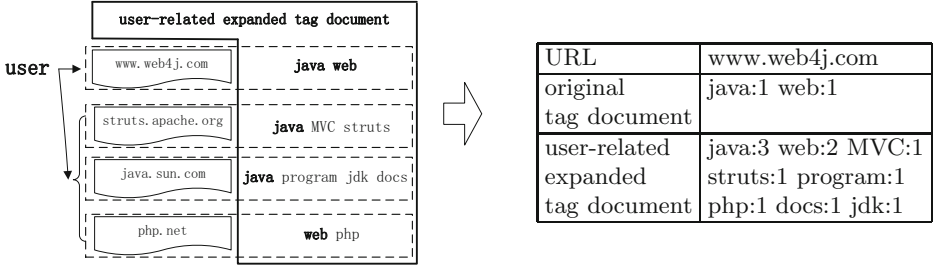
## 1 Introduction

Clustering is a central problem in text mining and information retrieval. Traditional methods are mostly based on the analysis of document words and develop similarity measures between documents [1, 2]. With the popularity of bookmarking websites like Delicious [1], Bibsonomy [2], web documents have been provided additional meaningful information such as tag representations given by users. In fact, Tags, as a new information source, have been proved to be accurate descriptors of document content and supplied high-level semantics [10], which could be beneficial to the clustering task. This paper mainly focuses on how to utilize these tags as background knowledge to improve web document clustering.

---

<sup>1</sup> <http://delicious.com/>

<sup>2</sup> <http://www.bibsonomy.org/>



**Fig. 1.** The construction of expanded virtual tag document. The original tag document for a web page `www.web4j.com` is its annotated tags (i.e., `java`, `web`). In the user’s bookmarks, there are three related entries containing the same tag(s) as those for `www.web4j.com`. The user-related expanded tag document for “`www.web4j.com`” web page is then constructed through aggregating all the tags in the four bookmarks.

Recently Ramage et al. [20] demonstrated combining tags and words together can improve clustering quality. Moreover, they found that using tags alone achieves better performance on topic-specific and densely labeled document collections. In their experiments, each document had 1,307 tag tokens on average. Motivated by their observation, we are attempting to answer the following question in this work: can we extend the advantage of using only tags on document clustering to more general and large-scale web pages?

It turns out the tag-based web document clustering is challenging because most web pages have few annotations. After the analysis of one of the largest collaborative tagging datasets [1], the statistics show that nearly 90% of URLs have no more than 3 users or 10 tags each. The widespread tag sparsity seriously affects the accurate estimation of document similarity and limits the use of tags in most cases.

To address the problem, we present a novel approach in this research by incorporating additional related tags. Specifically, we create a virtual expanded tag document for each tagging user of each web page. The user-related expanded tag document consists of tags in the user’s bookmarks which are required to contain tags of associated web pages. The detailed construction process is illustrated in Figure 1. Intuitively, the expanded tag document will augment tag vocabulary and increase tag co-occurrence frequency, which can alleviate sparsity on measuring document similarity and improve topic-specific tag distribution estimation. However, for a given web page, directly merging its associated tags with all expanded tags from its related web pages may introduce noise and even cause the problem of topic drift. To solve this, a new generative model called Folk-LDA is proposed through carefully modeling original and expanded tag terms. The model has achieved high efficiency without the involvement of document content, while maintaining high-quality clustering performance.

Our contributions can be summarized as follows:(1)To the best of our knowledge, we are the first to bring up and analyze the sparsity problem on tag-based document clustering. (2)We have exploited the feasibility of tag expansion and

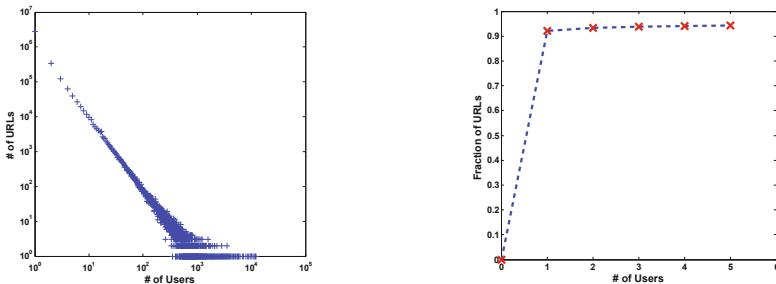
demonstrated its effect on addressing the sparsity problem.(3) A new topic modeling method, Folk-LDA, is proposed to utilize user-related expanded tag terms to improve web document clustering.(4)The observation that using only tags for web document clustering significantly outperforms word-based clustering methods shows the promise of employing human-centered knowledge for large scale document clustering and classification.

The rest of the paper is organized as follows. Section 2 analyzes the tag and user distributions on web pages and points out the sparsity problem that needs to be addressed. Section 3 presents our proposed user-related tag expansion approach and its variations. Section 4 discusses the experimental results and comparison with existing clustering methods. Section 5 introduces related work, and is followed by conclusions.

## 2 Tagging Data Analysis

In our experiments we used the dataset [1] from DAI-Labor, which is constructed through dumping a collaborative tagging system: Delicious - a repository of website bookmarks. The DAI dataset contains about 142 million bookmarks, 950,000 different users and 45 million unique URLs. The statistical analysis was based on randomly extracting about 10% URLs from the dataset.

Figure 2(left) shows the power law shaped distribution of URL frequency labeled by users from our analysis dataset. We observed that most URLs on the Web have few assigned tags as well as tagging users, while only a small number of pages have a lot. Specifically, the statistics illustrate that 90% of URLs have no more than 10 tags; 79.5% of URLs are bookmarked by only one user; 95.4% of URLs have users no more than 5; Only 2.6% URLs have more than 10 users. Similar findings were also mentioned in works [14][22].



**Fig. 2.** *left*:The distribution of URL frequency labeled by users. *right*:The fraction of URLs with augmented tag vocabulary through user-related tag expansion vs. Number of users.

We also explored the proportion of web pages with an augmented tag vocabulary after expansion. This was achieved by drawing 5 URL samples from the above 10% URL subset, each containing 10,000 URLs. The samples were classified according to tagging users ranging from 1 to 5. The lightly labeled

URLs were primarily examined, mainly because most web pages have few users as shown previously. From Figure 2(right), we can see that the more tagging users a URL has, the more probable additional new related terms appear in its constructed expanded tag documents. Even for the URLs with just one user, over 90% have new expanded tag terms. This shows that after tag expansion, an augmented tag vocabulary will apply in most cases. The observation that users tend to label URLs with previously used tag terms is also consistent with the analysis of user’s purpose of organizing or browsing web pages reported in [10].

### 3 The Proposed Techniques

#### 3.1 Tag Expansion for Document Clustering

In this section, we introduce our proposed tag expansion techniques on document clustering. Specifically, our approach is composed of two major sequential steps: the construction of expanded tag document, and followed by the use of expanded tag document for the clustering task.

Formally, we use  $D$  to represent the document collection to be clustered. For each document  $d \in D$ , suppose  $U^d$  is the set of tagging users of  $d$ , by aggregating the tags of the  $|U^d|$  users on  $d$ , we generate the original document representation  $T^d$  in tag space. As shown in section 2,  $|U^d|$  does not exceed 5 for most URLs on the Web.

In the first stage, a virtual document  $T^{u,d}$  for each user  $u$  of each document  $d$  is constructed.  $T^{u,d}$  is made up of the tags in  $u$ ’s bookmarks which match the tag(s) of document  $d$  labeled by user  $u$ . Note that  $T^{u,d}$  may incorporate hypernyms, synonyms or other descriptive tags of related topics from the user’s perspective. In this process, we consider the tag expansion in the scope of users who have labeled the original document. Alternative expansion methods, such as expansion based on tag co-occurrence only without user constraints, have also been examined. However, we believe user-related tag expansion is more feasible to control the vocabulary size and guarantees the semantic consistency of tag terms included after expansion.

In the second stage, to use the generated  $T^{u,d}$ , an intuitive way is to merge all the  $T^{u,d}(u \in U^d)$  and  $T^d$  as the new representation( $T^d + \sum_{u \in U^d} T^{u,d}$ ) for each document  $d$ . We call it a merged document for later use. It can be expected that the topic of original document will be strengthened by this aggregation process. However, the problem may occur in case expanded tags from related documents introduce different topic tags in the merged document. To overcome this, we consider using a generative topic model such as Latent Dirichlet Allocation(LDA) to guide clustering. LDA models document as a mixture of hidden topic variables, and we borrow this underlying idea to analyze the topic components of the merged tag document. Each topic is treated as a cluster and each document will be assigned to a cluster which represents its dominant topic. We call this method LDA-M. By considering the dominant topic only, the influence from other noisy topic tags can be reduced.



Note that the performance of topic models such as LDA is usually dependent on the number of topics specified. To estimate the optimal number of topics, several parameter selection methods such as perplexity measure [3, 8] can be used. In this work, we set the number of topics equal to that provided by the clustering standard to simplify the parameter space. The details of data sets and evaluation process are described in section 4.

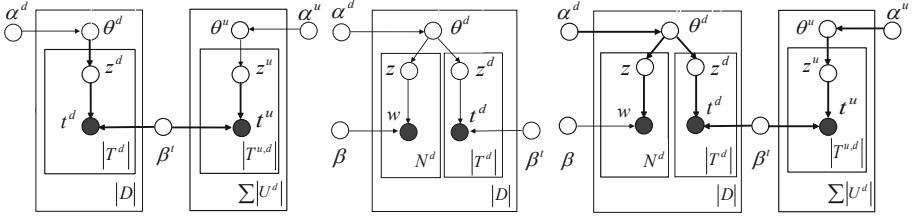
### 3.2 Folk-LDA

In the previous section, we mentioned that the topic distribution of  $T^d + \sum_{u \in U^d} T^{u,d}$  may be different from  $T^d$ . In fact, the situation gets worse when the dominant topic in  $T^d$  is not the same as in  $T^d + \sum_{u \in U^d} T^{u,d}$ , where LDA-M cannot work properly. Especially for clustering specific documents about the same theme but classified into multiple classes, e.g., those returned by a search query, the expanded tag document for one cluster is prone to contain excessive representative tag terms of other clusters. For example, for the URL "http://www.mojava.org/" under the category *PHP*, we find a user labels it with tags "mvc" and "php", however, the constructed virtual tag document has incorporated "java", "framework", "spring" through "mvc". This is because the user has stored many URL entries on "mvc" and its associated programming languages, A large number of representative tag terms of *Java* topic occur in the merged document so that *Java*, instead of *PHP*, becomes the dominate topic of the web page.

To overcome this, we extend LDA to jointly model tags of original and expanded tag documents as distinctive sets of observations but sharing the same tag distribution over topics. Specifically, we name this model Folk-LDA and outline the generative process in Figure 3 and Table 1. Superscripts d and u represent the original tag document related and expanded tag document related parameters respectively. The number of topics on the original document side equals to that on the expanded document side.

Folk-LDA has demonstrated the following advantages: (1) The separation of original and expanded tag documents reduces the negative impact of expansion on influencing the topic of original document to the minimum. (2) The topic-specific tag distributions ( $\beta^t$ ) are estimated based on globally necessary information, which guides the estimation of document topic accurately. (3) Different prior distributions are assigned for original tag document and expanded tag document considering their different generative processes, which fits the data better compared to that using the same prior structure for both documents.

**Inference and Estimation.** The main variables of interest are  $\theta^d, \theta^u, \beta^t$ , which can be learned by maximizing the likelihood for the observation. Gibbs sampling [8] and variational inference [3] are two representative methods for solving LDA like models. Gibbs sampling has theoretical assurance that the final results will converge to the true distribution while variational inference executes faster but with approximations. Here, we develop a variational inference based method considering the potential applications of our algorithm on large scale document collections.



**Fig. 3.** Graphical representations of Folk-LDA (left), MM-LDA (middle) and Folk-MM-LDA (right)

**Table 1.** The Generative process for Folk-LDA:  $K$  is the number of underlying latent topics;  $T$  represents the tag vocabulary after expansion

For each document  $d \in D$   
 Generate  $\theta^d \sim \text{Dirichlet}(\cdot | \alpha^d)$   
 For each tag index  $n \in 1, \dots, |T^d|$ :  
 Generate  $z_{dn}^d \in 1, \dots, K \sim \text{Multinomial}(\cdot | \theta^d)$   
 Generate tag  $t_{dn}^d \in 1, \dots, |T| \sim \text{Multinomial}(\cdot | \beta_{z_{dn}^d}^t)$   
 For each expanded document  $T^{u,d}, u \in U^d$   
 Generate  $\theta_{du}^u \sim \text{Dirichlet}(\cdot | \alpha^u)$   
 For each tag index  $n \in 1, \dots, |T^{u,d}|$ :  
 Generate  $z_{dun}^u \in 1, \dots, K \sim \text{Multinomial}(\cdot | \theta_{du}^u)$   
 Generate tag  $t_{dun}^u \in 1, \dots, |T| \sim \text{Multinomial}(\cdot | \beta_{z_{dun}^u}^t)$

Specifically, we use a fully factorized distribution to approximate the posterior distribution of the latent variables as follows:

$$q(\boldsymbol{\theta}^d, \boldsymbol{\theta}^u, \mathbf{z}^d, \mathbf{z}^u | \boldsymbol{\gamma}^d, \boldsymbol{\gamma}^u, \boldsymbol{\phi}^d, \boldsymbol{\phi}^u) = \prod_{d=1}^{|D|} \left( q(\boldsymbol{\theta}^d | \boldsymbol{\gamma}^d) \prod_{n=1}^{|T^d|} q(z_{dn}^d | \phi_{dn}^d) \prod_{u=1}^{|U^d|} \left( q(\boldsymbol{\theta}_{du}^u | \boldsymbol{\gamma}_{du}^u) \prod_{n=1}^{|T^{u,d}|} q(z_{dun}^u | \phi_{dun}^u) \right) \right)$$

where  $\boldsymbol{\gamma}^d, \boldsymbol{\gamma}^u$  are Dirichlet parameters and  $\boldsymbol{\phi}_{dn}^d, \boldsymbol{\phi}_{dun}^u$  are multinomial parameters. Using the variational EM algorithm, we get the final parameter updates:

**E-step**

$$\begin{aligned} \phi_{dni}^d &\propto \beta_{ij}^t \exp \left( \Psi(\gamma_{di}^d) - \Psi \left( \sum_{j=1}^k \gamma_{dj}^d \right) \right) & \gamma_{di}^d &= \alpha_i^d + \sum_{n=1}^{|T^d|} \phi_{dni}^d \\ \phi_{dun}^u &\propto \beta_{ij}^t \exp \left( \Psi(\gamma_{dun}^u) - \Psi \left( \sum_{j=1}^k \gamma_{dun}^u \right) \right) & \gamma_{dun}^u &= \alpha_i^u + \sum_{n=1}^{|T^{u,d}|} \phi_{dun}^u \end{aligned}$$

**M-step**

$$\beta_{ij}^t \propto \sum_{d=1}^D \left( \sum_{n=1}^{|T^d|} \phi_{dni}^d \delta_j(t_{dn}^d) + \sum_{u=1}^{|U^d|} \sum_{n=1}^{|T^{u,d}|} \phi_{dun}^u \delta_j(t_{dun}^u) \right)$$

**Clustering.** Once the parameters have been estimated, document  $d$  has topic components proportional to  $\boldsymbol{\gamma}_d^d$ . We take each topic as a cluster and each document is assigned to the topic with the maximum value among  $\boldsymbol{\gamma}_d^d$ . This process can also be seen as mapping the soft clustering results to hard assignments.

### 3.3 Folk-MM-LDA

In previous discussions, we mainly focus on tag-based clustering, i.e., only the tagging data is used. In fact, Folk-LDA can be naturally combined with other types of document information such as words to better estimate the underlying topic embodied in a document. This idea has been incorporated into the model named Folk-MM-LDA and illustrated in Figure 4. Specifically, document words and tags are modeled by sharing the common document topic proportion  $\theta^d$  while have their distinct topic-specific distributions  $(\beta, \beta^t)$ . Similar as Folk-LDA,  $\beta^t$  in Folk-MM-LDA is estimated from a globally related context through expansion, which demonstrates its advantage over MM-LDA [20] in the general case where few tags are available. Folk-MM-LDA can be seen as an extension of MM-LDA by further considering the expanded tag document information. The variational inference methods used by Folk-MM-LDA and MM-LDA are listed in Table 2.

**Table 2.** Variational methods for Folk-MM-LDA and MM-LDA.  $\phi_{dni}$  is a variational parameter corresponding to the posterior probability that the  $n^{\text{th}}$  word in document  $d$  comes from topic  $i$ .  $\beta_{ij}$  is a parameter for generating the word term  $j$  from topic  $i$ .

	Folk-MM-LDA	MM-LDA
E-step	$\phi_{dni} \propto \beta_{ij} \exp\left(\Psi(\gamma_{di}^d) - \Psi(\sum_{j=1}^k \gamma_{dj}^d)\right)$ $\phi_{dni}^d \propto \beta_{ij}^t \exp\left(\Psi(\gamma_{di}^d) - \Psi(\sum_{j=1}^k \gamma_{dj}^d)\right)$ $\phi_{duni}^u \propto \beta_{ij}^t \exp\left(\Psi(\gamma_{dui}^u) - \Psi(\sum_{j=1}^k \gamma_{duj}^u)\right)$ $\gamma_{di}^d = \alpha_i^d + \sum_{n=1}^{ T^d } \phi_{dni}^d + \sum_{n=1}^{N^d} \phi_{dni}$ $\gamma_{dui}^u = \alpha_i^u + \sum_{n=1}^{ T^{u,d} } \phi_{duni}^u$	$\phi_{dni} \propto \beta_{ij} \exp\left(\Psi(\gamma_{di}^d) - \Psi(\sum_{j=1}^k \gamma_{dj}^d)\right)$ $\phi_{dni}^d \propto \beta_{ij}^t \exp\left(\Psi(\gamma_{di}^d) - \Psi(\sum_{j=1}^k \gamma_{dj}^d)\right)$ $\gamma_{di}^d = \alpha_i^d + \sum_{n=1}^{ T^d } \phi_{dni}^d + \sum_{n=1}^{N^d} \phi_{dni}$
M-step	$\beta_{ij}^t \propto \sum_{d=1}^D \left( \sum_{n=1}^{ T^d } \phi_{dni}^d \delta_j(t_{dn}^d) + \sum_{u=1}^{ U^d } \sum_{n=1}^{ T^{u,d} } \phi_{duni}^d \delta_j(t_{dun}^u) \right)$ $\beta_{ij} \propto \sum_{d=1}^D \sum_{n=1}^{N^d} \phi_{dni} \delta_j(w_{dn})$	$\beta_{ij}^t \propto \sum_{d=1}^D \sum_{n=1}^{ T^d } \phi_{dni}^d \delta_j(t_{dn}^d)$ $\beta_{ij} \propto \sum_{d=1}^D \sum_{n=1}^{N^d} \phi_{dni} \delta_j(w_{dn})$

## 4 Experiments

### 4.1 Evaluation Method and Dataset

On measuring the clustering quality, the output needs to be judged by humans explicitly. However, this involves considerable and expensive effort. For web document clustering, a broadly accepted evaluation method is to use Open Directory Project<sup>3</sup> as clustering standard [20, 17]. Specifically, when using ODP for evaluation, we pick out one node from the ODP hierarchy and then take each child and its descendants as a standard cluster. The generated clusters using different clustering methods are then compared to this ODP-derived cluster standard.

In the experiments, those lightly labeled general web pages were mainly focused on. Thus, a major task in evaluation was constructing an evaluation data set, which was composed of the following sequential steps:

<sup>3</sup> <http://www.dmoz.org/>

1. We randomly picked out lightly labeled URLs from the DAI dataset as seeds, each of which has no more than 5 tagging users. The number 5 is chosen based on previous analysis in Section 2.

2. The seed URLs were submitted to google directory search<sup>4</sup> to acquire their ODP category names.

3. Those seed URLs with no search results returned were discarded. For others, we stored at most top 2 search results and extracted their URLs and ODP category names.

4. The stored URLs in step 3 were further cleaned if they meet any of the following conditions: (1)They have no bookmarks in DAI dataset. (2)They are under the ODP category “Regional”. (3)Their contents cannot be downloaded. (4)They are not written in English.

The above process resulted in an augmented set of URLs because google may return related URLs beyond the original query URL, which were captured in step 3. The documents under ODP category “Regional” were discarded because they are grouped according to geographical information. Our final test collection includes 13,188 documents from 15 categories. However, most URLs in the evaluation set have involved more users than those for seed URLs. This may be caused by the fact that ODP tends to include popular pages. To simulate sparsity in real scenarios, we randomly picked 5 users for those heavily labeled URLs and took the annotations given by a maximum of 5 users as the complete tag information each URL has.

We choose F1 evaluation measure [18] to compare our results with previous work. In our experiments, each evaluation result is the mean across 10 runs. When we mention that the improvement is significant, it means  $p < 0.01$  in a one-tailed two sample t-test.

## 4.2 The Tag Expansion Effect on Clustering

We conducted experiments to answer the following two questions: (1)Is user-related tag expansion effective in alleviating the sparsity problem? (2)Is the LDA-based approach better than traditional k-means[18] on clustering the merged documents? Specifically, four methods are compared: k-means on the original document  $T^d$  (KMeans-T); k-means on the merged document (KMeans-M); LDA on the original document (LDA-T); LDA on the merged document (LDA-M).

For k-means, tags are weighted by their term frequencies and the number of clusters is set to that of ODP hierarchies as is in LDA. For hyper parameters in LDA derived models, we use default values, that is, set  $\alpha^d = \alpha^u = 1$  and smooth  $\beta^t, \beta$  with uniform dirichlet prior whose parameter equals to 1. The new tag terms incorporated through expansion are required to appear in at least 2 tag documents including the original tag document. Tags of original documents are all used. The tag terms are tokenized and stemmed by Lemur toolkit<sup>5</sup>. The maximum number of URL tagging users ranges from 1 to 5. These parameter settings are also applied in later experiments.

<sup>4</sup> <http://www.google.com/dirhp>

<sup>5</sup> <http://www.lemurproject.org/>

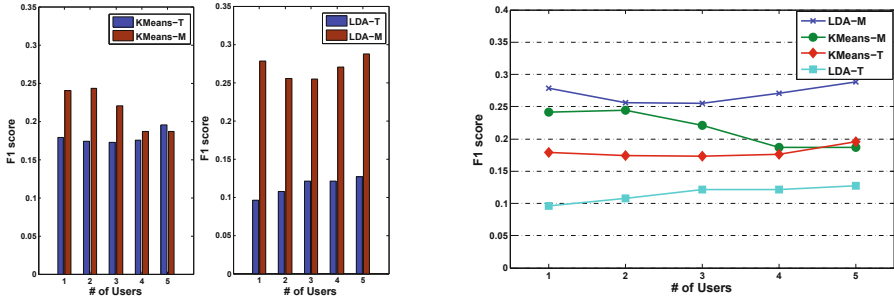


Fig. 4. Comparison of F1 scores on clustering the documents under the “Top” node

Figure 4 gives the comparison of clustering results under the “Top” node. Significant improvements are found by comparing methods with expansion (KMeans-M, LDA-M) to corresponding methods without expansion (KMeans-T, LDA-T). Specifically, KMeans-M outperforms KMeans-T by over 40% at best while LDA-M outperforms LDA-T by at least 100 percent. From the figure, we also observe the benefit of incorporating expanded tag terms in LDA-M on clustering merged multi-topic documents. The figure shows LDA-M outperforms KMeans-M by 54% at most and by 27% on average. Though KMeans-M also improves the clustering quality, the results tend to go down as more users are involved, which shows k-means is sensitive to the introduction of other topic tags.

### 4.3 Usage Comparison of Expanded Tag Documents

These experiments are designed to evaluate if independent modeling of the original document and expanded document outperforms the heuristic merging method on clustering quality. Specifically, we compare Folk-LDA and LDA-M for different scenarios: clustering the top-level ODP subtrees and clustering low-level ODP subtrees. For the former case, we cluster the documents under the “Top” node (*Top*) as in section 4.2. For the latter, we select two representatives with their small categories filtered. One is “Top/Computers/Programming/Languages” (*Languages*) which includes 690 documents belonging to 9 categories: Delphi, Fortran, Java, JavaScript, Lisp, Perl, PHP, Python and Ruby. The other is “Top/Arts” (*Arts*) which includes 655 documents from 7 categories: Architecture, Comics, Education, Literature, Music, Photography, Television.

The comparison of clustering performance is shown in Table 3. As we expect, Folk-LDA achieves better results than LDA-M on more specific document collections (*Languages*, *Arts*) where users often assign the same tag to the documents from different categories due to their semantic commonness, and therefore the topic drift problem is often observed in the merged document after tag expansion. The experimental results demonstrate effectiveness of the proposed Folk-LDA in overcoming the problem. However, LDA-M generally outperforms Folk-LDA on clustering documents under the “Top” node. A possible reason is that different

**Table 3.** Results for clustering the documents under the nodes “Top/Computers/Programming/Languages”, “Top/Arts” and “Top”. ▲ indicates Folk-LDA achieves a significant improvement over the LDA-M method.

no. of users	<i>Languages</i>		<i>Arts</i>		<i>Top</i>	
	LDA-M	Folk-LDA	LDA-M	Folk-LDA	LDA-M	Folk-LDA
1	0.514	0.522	0.444	0.488	0.279	0.231
2	0.521	0.601▲	0.561	0.568	0.256	0.244
3	0.628	0.690▲	0.579	0.636▲	0.255	0.255
4	0.601	0.697▲	0.588	0.632▲	0.271	0.234
5	0.641	0.652	0.607	0.657▲	0.288	0.248

subcategories of “Top” node have little in common and people seldom tag them with common tag terms. This dramatically reduces the factors that lead to topic drift in merged tag document.

#### 4.4 Comparison of Clustering Based on Tags and/or Words

Since user-related tag expansion greatly improves the performance of tag based web document clustering, it is natural to consider the following questions:(1)Are tags a better resource than words on document clustering? (2)How efficient tag-based document clustering is?

We further conducted the experiments and compared clustering performance based on different information sources: (1) using only words (KMeans-W, LDA-W) (2) using only tags (LDA-M) (3) combining words and tags (MM-LDA, Folk-MM-LDA). As for clustering based on words, for efficiency consideration, we keep the most frequent 100,000 terms in the vocabulary whose total size is 686,310. Other feature selection methods [15,23] can also be applied here.

Table 4 shows that the clustering methods considering tagging data(LDA-M, MM-LDA, Folk-MM-LDA) all significantly outperform the word-based clustering methods(KMeans-W,LDA-W), where MM-LDA achieves better performance than LDA-W, which is consistent with the conclusion in [20]. These experimental results also indicate that tagging data could be a better resource than words in representing document contents. Specifically, Folk-MM-LDA performs significantly better than MM-LDA when the number of users involved is quite small(no. of users=1), which owes to the better estimation of topic-specific tag distribution from the abundant tag information after expansion. But the improvement is not significant when the number of tagging users increases.

**Table 4.** Comparison of clustering the documents under the “Top” node based on different information sources: Words (KMeans-W,LDA-W), Tags (LDA-M), Tags+Words (MM-LDA, Folk-MM-LDA)

no. of users	KMeans-W	LDA-W	LDA-M	MM-LDA	Folk-MM-LDA
1	0.193	0.200	0.279	0.255	0.291
2	0.193	0.200	0.256	0.215	0.217
3	0.193	0.200	0.255	0.274	0.294
4	0.193	0.200	0.271	0.286	0.296
5	0.193	0.200	0.288	0.226	0.213

**Table 5.** Average document lengths based on different information sources and the number of one-tag documents before expansion under the ‘‘Top’’ node

no. of users	average document length			number of documents	
	$ T^d $	$ T^d + \sum_{u \in U^d} T^{u,d} $	$N^d$	$ T^d  = 1$	$ T^d  =  t^d  = 1$
1	2.81	383.3	1346.9	3092	385
2	5.39	779.0	1346.9	591	53
3	7.81	1126.5	1346.9	72	43
4	10.15	1555.4	1346.9	327	40
5	12.42	1917.5	1346.9	315	37

We also compared term scales involved in computation. As shown in Table 5, the number of tags used for clustering is significantly less than that of words. For no. of users=1, the average document length in tag space is 383.3, which accounts for only 28% of that in word space(1346.9). Besides, the tag sparsity problem before expansion is also obvious from our statistical data. For no. of users=1, 3092 documents, which account for 23.4% (3092/13188), have only one tag ( $|T^d| = 1$ ); 385 documents have a tag which only appears once in the whole document collection ( $|t^d| = 1$ ).

## 5 Related Work

Clustering as a key tool has been broadly used in organizing documents [19], presenting search results [24] and improving search models [16]. Traditional methods are based on the ‘‘bag of words’’ representation of document content, and recently several works have been reported to enrich the semantic information by using external knowledge such as WordNet [4], ODP [5] and Wikipedia [6, 13].

The most related work to ours is [20] but a different aspect was addressed. They proved usefulness of tags as a complementary information source while we go one step further to explore how to maximize tag usage to tackle the sparsity problem and improve clustering effects and efficiency. [17] proposed a joint cluster model which considered resources, users and tags together, however, it still suffers the sparsity problem. Zhou [25] explored tagging data for improving information retrieval, where the proposed user-content-annotation(UCA) model assigns the same prior topic distribution on both word-based documents and user-based documents whereas in our approach, an explicit distinction is made between these documents to better formulate the underlying model.

Tag clustering has also been explored as a main tool in the following applications: finding relationship between tags [2], generating a taxonomy from folksonomy [9], discovering social interests [14] and personalized navigation recommendation [7, 21], etc.

## 6 Conclusion and Future Work

In this paper, we first bring up the tag sparsity problem that widely exists in web scale document clustering, and then propose a new model to alleviate this

problem by developing a user-related tag expansion method to enhance clustering performance. Specifically, a novel generative model called Folk-LDA and its several variations have been designed and evaluated, which effectively tackle the problems of tag sparsity and topic drift. Experimental results based on a human-edited Web directory lead to the conclusion that LDA based on a direct merge method is suitable for clustering documents whose standard clusters have nearly no semantic overlaps while Folk-LDA is more capable of shielding augmented noisy topics after tag expansion, which is appropriate for more focused document clustering.

Future directions include the examination of the selection strategies to better choose expanded tag terms and consideration of using and evaluating various weighting schemes for expanded terms, which we believe will further improve Web scale document clustering performance and efficiency. We are also interested in the actual performance of applying expansion methods on other data such as videos, images, etc., which may also suffer the sparsity problem.

**Acknowledgments.** This work is supported by the Major State Basic Research Project of China under Grant No. 2007CB311103, the National High Technology Research and Development Program of China under Grant No. 2006AA010105, and the National Science Foundation of China under Grant No. 60776797, 61070111 and 60873166.

## References

1. [http://www.dai-labor.de/en/competence\\_centers/irml/datasets/](http://www.dai-labor.de/en/competence_centers/irml/datasets/)
2. Begelman, G.: Automated tag clustering: Improving search and exploration in the tag space. In: Proc. of the Collaborative Web Tagging Workshop at WWW 2006 (2006)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* 3 (2003)
4. Dave, K., Lawrence, S., Pennock, D.M.: Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: WWW 2003 (2003)
5. Gabrilovich, E., Markovitch, S.: Feature generation for text categorization using world knowledge. In: IJCAI 2005, pp. 1048–1053 (2005)
6. Gabrilovich, E., Markovitch, S.: Overcoming the brittleness bottleneck using wikipedia: enhancing text categorization with encyclopedic knowledge. In: AAAI 2006, pp. 1301–1306 (2006)
7. Gemmell, J., Shepitsen, A., Mobasher, B., Burke, R.: Personalizing navigation in folksonomies using hierarchical tag clustering. In: Data Warehousing and Knowledge Discovery, pp. 196–205 (2008)
8. Griffiths, T.L., Steyvers, M.: Finding scientific topics. *Proceedings of the National Academy of Sciences* 101(suppl. 1), 5228–5235 (2004)
9. Heymann, P., Garcia-Molina, H.: Collaborative creation of communal hierarchical taxonomies in social tagging systems. Tech. Rep. 2006-10, Computer Science Department (2006)
10. Heymann, P., Koutrika, G., Garcia-Molina, H.: Can social bookmarking improve web search? In: WSDM 2008, pp. 195–206 (2008)



11. Hotho, A., Staab, S., Stumme, G.: Wordnet improves text document clustering. In: Proc. of the SIGIR 2003 Semantic Web Workshop, pp. 541–544 (2003)
12. Hu, J., Fang, L., Cao, Y., Zeng, H.J., Li, H., Yang, Q., Chen, Z.: Enhancing text clustering by leveraging wikipedia semantics. In: SIGIR 2008 (2008)
13. Hu, X., Zhang, X., Lu, C., Park, E.K., Zhou, X.: Exploiting wikipedia as external knowledge for document clustering. In: KDD 2009, pp. 389–396 (2009)
14. Li, X., Guo, L., Zhao, Y.E.: Tag-based social interest discovery. In: WWW 2008, pp. 675–684 (2008)
15. Liu, T., Liu, S., Chen, Z.: An evaluation on feature selection for text clustering. In: ICML, pp. 488–495 (2003)
16. Liu, X., Croft, W.B.: Cluster-based retrieval using language models. In: SIGIR 2004, pp. 186–193 (2004)
17. Lu, C., Chen, X., Park, E.K.: Exploit the tripartite network of social tagging for web clustering. In: CIKM 2009, pp. 1545–1548 (2009)
18. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
19. McKeown, K.R., Barzilay, R., Evans, D., Hatzivassiloglou, V., Klavans, J.L., Nenkova, A., Sable, C., Schiffman, B., Sigelman, S.: Tracking and summarizing news on a daily basis with columbia’s newsblaster. In: HLT 2002, pp. 280–285 (2002)
20. Ramage, D., Heymann, P., Manning, C.D., Garcia-Molina, H.: Clustering the tagged web. In: WSDM 2009, pp. 54–63 (2009)
21. Shepitsen, A., Gemmell, J., Mobasher, B., Burke, R.D.: Personalized recommendation in social tagging systems using hierarchical clustering. In: RecSys, pp. 259–266 (2008)
22. Wetzker, R., Zimmermann, C., Bauckhage, C.: Analyzing social bookmarking systems: A delicio.us cookbook. In: Mining Social Data (MSoDa) Workshop Proceedings, ECAI 2008, pp. 26–30 (2008)
23. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: ICML 1997, pp. 412–420 (1997)
24. Zeng, H.J., He, Q.C., Chen, Z., Ma, W.Y., Ma, J.: Learning to cluster web search results. In: SIGIR 2004, pp. 210–217 (2004)
25. Zhou, D., Bian, J., Zheng, S., Zha, H., Giles, C.L.: Exploring social annotations for information retrieval. In: WWW 2008, pp. 715–724 (2008)

# A Comparative Experimental Assessment of a Threshold Selection Algorithm in Hierarchical Text Categorization

Andrea Addis, Giuliano Armano, and Eloisa Vargiu

University of Cagliari,  
Department of Electrical and Electronic Engineering  
{addis,armano,vargiu}@diee.unica.it  
<http://iasc.diee.unica.it>

**Abstract.** Most of the research on text categorization has focused on mapping text documents to a set of categories among which structural relationships hold, i.e., on hierarchical text categorization. For solutions of a hierarchical problem that make use of an ensemble of classifiers, the behavior of each classifier typically depends on an acceptance threshold, which turns a degree of membership into a dichotomous decision. In principle, the problem of finding the best acceptance thresholds for a set of classifiers related with taxonomic relationships is a hard problem. Hence, devising effective ways for finding suboptimal solutions to this problem may have great importance. In this paper, we assess a greedy threshold selection algorithm aimed at finding a suboptimal combination of thresholds in a hierarchical text categorization setting. Comparative experiments, performed on Reuters, report the performance of the proposed threshold selection algorithm against a relaxed brute-force algorithm and against two state-of-the-art algorithms. Results highlight the effectiveness of the approach.

## 1 Introduction

The new information era has widely changed our lives, thanks to a great deal of new possibilities to create content (i.e., knowledge) and share it all over the world throughout new and widespread communication systems. Human beings have always known the importance of organizing entities or notions in hierarchies, according to the “divide et impera” paradigm. In the last few decades, with the advent of modern information systems, this concept has been widely used to partition items and concepts into smaller parts, each being effectively and efficiently managed. The benefits of this approach are particularly evident in the Web 2.0, where the main and most crucial knowledge bases exploit its peculiarities to organize large collections of web pages<sup>1</sup>, articles<sup>2</sup>, or emails<sup>3</sup> in

<sup>1</sup> E.g., Google Directory (<http://www.google.com/dirhp>) and the DMOZ project (<http://www.dmoz.org>)

<sup>2</sup> E.g., Wikipedia (<http://www.wikipedia.org>) and Reuters (<http://www.reuters.com/>)

<sup>3</sup> E.g., Thunderbird 3 (<http://www.mozillamessaging.com/thunderbird/>)

hierarchies of topics. This organization allows to focus on specific levels of detail, ignoring specialization at lower levels and generalization at upper levels. In this scenario, the main goal of automated Text Categorization (TC) is to deal with reference taxonomies in an effective and efficient way –the corresponding research subfield being Hierarchical Text Categorization (HTC).

How to find the best acceptance thresholds for a set of classifiers correlated to taxonomic relationships is a hard problem. In this paper, we perform a comparative experimental assessment of a greedy threshold selection algorithm aimed at finding a suboptimal combination of thresholds in the context of Progressive Filtering (PF), the HTC technique discussed in [2]. Experimental results, performed on the Reuters data collections, show that the proposed approach is able to find suboptimal solutions while maintaining a quadratic complexity, which allows to adopt the algorithm also for large taxonomies [4].

The rest of the paper is organized as follows: Section 2 introduces the main thresholding strategies proposed in the literature. In Section 3, we briefly recall the PF approach. Section 4 describes TSA, putting into evidence the theoretical background that allowed to build the algorithm, together with its computational benefits. Experiments and results are illustrated in Section 5. Conclusions and future work, discussed in Section 6, end the paper.

## 2 Thresholding Strategies

Thresholding strategies in TC are an under-explored area of research. Indeed, they were often briefly mentioned as an unimportant post-processing step, the underlying assumptions being that thresholding strategies do not make much difference in the performance of a classifier and that finding the thresholding strategy for any given classifier is trivial. Neither these assumptions are true [13].

In TC, the three commonly used thresholding strategies are RCut, PCut, and SCut [12]. For each document, RCut sorts categories by score and assigns “yes” to each of the  $t$  top-ranking categories, where  $t \in [1 \dots m]$ ,  $m$  is the overall number of categories. For each category,  $C_j$ , PCut sorts the test documents by score and assigns “yes” to each of the  $k_j$  top-ranking documents, where  $k_j$  is the number of documents assigned to  $C_j$ . SCut scores a validation set of documents for each category and tunes the threshold over the local pool of scores, until the optimal performance of the classifier is obtained for that category.

Few threshold-selection algorithms have been proposed [8] [11] [6] for HTC. Starting with a threshold set to 0 and using increments of 0.1, D’Alessio et al. [8] determine the number of documents that would be incorrectly placed in the category with that threshold, and the number of documents that would incorrectly be placed in the category with that threshold. Subsequently, they associate a goodness measure to each threshold value by subtracting the number of incorrect documents from the number of correct documents. The threshold that corresponds to the maximum goodness value is selected. In the event that the same maximum goodness value occurs multiple times, the lowest threshold with that value is selected. Ruiz [11] performs thresholding by selecting thresholds

that optimize the  $F_1$  values for the categories, using the whole training set to select the optimal thresholds. As he works with a modular hierarchical structure, several choices can be made to perform thresholding. In his expert-based system, he makes a binary decision at each expert gate and then optimizes the thresholds using only examples that reach leaf nodes. The automated threshold determination proposed by Ceci and Malerba [6] is based on a bottom-up strategy and tries to minimize a measure based on a tree distance. The algorithm takes as input a category  $C$  and the set of thresholds already computed for some siblings of  $C$  and their descendants. It returns the union of the input set with the set of thresholds computed for all descendants of  $C$ . In particular, if  $C'$  is a direct subcategory of  $C$ , the threshold associated to  $C'$  is determined by examining the sorted list of classification scores and by selecting the middle point between two values in the list, to minimize the expected error.

### 3 Progressive Filtering

Progressive Filtering (PF) is a simple categorization technique that operates on hierarchically structured categories. Given a taxonomy, where each node represents a classifier entrusted with recognizing all corresponding positive inputs (i.e., interesting documents), each input traverses the taxonomy as a “token”, starting from the root. If the current classifier recognizes the token as positive, it is passed on to all its children (if any), and so on. A typical result consists of activating one or more branches within the taxonomy, in which the corresponding classifiers have been activated by the given token.

A way to implement PF consists of unfolding the given taxonomy into pipelines of classifiers, as depicted in Figure 1. Each node of the pipeline represents a category that embeds a binary classifier able to recognize whether or not an input belongs to the category itself.

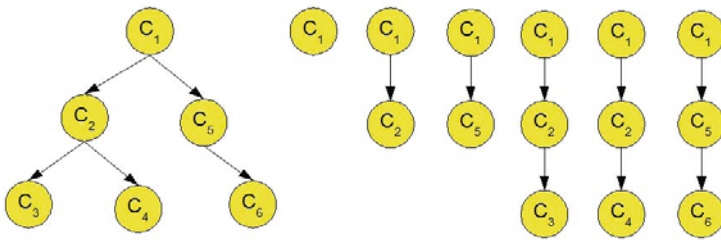


Fig. 1. A taxonomy and its corresponding pipelines

## 4 The Proposed Threshold Selection Algorithm

### 4.1 Motivations

According to classical text categorization, given a set of documents  $D$  and a set of labels  $C$ , a function  $CSV_i : D \rightarrow [0, 1]$  exists for each  $c_i \in C$ . The behavior

of  $c_i$  is controlled by a threshold  $\theta_i$ , responsible for relaxing or restricting the acceptance rate of the corresponding classifier. Let us recall that, with  $d \in D$ ,  $CSV_i(d) \geq \theta_i$  is interpreted as a decision to categorize  $d$  under  $c_i$ , whereas  $CSV_i(d) < \theta_i$  is interpreted as a decision not to categorize  $d$  under  $c_i$ .

In PF, we assume that  $CSV_i$  exists, with the same semantics adopted by the classical setting. Considering a pipeline  $\pi$ , composed by  $n$  classifiers, the acceptance policy strictly depends on the vector of thresholds  $\theta_\pi = \langle \theta_1, \theta_2, \dots, \theta_n \rangle$  that embodies the thresholds of all classifiers in  $\pi$ . In order to categorize  $d$  under  $\pi$ , the following constraint must be satisfied: for  $k = 1 \dots n$ ,  $CSV_i(d) \geq \theta_k$ . On the contrary,  $d$  is not categorized under  $c_i$  in the event that a classifier in  $\pi$  rejects it. Let us point out that we allow different behaviors for a classifier, depending on which pipeline it is embedded in. As a consequence, each pipeline can be considered in isolation from the others. For instance, given  $\pi_1 = \langle C_1, C_2, C_3 \rangle$  and  $\pi_2 = \langle C_1, C_2, C_4 \rangle$ , the classifier  $C_1$  is not compelled to have the same threshold in  $\pi_1$  and in  $\pi_2$  (the same holds for  $C_2$ ). In so doing, the proposed approach performs a sort of “flattening”, though preserving the information about the hierarchical relationships embedded in a pipeline. For instance, the pipeline  $\langle C_1, C_2, C_3 \rangle$  actually represents the classifier  $C_3$ , although the information about the existing subsumption relationships are preserved (i.e.,  $C_3 \prec C_2 \prec C_1$ , where “ $\prec$ ” denotes the usual covering relation).

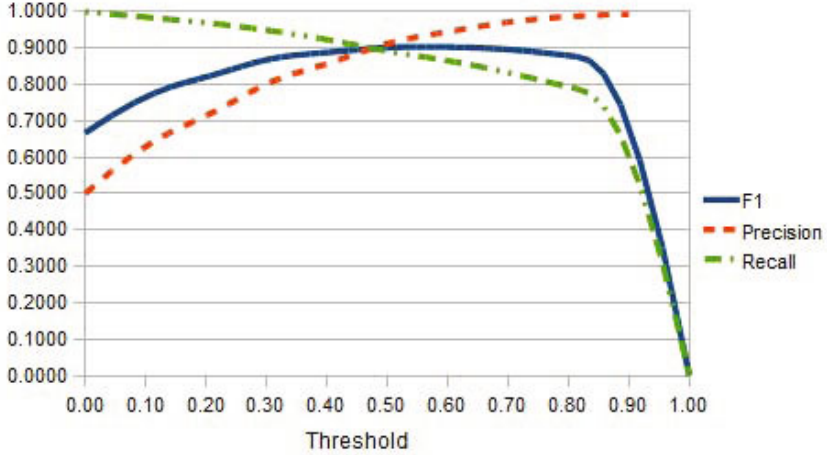
In PF, given a utility function<sup>4</sup>, we are interested in finding an effective and computationally “light” way to reach a sub-optimum in the task of determining the best vector of thresholds. To this end, for each pipeline  $\pi$ , a sub-optimal combination of thresholds is searched for. Unfortunately, finding the best acceptance thresholds is a difficult task. In fact, exhaustively trying each possible combination of thresholds (brute-force approach) is unfeasible, the number of thresholds being virtually infinite. However, the brute-force approach can be approximated by defining a granularity step that requires to check only a finite number of points in the range  $[0, 1]$ , in which the thresholds are permitted to vary with step  $\delta$ . Unfortunately, this “relaxed” brute force algorithm for calibrating thresholds (RBF for short), although potentially useful, is still too heavy from a computational point of view. Thus, in this paper we propose a novel Threshold Selection Algorithm (TSA) characterized by low time complexity, which maintains the capability of finding near-optimum solutions.

## 4.2 The TSA Algorithm

Utility functions typically adopted in TC and in HTC, are nearly-convex with respect to the acceptance threshold. Figure 2 depicts three typical trends of utility functions, i.e., precision, recall, and  $F_1$ .

Setting the threshold of a classifier to 0, no matter which utility function is adopted, forces the classifier to reach its maximum error in terms of false positives (FP). Conversely, setting the threshold to 1 forces the classifier to reach its maximum error in terms of false negatives (FN).

<sup>4</sup> Different utility functions (e.g., precision, recall,  $F_\beta$ , user-defined) can be adopted, depending on the constraint imposed by the underlying scenario.



**Fig. 2.** Example of utility functions

Due to the shape of the utility function and to its dependence on FP and FN, it becomes feasible to search its maximum around a restricted range (i.e., a sub-range of  $[0, 1]$ ). Bearing in mind that the lower the threshold the less restrictive is the classifier, we propose a greedy bottom-up algorithm for selecting decision threshold that relies on two functions:

- *Repair* ( $\mathcal{R}$ ), which operates on a classifier  $C$  by increasing or decreasing its threshold –i.e.,  $\mathcal{R}(up, C)$  and  $\mathcal{R}(down, C)$ , respectively– until the selected utility function reaches and maintains a local maximum.
- *Calibrate* ( $\mathcal{C}$ ), which operates going downwards from the given classifier to its offspring by repeatedly calling  $\mathcal{R}$ . It is intrinsically recursive and at each step it calls  $\mathcal{R}$  to calibrate the current classifier.

Given a pipeline  $\pi = \langle C_1, C_2, \dots, C_L \rangle$ , *TSA* is defined as follows (all thresholds are initially set to 0):

$$TSA(\pi) := \text{for } k = L \text{ downto } 1 \text{ do } \mathcal{C}(up, C_k) \quad (1)$$

which indicates that  $\mathcal{C}$  is applied to each node of the pipeline, starting from the leaf ( $k = L$ ).

Under the assumption that  $p$  is a structure that contains all information about a pipeline, including the corresponding vector of thresholds and the utility function to be optimized, the pseudo-code of *TSA* is:

```
function TSA(p: pipeline):
  for k:=1 to p.length
    do p.thresholds[i] = 0
  for k:=p.length downto 1
    do Calibrate(up, p, k)
  return p.thresholds
end TSA
```

The *Calibrate* function is defined as follows:

$$\begin{aligned} \mathcal{C}(up, C_k) &:= \mathcal{R}(up, C_k) \quad k = L \\ \mathcal{C}(up, C_k) &:= \mathcal{R}(up, C_k); \mathcal{C}(down, C_{k+1}) \quad k < L \end{aligned} \quad (2)$$

and

$$\begin{aligned} \mathcal{C}(down, C_k) &:= \mathcal{R}(down, C_k) \quad k = L \\ \mathcal{C}(down, C_k) &:= \mathcal{R}(down, C_k); \mathcal{C}(up, C_{k+1}) \quad k < L \end{aligned} \quad (3)$$

where the “;” denotes a sequence operator, meaning that in “ $a;b$ ” action  $a$  is performed *before* action  $b$ . The pseudo-code of *Calibrate* is:

```
function Calibrate(dir:{up,down}, p:pipeline, level:integer):
  Repair(dir,p,level)
  if level < p.length then Calibrate(toggle(dir),p,level+1)
end Calibrate
```

where *toggle* is a function that reverses the current direction (from *up* to *down* and vice versa). The pseudo-code of *Repair* is:

```
function Repair(dir:{up,down}, p:pipeline, level:integer):
  delta := (dir = up) ? p.delta : -p.delta
  best_threshold := p.thresholds[level]
  max_uf := p.utility_function()
  uf := max_uf
  while uf >= max_uf * p.sf and p.thresholds[level] in [0,1]
    do p.thresholds[level] := p.thresholds[level] + delta
    uf := p.utility_function()
    if uf < max_uf then continue
    max_uf := uf
    best_threshold := p.thresholds[level]
  p.thresholds[level] := best_threshold
end Repair
```

The scale factor ( $p.sf$ ) is used to limit the impact of local minima during the search, depending on the adopted utility function (e.g., a typical value of  $p.sf$  for  $F_1$  is 0.8).

To better illustrate the algorithm, let us consider the unfolding reported in Figure 3, which corresponds to  $\pi = \langle C_1, C_2, C_3 \rangle$ :

**step 1**

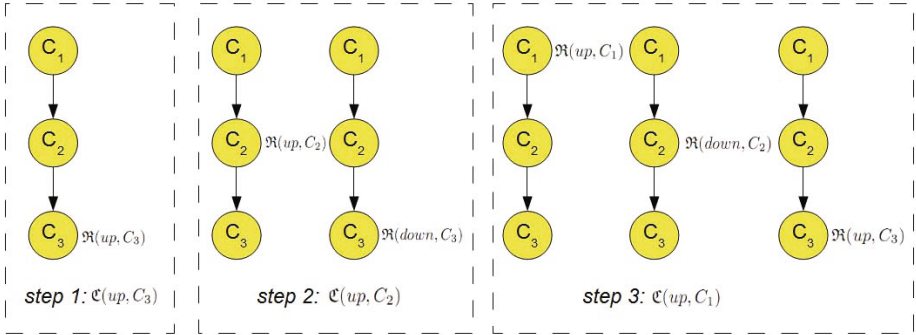
$$\mathcal{C}(up, C_3) = \mathcal{R}(up, C_3)$$

**step 2**

$$\begin{aligned} \mathcal{C}(up, C_2) &= \mathcal{R}(up, C_2); \mathcal{C}(down, C_3) \\ &= \mathcal{R}(up, C_2); \mathcal{R}(down, C_3) \end{aligned} \quad (4)$$

**step 3**

$$\begin{aligned} \mathcal{C}(up, C_1) &= \mathcal{R}(up, C_1); \mathcal{C}(down, C_2) \\ &= \mathcal{R}(up, C_1); \mathcal{R}(down, C_2); \mathcal{C}(up, C_3) \\ &= \mathcal{R}(up, C_1); \mathcal{R}(down, C_2); \mathcal{R}(up, C_3) \end{aligned}$$



**Fig. 3.** Unfolding the threshold-selection procedure for a pipeline composed by three classifiers

Once calculated the sub-optimal combination of thresholds, which (as noted in [10]) depends on the adopted dataset, the pipelines are ready to be used in the corresponding scenario.

### 4.3 Computational Complexity of TSA

Searching for a sub-optimal combination of thresholds in a pipeline  $\pi$  can be actually viewed as the problem of finding a maximum of a utility function  $F$ , the maximum being dependent on the corresponding thresholds  $\theta$ ; in symbols:

$$\theta^* = \underset{\theta}{argmax} F(\theta; \pi) \quad (5)$$

Unfortunately, the above task is characterized by high time complexity, as it is in fact a problem of meta-learning (i.e., a learning problem whose *instances* are learning problems themselves). In particular, two sources of intractability hold: (i) the optimization problem that involves the thresholds and (ii) the need of retraining classifiers after modifying thresholds. In this work, we concentrate on the former issue while deciding not to retrain the classifiers. In any case it is clear that threshold optimization requires a solution that is computationally light. To calculate the computational complexity of TSA, let us define a granularity step that requires to visit only a finite number of points in a range  $[\rho_{min}, \rho_{max}]$ ,  $0 \leq \rho_{min} < \rho_{max} \leq 1$ , in which the thresholds vary with step  $\delta$ . As a consequence,  $p = \lfloor \delta^{-1} \cdot (\rho_{max} - \rho_{min}) \rfloor$  is the maximum number of points to be checked for each classifier in a pipeline. For a pipeline  $\pi$  of length  $L$ , the expected running time for TSA, say  $T(TSA)$ , is proportional to  $(L + L^2) \cdot p \cdot (\rho_{max} - \rho_{min})$ , which implies that TSA has complexity  $O(L^2)$ . In other words, the time complexity is quadratic with the number of classifiers embedded by a pipeline. A comparison between TSA and the brute-force approach is unfeasible, as the generic element of the threshold vector is a real number. However, a comparison between TSA and RBF is feasible although RBF is still computationally heavy. Assuming that  $p$  points are checked for each classifier in a pipeline, the expected running time



for  $RBF$ ,  $T(RBF)$ , is proportional to  $p^L$ , which implies that its computational complexity is  $O(p^L)$ .

To show the drastic reduction of complexity brought by the TSA algorithm, let us consider a pipeline composed of 4 classifiers (i.e.,  $L = 4$ ), and  $p = 100$ . In this case, the orders of magnitude of  $T(RBF)$  and  $T(TSA)$  are  $10^8$  and  $10^3$ , respectively. It is also important noting that, due its intrinsic complexity, the RBF approach can be applied in practice only setting  $p$  to a value much lower than the one applied to TSA. For instance, with  $p_{TSA} = 2000$ ,  $\rho_{max} = 1$  and  $\rho_{min} = 0$ , and  $L = 4$ ,  $T(TSA) \propto 32,000$ . To approximately get the same running time for RBF,  $p_{RBF} \simeq 6.7$ , which is two orders of magnitude lower than  $p_{TSA}$ .

## 5 Experimental Results

The Reuters Corpus Volume I (RCV1-v2) [9] has been chosen as benchmark dataset. In this corpus, stories are coded into four hierarchical groups: Corporate/Industrial (CCAT), Economics (ECAT), Government/Social (GCAT), and Markets (MCAT). Although the complete list consists of 126 categories, only some of them have been used to test our hierarchical approach. The total number of codes actually assigned to the data is 93, whereas the overall number of documents is about 803,000. Each document belongs to at least one category and, on average, to 3.8 categories. To calculate the time complexity of TSA with respect to RBF and to the selected state-of-the-art algorithms, we used the 24 pipelines of depth 4 that end with a leaf node.

Experiments have been performed on a SUN Workstation with two Opteron 280, 2Ghz+ and 8Gb Ram. The system used to perform benchmarks has been implemented using X.MAS [1], a generic multiagent architecture built upon JADE [5] and devised to make it easier the implementation of information retrieval/filtering applications.

Experiments have been carried out by using classifiers based on the  $wk$ -NN technology [7], which do not require specific training and are very robust with respect to noisy data. As for document representation, we adopted the bag of words approach, a typical method for representing texts in which each word from a vocabulary corresponds to a feature and a document to a feature vector. First, all non-informative words such as prepositions, conjunctions, pronouns and very common verbs have been disregarded by using a stop-word list. Subsequently, the most common morphological and inflexional suffixes have been removed by adopting a standard stemming algorithm. After having determined the overall sets of features, their values have been computed for each document resorting to the well-known TFIDF method. To reduce the high dimensionality of the feature space, we locally selected the features that represent a node by adopting the information gain method. During the training activity, each classifier has been trained with a balanced data set of 1000 documents, characterized by 200 (TFIDF) features selected in accordance with their information gain.

Experiments, performed on a balanced dataset of 2000 documents for each class, were focused on (i) calculating the performance improvement of TSA vs. RBF, as done in [3], and on (ii) comparing TSA with two selected

state-of-the-art algorithms, proposed by D’Alessio et al. [8] and by Ceci and Malerba [6], respectively.

**Table 1.** Time comparison between TSA and RBF (in seconds), averaged on pipelines with  $L = 4$

Step	RBF	TSA
Step1 (Level4)	0.007	0.684
Step2 (Level3)	0.142	1.561
Step3 (Level2)	3.017	2.683
Step4 (Level1)	62.276	4.011

**TSA vs. RBF.** A step  $\delta_{TSA} = 5 \times 10^{-4}$  (hence,  $p_{TSA} = 2 \times 10^3$ ) has been adopted to increment thresholds in TSA; whereas a step  $\delta_{RBF} = 5 \times 10^{-2}$  (hence,  $p_{RBF} = 20$ ) has been adopted for the RBF approach<sup>5</sup>. Table 1 illustrates the results comparing the time spent by RBF and TSA. Each row of the table reports the time spent to perform a calibrate step; the last row also coincides with the total elapsed time. Although the ratio  $p_{TSA}/p_{RBF} = 10^2$ , Table 1 clearly shows that the cumulative running time for RBF tends to rapidly become intractable.

**TSA vs. state-of-the-art algorithms.** Before reporting results, let us recall the algorithms used for comparative evaluations. The algorithm proposed by D’Alessio et al. [8] performs an RBF search in the space of thresholds. First, the algorithm sets each child category to 0, then it tunes thresholds scrolling categories depending on their level number. The search is performed incrementing the threshold with steps of 0.1 and determining the number of documents that would be correctly placed in the category with that threshold (i.e., True Positives,  $TP$ ), and the number of documents that would incorrectly be placed in the category with that threshold (i.e., False Positives,  $FP$ ). The goodness measure that must be maximized for each threshold (i.e., the utility function) is calculated by subtracting the number of incorrect documents from the number of correct documents (i.e.,  $TP - FP$ ). The algorithm proposed by Ceci and Malerba [6] is based on a recursive bottom-up threshold determination. Given a vector of thresholds, computed from the scores output by the involved classifiers, the algorithm tries to minimize the classification error for each category. The computation proceeds bottom-up, from the leaves to the root. In fact, a top-down approach would be too conservative, as it would tend to classify documents in a higher category. Moreover, this problem would be enhanced by the fact that wrong decisions taken by high-level classifiers negatively affect low-level classifiers. The error function is estimated on the basis of the distance between two nodes in a tree structure ( $TD$ ), the distance being computed as the sum of the weights of all edges of the unique path connecting the two categories in the hierarchy (a unit weight is associated to each edge).

<sup>5</sup> The motivation of this choice has been discussed in the previous section.

For each algorithm, we performed three sets of experiments in which a different utility function has been adopted. The baseline of our experiments is an experimental comparison among the three algorithms. In particular, we used:

- $F1$ , according to the metric adopted in our previous work on PF [2] and in [11];
- $TP - FP$ , according to the metric adopted in [8];
- $TD$ , according to the metric adopted in [6].

As reported in Table 2, for each experimental setting, we calculated the performance and the time spent for each selected metric. Table 2 summarizes the results. Let us note that, for each experimental setting, the most relevant results (highlighted in bold in the table) are those which correspond to the metric that has been used as utility function. As shown, TSA always performs better in terms of  $F1$ ,  $TP - FP$ , and  $TD$ . As for the running time, the algorithm proposed by Ceci and Malerba shows the best performance. However, let us note that the threshold selection activity is usually made offline, so that the performance in terms of elapsed time does not affect the overall performance of the corresponding text categorizer.

**Table 2.** Comparison between TSA and two state-of-te-art algorithms (UF stands for Utility Function)

<i>UF: F1</i>	<b>F1</b>	<b>TP-FP</b>	<b>TD</b>	<b>Time (s)</b>
TSA	<b>0.8972</b>	702.22	493.09	0.184
D'Alessio et al.	<b>0.8959</b>	697.25	526.5	5.923
Ceci & Malerba	<b>0.8854</b>	682.06	542.19	0.098
<i>UF: TP-FP</i>	<b>F1</b>	<b>TP-FP</b>	<b>TD</b>	<b>Time (s)</b>
TSA	0.8970	<b>703.41</b>	470.47	0.112
D'Alessio et al.	0.8971	<b>701.44</b>	489.28	4.153
Ceci & Malerba	0.8856	<b>685.72</b>	484.44	0.062
<i>UF: TD</i>	<b>F1</b>	<b>TP-FP</b>	<b>TD</b>	<b>Time (s)</b>
TSA	0.8345	621.03	<b>353.93</b>	0.137
D'Alessio et al.	0.8223	606.54	<b>364.15</b>	4.415
Ceci & Malerba	0.8235	609.27	<b>358.09</b>	0.076

## 6 Conclusions and Future Work

After describing TSA, a threshold selection algorithm for hierarchical text categorization, in this paper have been made an experimental assessment aimed at comparing TSA with a relaxed brute-force approach and with two state-of-the-art algorithms. Experimental results confirm the validity of the proposed algorithm.

As for the future work, we are currently implementing the threshold selection algorithm proposed by Ruiz [11]. Furthermore, as real-world data are typically characterized by imbalance, we are performing comparative experiments in a domain in which positive and negative examples are imbalanced.

## Acknowledgments

This research was partly sponsored by the RAS (Autonomous Region of Sardinia), through a grant financed with the “Sardinia POR FSE 2007-2013” funds and provided according to the L.R. 7/2007 “Promotion of the Scientific Research and of the Technological Innovation in Sardinia”.

## References

1. Addis, A., Armano, G., Vargiu, E.: From a generic multiagent architecture to multiagent information retrieval systems. In: AT2AI-6, Sixth International Workshop, From Agent Theory to Agent Implementation, pp. 3–9 (2008)
2. Addis, A., Armano, G., Vargiu, E.: Assessing progressive filtering to perform hierarchical text categorization in presence of input imbalance. In: Proceedings of International Conference on Knowledge Discovery and Information Retrieval, KDIR 2010 (2010)
3. Addis, A., Armano, G., Vargiu, E.: Experimental assessment of a threshold selection algorithm for tuning classifiers in the field of hierarchical text categorization. In: Proceedings of 17th RCRA International Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (2010)
4. Addis, A., Armano, G., Vargiu, E.: Using the progressive filtering approach to deal with input imbalance in large-scale taxonomies. In: Large-Scale Hierarchical Classification Workshop (2010)
5. Bellifemine, F., Caire, G., Greenwood, D.: *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology)*. John Wiley and Sons, Chichester (2007)
6. Ceci, M., Malerba, D.: Classifying web documents in a hierarchy of categories: a comprehensive study. *Journal of Intelligent Information Systems* 28(1), 37–78 (2007)
7. Cost, W., Salzberg, S.: A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning* 10, 57–78 (1993)
8. D’Alessio, S., Murray, K., Schiaffino, R.: The effect of using hierarchical classifiers in text categorization. In: Proceedings of the 6th International Conference on Recherche d’Information Assistée par Ordinateur (RIA/O), pp. 302–313 (2000)
9. Lewis, D., Yang, Y., Rose, T., Li, F.: RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research* 5, 361–397 (2004)
10. Lewis, D.D.: Evaluating and optimizing autonomous text classification systems. In: SIGIR 1995: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 246–254. ACM, New York (1995)
11. Ruiz, M.E.: Combining machine learning and hierarchical structures for text categorization. Ph.D. thesis, supervisor-Srinivasan, Padmini (2001)
12. Yang, Y.: An evaluation of statistical approaches to text categorization. *Information Retrieval* 1(1/2), 69–90 (1999)
13. Yang, Y.: A study of thresholding strategies for text categorization. In: SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 137–145. ACM, New York (2001)

# Improving Tag-Based Recommendation by Topic Diversification

Christian Wartena and Martin Wibbels

Novay, Brouwerijstraat 1, 7523 XC Enschede, The Netherlands  
{Christian.Wartena,Martin.Wibbels}@novay.nl

**Abstract.** Collaborative tagging has emerged as a mechanism to describe items in large on-line collections. Tags are assigned by users to describe and find back items, but it is also tempting to describe the users in terms of the tags they assign or in terms of the tags of the items they are interested in. The tag-based profile thus obtained can be used to recommend new items.

If we recommend new items by computing their similarity to the user profile or to all items seen by the user, we run into the risk of recommending only neutral items that are a bit relevant for each topic a user is interested in. In order to increase user satisfaction many recommender systems not only optimize for accuracy but also for diversity. Often it is assumed that there exists a trade-off between accuracy and diversity.

In this paper we introduce topic aware recommendation algorithms. Topic aware algorithms first detect different interests in the user profile and then generate recommendations for each of these interests. We study topic aware variants of three tag based recommendation algorithms and show that each of them gives better recommendations than their base variants, both in terms of precision and recall and in terms of diversity.

## 1 Introduction

Collaborative tagging has emerged in the past decade as a mechanism to describe items in large collections available on-line. Tags are assigned by users to describe and find back previously viewed items. Thus a ternary relation between users, tags and items is established. In this paper we will investigate some possibilities to construct a user profile based on tags, to identify distinct interests in this profile, and to recommend items relevant to those interests.

When we use a tag based user profile an item is recommended if it is relevant to all tags in the user profile. Similarly, if we use a collaborative filtering approach, we require the item to be similar to all items in the user profile. However, for a user who has some distinct interests, an item that fits the average of all his interests might be less accurate than an item that fits exactly one of his interests. Thus we expect, at least in some cases, that recommendations will improve if we identify different interests in the user profile and take these into account for the recommendation of new items. If a recommendation strategy does so, we will call this strategy *topic aware*. In the following we will make a number

of different algorithms for top-n recommendation topic aware by clustering the tags or items in the profile and generating separate recommendation lists for each topic cluster. The final list of recommendations is obtained by merging the topic specific lists. We do not consider rating prediction.

Lists of recommended items are more interesting to a user if they are more diverse. Thus diversity is regarded as a desirable property of recommendations. In most studies on topic diversification, the diversity is increased by reordering the elements in an initial list of recommended items. If a list is constructed aiming at optimal results for precision and recall, the reordering usually causes a decrease of performance on these evaluation measures. Thus a trade-off between accuracy and diversity emerges. In our approach, however, adding diversity improves precision and recall. We do not re-rank results that are already optimal for precision and recall, but the final diversity of the recommendation is a core property of the recommendation strategy. Thus our method is fundamentally different from the re-ranking approaches: we first distinguish different topics and then generate a list of relevant items.

The remainder of this paper is organized as follows. In the next section we discuss related work. In section 3 we introduce three different tag based recommendation algorithms. In section 4 we discuss topic detection for tagging systems and define topic aware versions of the tag based algorithms. In section 5 we report on an evaluation of the proposed algorithms with data from LibraryThing, a bookmarking service for books.

## 2 Related Work

Most work on recommendation and tagging is about recommending tags. Using tags for item prediction has received less attention. Basically, we find two approaches for tag-based item recommendation. The first approach uses tags to compute item-item or user-user similarities that then are used in classical user or item based nearest neighbor recommendation algorithms. The second approach characterizes both users and items by tag vectors, making it possible to compute the similarity between users and items. The items that are most similar to a user now are recommended.

One of the first papers that integrates tag-based similarities in a nearest neighbors recommender is [1], who extend the user-item matrix with user-tag similarities in order to compute user-user similarities, and extend it with tag-item relations in order to compute item-item similarities. Both similarities are used to compute recommendations. This approach was refined by [2] taking also into account whether users used the tags for the same or for different items. Said et al. [3] use probabilistic latent semantic analysis to model both the user-item matrix and the item-tag matrix. By sharing the latent variables between the models they are able to use the tagging information in the user-item model. Bogers and Van den Bosch [4] use the tag based similarities instead of the classical similarities based on the user-item matrix. They show improvements for item-based nearest neighbor recommendation on various datasets, but did not compare their method to approaches combining both types of similarity.

The second approach, recommendation based on the distance between an item and the tag based user profile, is e.g. followed by Firan et al. [5]. The focus of their work is to determine the optimal set of tags to represent a user. The obvious idea is to represent a user by the tags that he has assigned. However, the resulting tag vector is usually too sparse to compute useful user-item similarities. Some users employ only a very small set of tags, and even for more actively tagging users it might be well the case that a relevant item is tagged only with synonyms of a tag employed by the user. Thus [5] investigate various methods to condense the user profile. The most effective method is to use the tags of all items the user has bookmarked. The same observation was also made by [6]. The problem of the sparse user profiles was also identified by [7]. They solve this problem not by condensing the user profile, but by taking co-occurring tags into account in the computation of similarities. For each user and each tag  $t$  a user specific tag weight is computed. The weight for  $t$  is determined by the weight of the most similar tag  $t'$  in the user profile and the similarity between  $t$  and  $t'$ , where the inter tag similarity is determined by a variant of the Jaccard-coefficient. The relevance of an item finally is the sum of weights of its tags.

In [8] the two approaches sketched above are combined: a nearest neighbor algorithm is used to find an initial set of items and subsequently user-item similarities are computed for the preselected items to obtain a final recommendation. The more interesting aspect of their approach is, that they replace each tag  $t$  in the user profile with the tags co-occurring with  $t$  on items tagged by the user, and restrict the set of tags to the (globally) most popular tags. This results in roughly the same profiles as would be obtained by using the (most popular) tags of all items bookmarked (or tagged) by the user, as we have seen in other approaches.

The problem that many recommender systems tend to recommend a set of very similar items in order to optimize accuracy was noted by [9] and coined the *portfolio effect*. Reordering of recommended elements to alleviate this problem was proposed by [10]. As discussed above, the reordering increases diversity, but at the cost of accuracy. An approach that is very similar to ours is proposed by Zhang and Hurley [11]. They cluster the set of items of a user and apply a item based nearest neighbor recommender to each of the clusters. Finally they merge the results of the sub-recommendation to obtain a list of recommended items for the user. The focus of their work is to avoid that a small set of very popular items is recommended very often at the cost of novel or less common items. The main difference with our recommendation strategy is that all item similarities are based on the user-item matrix, whereas we base similarities on descriptive meta data, especially tags. Zhang and Hurley can improve diversity of the recommendation lists in some cases while the influence of the partitioning on the precision is not very large.

Gemmell et al. [12] propose to cluster tags in a user profile to improve personalized search. They show that clustering improves the search results. Gemmell et al. did not consider recommendation, but our approach for recommendation is both in spirit and results similar to their work. An interesting solution targeting

at recommendation is proposed by [6] who use non-negative matrix factorization to obtain a weak clustering of tags into topics. Now recommendations are computed by using the probability that a user is interested in a topic and the probability that an item is relevant to that topic. This idea comes very close to the approach for topic diversification that we propose below. However, [6] use global tag clusters, whereas we apply clustering to the tags of each user profile. Moreover, we do not sum up the probabilities for an item over all clusters.

### 3 Tag Based Recommendation

In the following we will use two basic strategies for tag based recommendation. We will show that both strategies can be modified easily to become topic aware and that this leads to improvement of recommendation results in all cases. The first type of algorithm we will consider is item based collaborative filtering. The second type of algorithms uses the similarity between an item and a user profile directly. Therefore we call these algorithms profile based.

In all cases the recommendations are based on the tags assigned to the items. To be more precise, we will represent each item by a probability distribution over tags. Consider collections of items  $\mathcal{C} = \{i_1, \dots, i_k\}$ , tags  $\mathcal{T} = \{t_1, \dots, t_l\}$  and users  $\mathcal{U} = \{u_1, \dots, u_m\}$ . Let  $n(i, t, u)$  be the number of times a user  $u$  assigned tag  $t$  to item  $i$ , usually 0 or 1. To consider the tags assigned to an item and the tags assigned by a user, respectively, we let

$$n(i, t) = \sum_{u \in \mathcal{U}} n(i, t, u) \text{ and} \quad (1)$$

$$n(u, t) = \sum_{i \in \mathcal{C}} n(i, t, u). \quad (2)$$

Furthermore, let

$$n_{\mathcal{T}}(t) = \sum_{i \in \mathcal{C}} n(i, t), \quad (3)$$

$$n_{\mathcal{C}}(i) = \sum_{t \in \mathcal{T}} n(i, t) \text{ and} \quad (4)$$

$$n_{\mathcal{U}}(u) = \sum_{t \in \mathcal{T}} n(u, t). \quad (5)$$

Now we define probability distributions  $p_{\mathcal{T}}(t|i)$  and  $p_{\mathcal{C}}(i|z)$  on respectively the set of tags  $\mathcal{T}$  and the corpus  $\mathcal{C}$  that describe how tag occurrences of a given item  $i$  are distributed over different tags, respectively how the occurrences of a given tag  $z$  are distributed over different items:

$$p_{\mathcal{T}}(t|i) = n(i, t)/n_{\mathcal{C}}(i), \quad (6)$$

$$p_{\mathcal{C}}(i|t) = n(i, t)/n_{\mathcal{T}}(t). \quad (7)$$

Finally, for each  $u \in \mathcal{U}$  let  $\mathcal{C}_u$  be the set of items seen/bookmarked by  $u$ . Note that in many cases a user did not tag all the items he has bookmarked.



### 3.1 Item Based Collaborative Filtering

The first strategy for recommendation we use is a nearest neighbor approach ([13]). Given a distance measure between items, we express the relevance of an item for a given user as the average distance between the item and all items bookmarked (or seen) by the user. Formally, we define the distance of an item  $i \in C$  to a user  $u \in U$  as

$$d(u, i) = \frac{\sum_{j \in C_u} d(j, i)}{|C_u|} \quad (8)$$

where  $d(i, j)$  is the distance between items  $i$  and  $j$ . The items with the smallest distance are recommended to the user.

Given our perspective of tag distributions it is natural to use divergences for the distance between items. We will base our distance on the Jensen Shannon divergence, which can be considered as a symmetrical variant of the Kullback Leibler divergence or relative entropy. Since the square root of the Jensen Shannon divergence is a proper metric satisfying the usual axioms of non-negativity, identity of indiscernibles and triangle inequality ([14]), we will use

$$d(i, j) = \sqrt{\text{JSD}(p_{\mathcal{T}}(t|i), p_{\mathcal{T}}(t|j))} \quad (9)$$

where  $\text{JSD}(p, q)$  is the Jensen Shannon divergence or information radius between probability distributions  $p$  and  $q$ .

### 3.2 Profile Based Recommendation

In the nearest neighbor approach we compute the average distance of an item to the items seen by the user. Alternatively, we can compute the distance of an item to the user that also can be represented by a distribution over tags. For each user we compute the characteristic tag distribution  $p(t|u)$ . The obvious way to define this distribution is:

$$p_{\mathcal{T}}(t|u) = n(u, t)/n_{\mathcal{U}}(u). \quad (10)$$

This allows us to define a distance between an item  $i$  and a user  $u$  by setting  $d(u, i) = \sqrt{\text{JSD}(p_{\mathcal{T}}(t|u), p_{\mathcal{T}}(t|i))}$  and to recommend items with a small distance to the user. However, it was already shown by [5] that this strategy does not perform very well. In our experiments the results were even worse than those obtained by the simple non-personalized baseline of recommending the most popular items. The main reason for this bad performance is that the distribution is too sparse and thus many relevant items will be tagged with synonyms of the tags in the user profile, but not with exactly those tags. In the following we will present two possibilities to alleviate this problem.

**Profiles based on item tags.** Firan et al. ([5]) propose several methods to construct better user profiles. One of the most successful ones is to use the tags of the items considered by the user. We define the item based user profile as

$$p'_{\mathcal{T}}(t|u) = \frac{1}{|C_u|} \sum_{i \in C_u} p_{\mathcal{T}}(t|i). \quad (11)$$

**Profiles based on co-occurring tags.** In [15] we have proposed to condense the user profile by adding co-occurring tags. This is achieved by propagating tag probabilities in a Markov chain on  $\mathcal{T} \cup \mathcal{C}$  having transitions  $\mathcal{C} \rightarrow \mathcal{T}$  with transition probabilities  $p_{\mathcal{T}}(t|i)$  and transitions  $\mathcal{T} \rightarrow \mathcal{C}$  with transition probabilities  $p_{\mathcal{C}}(i|t)$ . The characteristic tag distribution for a user now is defined as:

$$\bar{p}_{\mathcal{T}}(t|u) = \sum_{i,t'} p_{\mathcal{T}}(t|i) p_{\mathcal{C}}(i|t') p_{\mathcal{T}}(t'|u). \quad (12)$$

## 4 Topic Aware Recommendation

In the three basic algorithms discussed above the relevance of an item for a user is predicted by its similarity to all items considered by the user or by its similarity to all tags in the user profile. As discussed above this might result in uninteresting lists of similar and unspecific items. In order to recommend items more specific for one of the interests a user might have, we propose to cluster the items or the tags in his profile. Now we can generate lists of recommended items for each of the clusters and merge them to obtain a final recommendation. Thus we can construe topic aware variants of all three algorithms discussed above.

### 4.1 Topic Detection by Clustering Items or Tags

In order to cluster tags or items we need a distance measure between tags or items, respectively. For clustering items we use the item distance defined in (9). For the distance between tags we use the co-occurrence based similarity proposed in [16]. The co-occurrence distribution of a tag  $z$  is defined as

$$\bar{p}_{\mathcal{T}}(t|z) = \sum_{i \in \mathcal{C}} p_{\mathcal{T}}(t|i) p_{\mathcal{C}}(i|z). \quad (13)$$

Now the distance between tags is defined straightforwardly as the square root of the Jensen Shannon divergence of their co-occurrence distributions.

For clustering we use a variant of the complete link algorithm in which in each step we merge two cluster whose merger has a minimal average distance between all elements. This criterion guarantees that at each step the option is chosen that yields the best Calinski Harabasz index [17]. As a stopping criterion, we require the number of clusters to be equal to the square root of the number of tags. This criterion is rather arbitrary but works quite well.

### 4.2 Using Topic Clusters for Recommendation

The topic aware variant of the nearest neighbor algorithm described in section 3.1 can be defined as follows: we cluster the items in  $C_u$  and apply the algorithm to each of the clusters.

In order to merge the recommendation lists the best elements from each cluster are selected. The number of items selected from each recommended list is

proportional to the size of the cluster. Merging starts with the items from the largest cluster. If an item is already added from a previous list, the next item is taken.

For the profile based algorithms it is not that obvious how to make them topic aware. Simply clustering the tags in the profile distribution will result in strange distributions that give bad recommendation results. Some common tags, like *non fiction* in our data set, are relevant for several topics and should not be assigned exclusively to one cluster. For the profiles formed by adding co-occurring tags to the actively used tags this can be achieved by first clustering the tags and then condensing the profile. Thus, for some user  $u \in U$  let  $T_u = \{t \in T \mid n(u, t) > 0\}$ . This set of tags now is clustered into clusters  $T_{u,1}, \dots, T_{u,k}$ . For each cluster  $T_{u,c}$  we compute characteristic tag distributions

$$p_{\mathcal{T}}(t|u, c) = \frac{\sum_i T_{u,c}(t)n(i, t, u)}{\sum_{i,t'} T_{u,c}(t')n(i, t', u)} \text{ and} \quad (14)$$

$$\bar{p}_{\mathcal{T}}(t|u, c) = \sum_{i,t'} p_{\mathcal{T}}(t|i)p_{\mathcal{C}}(i|t')p_{\mathcal{T}}(t'|u, c), \quad (15)$$

where we use  $T_{u,c}$  as the indicator function of the set  $T_{u,c}$ . Note that these formulas are very similar to (10) and (12). Now we can use (15) in the algorithm of section 3.2 to generate recommendations for each topic cluster.

For the profiles based on the tags of all viewed items we again start clustering the tags in the profile and then compute tag distributions for each cluster by adding co-occurring tags. However, in order to end up with distributions over the restricted set of tags used in (11) we compute tag co-occurrence only using the set of items considered by the user. Technically this can be obtained by restricting the item distribution of a tag  $z$  (see (7)) to  $\mathcal{C}_u$  for each user  $u$ :

$$p_{\mathcal{C}}(i|z, u) = \begin{cases} \frac{n(i,z)}{\sum_{i' \in \mathcal{C}_u} n(i',z)} & \text{if } i \in \mathcal{C}_u \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

To obtain the tag distributions for each cluster we substitute  $p_{\mathcal{C}}(i|t', u)$  for  $p_{\mathcal{C}}(i|t')$  in (15). Now it is also natural to use this personalized item distribution for each tag computation of the co-occurrence distribution in (13). Thus each tag gets a personalized co-occurrence distribution and consequently also the distances between tags become personalized. This reflects the fact that different users might use the same tag with a different meaning in different contexts. E.g. for one person the tag *Italy* is related to *food* while for some other user it is more closely related to *renaissance*.

## 5 Evaluation

### 5.1 Dataset

For evaluation we used a selection of data from LibraryThing from [18], that was collected such that each user has supplied tags and ratings to at least 20 books

**Table 1.** Comparison of diversity (average squared distance) for top-10 recommendation lists

Algorithm	MAP	prec@10	div@10
Most Viewed	0,024	0,050	0,56
BPR-MF	0,044	0,090	0,69
Co-occur. tags	0,033	0,064	0,50
Item tags	0,052	0,099	0,50
Nearest Neighbor	0,048	0,075	0,48
Co-occur. tags (TA)	0,037	0,078	0,71
Item tags (TA)	<b>0,084</b>	<b>0,15</b>	0,72
Nearest Neighbor (TA)	0,072	0,12	<b>0,84</b>

and each book has received at least 5 tags. The dataset consists of 37,232 books tagged by 7,279 users with in total 10,559 different tags. The total number of tag assignments is 2,056,487.

In order to validate our recommendation techniques, we split the dataset in a training and a test set, such that the training set contains for each user 80% of his tag assignments. Since there were no time stamps available, the split was made randomly.

## 5.2 Baseline Algorithms

We compare the 6 different tag based recommendation algorithms with 2 baselines that do not use the tags but only the information whether an item was bookmarked by a user or not. As a first baseline we use the strategy that always recommends the most popular items. This recommendation is not personalized and every personalized algorithm should at least be better than this one.

As a second baseline we use one of the strongest recommendation techniques for top- $n$  recommendation, Bayesian Personalized Ranking (BPR), proposed in [19]. We use BPR to learn a matrix factorization model with 16 dimensions. Optimized hyperparameters for this model were determined by grid search on the training data<sup>1</sup>. Tag based recommendation will of course only be useful if its results are better than those obtained by BPR with Matrix Factorization (BPR-MF).

## 5.3 Results

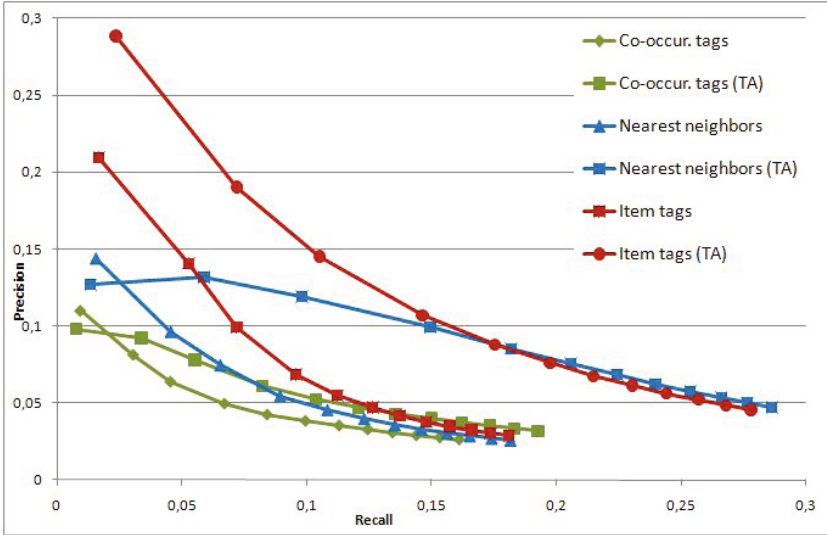
For all algorithms we computed top 1 to 100 recommendations. Since we did not compute a complete ranking of all 37,232 items for all users and all algorithms we will use the mean average precision (MAP) computed over the first 100 recommended items as a measure to compare the algorithms. Furthermore, for all

<sup>1</sup> Regularization parameters for matrix factorization are:  $\lambda_w = 0,25$ ,  $\lambda_{H+} = 0,0001$ ,  $\lambda_{H-} = 0,01$ . The learning rate is set to 0,02 and 160 iterations are user to compute the model.

algorithms we also compare precision and diversity of a top 10 recommendation. As a measure of diversity we take the average squared distance between the recommended items. The diversity for a set items  $I$  is thus defined as

$$div(I) = \frac{\sum_{i,j \in I} JSD(q(i), q(j))}{|I|^2} \quad (17)$$

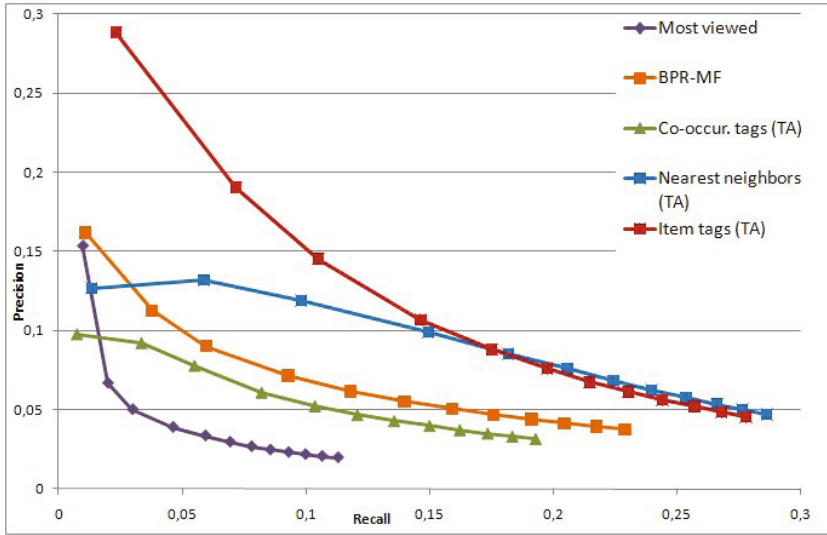
where JSD is the Jensen-Shannon divergence, which is the square of a proper distance measure. The results for MAP, precision at 10 and diversity at 10 are given in Table [11](#).



**Fig. 1.** Precision and recall for 3 proposed topic aware algorithms compared to their basic (non-topic aware) variants, using a sample of LibraryThing data. Data points are plotted for top- $n$  recommendation with  $n = 1, 5, 10, 20 \dots 100$ .

In Figure [11](#) precision and recall for the 6 tag based algorithms are shown. Comparison of the algorithms clearly shows that each of the 3 discussed basic algorithms benefits from making them topic aware in the proposed way. Only for a top 1 and a top 2 the topic aware variants of the tag based collaborative filtering and the algorithm using a profile based on co-occurring tags have a lower precision than their non-topic aware variants. In these cases the number of clusters is larger than the number of items that we have to predict. Thus, most of the available information is not used. Remarkably, the third topic aware algorithm does not show a similar behavior for small recommendation lists.

If we compare the algorithms to the baselines (Figure [12](#)) we see that all algorithms are clearly better than the non-personalized most viewed recommendation. T-tests show that differences between each tag aware algorithm and its base variant for the MAP and for prec@10 are significant at the  $p < 0,001$  level. Also all differences to the base-line algorithms are significant at the same level.



**Fig. 2.** Precision and recall for 3 proposed topic aware algorithms compared to 2 base lines, using a sample of LibraryThing data. Data points are plotted for top- $n$  recommendation with  $n = 1, 5, 10, 20, \dots 100$ .

The results clearly show that tags can be very useful for recommendation. Finally, we see that the diversity for the topic aware algorithms is clearly higher than for the non-topic aware content based algorithms. Interestingly, also the diversity of BPR-MF is relatively high.

## 6 Conclusion and Future Work

We have shown that clustering user tags can significantly improve tag based item recommendation. This corresponds to the intuition that people have different interests and that it is better to recommend items on each separate topic than trying to find items that match more or less all interests. Though this is very intuitive, it is nevertheless a surprising result. Given results from previous research, we expect that improvement of diversity has to be paid for by loss of precision and recall. Our clustering approach in contrary improves both, diversity and precision and recall.

Another nice aspect of the proposed algorithms is that it is easy to explain to users why items are recommended: The topics detected can be displayed, e.g. by a cloud of most frequent or most central tags. The recommendation then can be motivated by the relevance of the items for one of the detected topics.

The two algorithms that turned out to be the best ones, do not or almost not rely on the fact that the user is an active tagger. Thus these methods can be applied for content based recommendation in general. An interesting question for future research is, to what type of item sets and meta-data the results carry over.

## Acknowledgments

This research has received funding from the European Community's Seventh Framework Program within the MyMedia project (grant agreement N<sup>o</sup> 215006) and the PetaMedia network of excellence (grant agreement N<sup>o</sup> 216444). We would like to thank Zeno Gantner (University of Hildesheim) for his help with BPR-MF and Rogier Brussee (Hogeschool Utrecht) for many helpful comments.

## References

1. Tso-Sutter, K.H.L., Balby Marinho, L., Schmidt-Thieme, L.: Tag-aware recommender systems by fusion of collaborative filtering algorithms. In: Wainwright, R.L., Haddad, H. (eds.) SAC, pp. 1995–1999. ACM, New York (2008)
2. Liang, H., Xu, Y., Li, Y., Nayak, R.: Tag based collaborative filtering for recommender systems. In: Wen, P., Li, Y., Polkowski, L., Yao, Y., Tsumoto, S., Wang, G. (eds.) RSKT 2009. LNCS, vol. 5589, pp. 666–673. Springer, Heidelberg (2009)
3. Said, A., Wetzker, R., Umbrath, W., Hennig, L.: A hybrid plsa approach for warmer cold start in folksonomy recommendation. In: Proceedings of the RecSys 2009 Workshop on Recommender Systems & The Social Web (2009)
4. Bogers, T., van den Bosch, A.: Collaborative and Content-based Filtering for Item Recommendation on Social Bookmarking Websites. In: Proceedings of the ACM RecSys 2009 Workshop on Recommender Systems & the Social Web, New-York, NY, USA, pp. 9–16 (2009)
5. Firan, C.S., Nejd, W., Paiu, R.: The benefit of using tag-based profiles. In: Almeida, V.A.F., Baeza-Yates, R.A. (eds.) LA-WEB, pp. 32–41. IEEE Computer Society, Los Alamitos (2007)
6. Peng, J., Zeng, D.: Exploring information hidden in tags: A subject-based item recommendation approach. In: Proceedings of Nineteenth Annual Workshop on Information Technologies and Systems (WITS 2009), Phoenix, Arizona, USA (2009)
7. Hung, C., Huang, Y., Hsu, J., Wu, D.: Tag-Based User Profiling for Social Media Recommendation. In: Workshop on Intelligent Techniques for Web Personalization & Recommender Systems at AAAI 2008, Chicago, Illinois (2008)
8. Liang, H., Xu, Y., Li, Y., Nayak, R., Weng, L.T.: Personalized recommender systems integrating social tags and item taxonomy. In: [20], pp. 540–547
9. Ali, K., van Stam, W.: Tivo: making show recommendations using a distributed collaborative filtering architecture. In: Kim, W., Kohavi, R., Gehrke, J., DuMouchel, W. (eds.) KDD, pp. 394–401. ACM, New York (2004)
10. Ziegler, C.N., McNeel, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: Ellis, A., Hagino, T. (eds.) WWW, pp. 22–32. ACM, New York (2005)
11. Zhang, M., Hurley, N.: Novel item recommendation by user profile partitioning. In: [20], pp. 508–515
12. Gemmell, J., Shepitsen, A., Mobasher, B., Burke, R.D.: Personalizing navigation in folksonomies using hierarchical tag clustering. In: Song, I.Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2008. LNCS, vol. 5182, pp. 196–205. Springer, Heidelberg (2008)
13. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: Brusilovsky, P., Kobsa, A., Nejd, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 325–341. Springer, Heidelberg (2007)

14. Fuglede, B., Topsoe, F.: Jensen-shannon divergence and hilbert space embedding. In: Proc. of the Internat. Symposium on Information Theory, p. 31 (2004)
15. Wartena, C., Brussee, R., Wibbels, M.: Using tag co-occurrence for recommendation. In: ISDA, pp. 273–278. IEEE Computer Society, Los Alamitos (2009)
16. Wartena, C., Brussee, R.: Instance-based mapping between thesauri and folksonomies. In: International Semantic Web Conference, pp. 356–370 (2008)
17. Caliński, T., Harabasz, J.: A dendrite method for cluster analysis. *Communications in Statistics Simulation and Computation* 3(1), 1–27 (1974)
18. <http://dmirlab.tudelft.nl/users/maarten-clements>
19. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009 (2009)
20. Main Conference Proceedings of 2009 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2009, Milan, Italy, September 15-18. IEEE, Los Alamitos (2009)



# A Joint Model of Feature Mining and Sentiment Analysis for Product Review Rating

Jorge Carrillo de Albornoz, Laura Plaza,  
Pablo Gervás, and Alberto Díaz

Universidad Complutense de Madrid,  
Departamento de Ingeniería del Software e Inteligencia Artificial,  
Madrid, Spain  
{jcalbornoz,lplazam,albertodiaz}@fdi.ucm.es,  
pgervas@sip.ucm.es

**Abstract.** The information in customer reviews is of great interest to both companies and consumers. This information is usually presented as non-structured free-text so that automatically extracting and rating user opinions about a product is a challenging task. Moreover, this opinion highly depends on the product features on which the user judgments and impressions are expressed. Following this idea, our goal is to predict the overall rating of a product review based on the user opinion about the different product features that are evaluated in the review. To this end, the system first identifies the features that are relevant to consumers when evaluating a certain type of product, as well as the relative importance or *salience* of such features. The system then extracts from the review the user opinions about the different product features and quantifies such opinions. The salience of the different product features and the values that quantify the user opinions about them are used to construct a *Vector of Feature Intensities* which represents the review and will be the input to a machine learning model that classifies the review into different rating categories. Our method is evaluated over 1000 hotel reviews from *booking.com*. The results compare favorably with those achieved by other systems addressing similar evaluations.

**Keywords:** automatic product rating, feature mining, polarity detection, sentiment analysis.

## 1 Introduction and Background

During the last decade, product review forums have become commonplace, and an increasing number of websites provide platforms for customers to publicize their personal evaluations and opinions of products and services. The information in product reviews is of great interest to both companies and consumers. Companies and organizations spend a huge amount of money to find customers' opinions and sentiments, since this information is useful to exploit their marketing-mix in order to affect consumer satisfaction. Individuals are interested in others' opinions when purchasing a product or hiring a service. In fact, according to a

survey of ComScore<sup>1</sup>, online customer-generated reviews have significant impact on purchase decision, so that consumers are willing to pay at least 20% more for services receiving an *Excellent*, or 5-star, rating than for the same service receiving a *Good*, or 4-star, rating.

This situation has raised many NLP challenges, commonly referred to as *Sentiment Analysis*, such as *subjectivity detection*, *polarity recognition* and *rating inference*. Subjectivity detection aims to discover subjective or neutral terms, phrases or sentences, and it is frequently used as a previous step in polarity and rating classification [1,2,3]. Polarity recognition attempts to classify texts into positive or negative [4,5,6]. The rating inference task goes a step further and tries to identify different degrees of positivity and negativity, e.g. *strongly-negative*, *weakly-negative*, *fair*, *weakly-positive* and *strongly-positive* [6,7,8,9].

Focusing on product review classification, various approaches have been proposed during the last decade. Most of them only consider the polarity of the opinions (i.e. negative *vs.* positive) and rely on machine learning (ML) techniques trained over vectors of linguistic feature frequencies. Pang et al. [4], for instance, present a comparison between three ML algorithms trained on the frequencies of positive and negative terms, and conclude that unigram-based SVM classifiers can be efficiently used in polarity classification of movie reviews. Martineau and Finin [10] use a similar approach on the same corpus where the words are scored using a Delta TF-IDF function before classifying the reviews into positive and negative. A more ambitious task is proposed by Brooke [7], whose goal is to classify reviews of different types of products into three and five rating classes, respectively, using a set of linguistic features including intensification, negation, modality and discourse structure. However, none of these approaches take into account other factors that affect the polarity of an opinion, and especially the strength of this polarity, such as the aspects or features on which the reviewer opinions are expressed and the relations between them. We hypothesize that humans have a conceptual model of what is relevant regarding a certain product or service that clearly influences the polarity and strength of their opinions. For instance, when evaluating a hotel, reviewers seem to be mainly concerned about its location, cleanliness, staff, etc; whereas other aspects, such as nearby shops and restaurants or the bed size, are less important. Therefore, we argue that, to successfully understand the user opinion about a product, it is necessary to combine feature mining and sentiment analysis strategies.

This assertion is not novel, others have noticed it [11,12,13]. Carenini and colleagues [11] present a system for summarizing evaluative arguments which relies on the detection of features of the entity that is evaluated. They use the association rule mining approach presented in [14] to obtain a first list of features. Since the number of features can be unmanageable (around 100-200 features per product), they use an *ad hoc* set of *User-Defined Features (UDF)* to reduce this list. Tivov and McDonald [12] propose a statistical model which is able to discover topics or rating aspects and to extract textual evidence from reviews supporting each of these ratings. They evaluate their approach on a corpus of

---

<sup>1</sup> ComScore, <http://www.comscore.com/>. Last visited on 15 October 2010.

hotel reviews from *TripAdvisor.com*. This approach has two main limitations: first, it needs a pre-defined set of aspects for the purpose of extraction, which also have to be accompanied by a user rating (e.g. *Food: 2; Decor: 1; Service: 3; Value: 2*). This information is not usually available in most review collections, where users usually give a unique score that represents their overall rate for the product along with a free-text describing their opinions about one or more product aspects. Second, their system describes the product aspects using expressions such as “great reception” or “helpful staff” for the aspect *Service*. In our opinion, the words “great” and “helpful” in the previous expressions should not be considered representative of the aspect *Service* of a hotel, but may affect other aspects (e.g. “great room” or “helpful shuttle service”). Kim and Hovy [13] present a system that automatically extracts the pros and cons from online reviews by finding the holder and the topic of the opinion. However, they do not quantify the strength of these pros and cons, nor they predict the overall rating of the reviews.

In this paper, we focus on measuring the polarity and strength of opinions, especially in those expressed in product reviews. We propose a model that leverages the user opinion on the different product features to predict the rating of a review. The model works in 4 phases. First, it identifies the features that are important to consumers when evaluating a certain type of product. Second, it locates in the review the sentences where the user opinions on the different product features are stated. Third, it computes the polarity and strength of the opinion expressed in each sentence. Finally, it computes a single score for each feature, based on the polarity of the sentences associated to it, and constructs a *Vector of Feature Intensities* which represents the review and will be the input to a machine learning algorithm that predicts a rating for the review.

Our approach improves previous work in three main points. First, it does not make use of any previous knowledge about the product features that are relevant to the user, but discovers them automatically from a set of reviews using an unsupervised model. This allows the system to be directly portable to new types of products and services. Second, the set of discovered features is small and meaningful enough for the user, but each feature is defined by a number of concepts able to accurately describe it, independently of the vocabulary used. Third, the system estimates the weight of each product feature in the overall user opinion to predict a more precise rating.

## 2 Data Collection: The *HotelReview* Corpus

We collected 25 reviews per hotel for 60 different hotels (1500 reviews) from *booking.com*<sup>2</sup>. Each review contains the following information:

- The city where the hotel is located, the reviewer nationality, the date when the review was written and the type of reviewer from a set of 7 categories, such as *solo traveler*, *young couple* and *group*.

---

<sup>2</sup> <http://www.booking.com/>

- A score in 0-10 describing the overall opinion of the reviewer. This score is not given by the reviewer, but automatically calculated by *booking.com* from the rates assigned by the reviewer to 5 aspects: *Hotel staff*, *Services/facilities*, *Cleanliness of your room*, *Comfort*, *Value for money* and *Location*. Unfortunately, these disaggregated scores are not available in the reviews.
- A brief free-text describing, separately, what the reviewer liked and disliked during the stay in the hotel.

We have observed that the overall score assigned to a review frequently bears no relation at all with the text describing the user opinion about the hotel, so that two reviews with nearly the same score may reflect very different opinions. For instance, the two following reviews are assigned the score ‘6.5’, but the second is clearly more negative than the first:

- *Good location. Nice roof restaurant - (I have stayed in the baglioni more than 5 times before). Maybe reshaping/redecorating the lobby.*
- *Noisy due to road traffic. The room was extremely small. Parking awkward. Shower screen was broken and there was no bulb in the bedside light.*

To overcome this drawback, we asked two annotators to assign a first category within the set [*Excellent*, *Good*, *Fair*, *Poor*, *Very poor*] and a second category within the set [*Good*, *Fair*, *Poor*] to each review based on the text describing it. To solve inter-judge disagreement, all the reviews with conflicting judgments were removed. Finally, we randomly selected 1000 reviews<sup>3</sup>. The final distribution of the reviews is 200 for each class in the 5-classes categorization and 349, 292 and 359, respectively, in the 3-classes categorization. An example of hotel review is shown in Table 1.

**Table 1.** An example of hotel review from the *HotelReview* corpus

---

<p>&lt;HotelReview idDoc="D_8" hotelID="H_2" hotelLocation="Paris" reviewerCategory="Young couple" reviewerNationality="Belgium" date="February 10, 2010" score="9.3" 5_classes.intensity="Good"&gt; 3_classes.intensity="Good"&gt;</p> <p>&lt;PositiveOpinion&gt; I liked the location, breakfast was nice as well as Tea Time Buffet, that was really nice. Parking on weekends is free (on the street, and it's safe). We got to the room and it was very smelly (cigarettes) so we asked and changed and got a nice room. I'd recommend this hotel definitely. But hey... it's not a 4 star hotel...&lt;/PositiveOpinion&gt;</p> <p>&lt;NegativeOpinion&gt;Staff is nice except for one receptionist (a man) at night, he was not helpful at all, I asked him for directions and he said there's nothing I can do if you don't know Paris. Anyways, everybody else was nice.&lt;/NegativeOpinion&gt;</p> <p>&lt;/HotelReview&gt;</p>
--

---

### 3 Automatic Product Review Rating

In this section we present a novel approach to product review rating. The method is based on identifying the features of concern to the consumers of a product,

<sup>3</sup> This collection is available for research purposes.

<http://nil.fdi.ucm.es/index.php?q=node/456>

extracting the product features that have been commented on by the reviewer, and weighting the comments on each product feature to estimate the overall sentiment of the reviewer about the product.

### 3.1 Step I: Detecting Salient Product Features

The aim of this step is to identify the features that are relevant to consumers when evaluating products of a certain type, as well as the relative importance or *salience* of such features. To this end, we adapt the summarization method presented in [15], which we explain here for completeness.

Given a set of reviews for a number of products of the same type, we first apply a shallow pre-processing over the text describing the users' opinions, including POS tagging and removing stopwords and high frequency terms. We next translate the text into WordNet concepts using the *lesk* algorithm [16] to disambiguate the meaning of each term according to its context. After that, the WordNet concepts for nouns are extended with their hypernyms, building a graph where the vertices represent distinct concepts in the text and the edges represent *is-a* relations between them. Our experimental results have shown that the use of verbs in this graph includes very general information that negatively affects the rating prediction step. Regarding adjectives and adverbs, we do not consider words from these grammatical categories to represent the product features, but to express the user opinions about them.

We next expand the graph with a semantic similarity relation, so that a new edge is added that links every pair of leaf vertices whose similarity exceeds a certain threshold. To calculate this similarity, different measures have been tested. Finally, each edge is assigned a weight in  $[0,1]$ . This weight is calculated as the ratio between the relative positions in their hierarchies of the concepts linked by the edge.

The vertices are next ranked according to their salience or prestige. The *salience* of a vertex,  $v_i$ , is calculated as the sum of the weight of the edges connected to it multiplied by the frequency of the concept represented by  $v_i$  in the set of reviews. The top  $n$  vertices are grouped into *Hub Vertex Sets (HVS)* [17], which represent sets of concepts strongly related in meaning. A degree-based clustering method is then run over the graph to obtain a non-predefined number of clusters, where the concepts belonging to the HVS represent the centroids. The working hypothesis is that each of these clusters represents a different product feature. Figure 1a shows the highest salience concepts or centroid for each of the 18 feature clusters generated from a set of 1500 hotel reviews from *booking.com* using the Jiang and Conrath [18] similarity measure and a 0.25 similarity threshold to build the graph. In Figure 1b, all the concepts belonging to the feature cluster “**room**” are shown.

### 3.2 Step II: Extracting the User Opinion on Each Product Feature

Once the system knows the product features or aspects that are of concern to consumers, the next step is to extract from the review the opinions expressed on



**Fig. 1.** (a) Highest salience concept for each product feature. (b) Concepts belonging to feature **room**. A bigger letter size indicates a higher salience.

such features. Thus, we need to locate in the review all textual mentions related to each product feature. To do this, we map the reviews to WordNet concepts in the same way than in the previous step, and we associate the sentences in the review to the product features they refer to using three heuristics:

- **Most Common Feature (MCF)**: The sentence is associated to the feature with which it has more WordNet concepts in common.
- **All Common Features (ACF)**: Since a sentence may contain information related to different features, we associate the sentence to every feature with some concept in common.
- **Most Salient Feature (MSF)**: For each feature and sentence, we compute a score by adding the salience of the concepts in the sentence that are also found in the feature cluster. Then, the sentence is associated to the highest score feature.

It must be noted that a sentence may consist only of concepts not included in any feature cluster, so that it cannot be associated to any of them. To avoid losing the information in these sentences we create a further cluster (*other features*), and associate these sentences to it<sup>4</sup>.

### 3.3 Step III: Quantifying the User Opinions

We next aim to quantify the opinion expressed by the reviewer on the different product features. To this end, we predict the polarity of the sentences associated to each feature. Since it is unlikely that a user will annotate every sentence in a review as being positive or negative, we use the polarity recognition system presented in Carrillo de Albornoz et al. [9]. The main idea of this method is to extract the WordNet concepts in a sentence that entail an emotional meaning, assign them an emotion within a set of categories from an affective lexicon, and use this information as the input of a logistic regression model to predict the polarity of the sentence and the probability of this polarity. The main points of this approach, as pointed by the authors, are: (1) the use of WordNet and a word sense disambiguation algorithm, which allows the system to work with concepts rather than terms, (2) the use of emotional categories instead of terms as classification attributes, and (3) the use of negations and quantifiers to invert, increase or dismiss the intensity of these emotions. This system has been shown to outperform previous systems which aim to solve the same task.

<sup>4</sup> We tried ignoring these sentences and found it to be less effective.

For instance, when the system is run over the sentence “*In the room, there were no problems with the heating system, so even if outside it was quite freezing, in the hotel was warm enough*”, the sentence is classified as positive with a probability of 0.963.

### 3.4 Step IV: Predicting the Rating of a Review

Once all the relevant product features are extracted, and the user opinions on each feature are quantified, the system should aggregate this information to provide an average rating for the review (e.g. *Good, Fair and Poor*). We translate the product review into a *Vector of Feature Intensities (VFI)*. A VFI is a vector of  $N+1$  values, each one representing a different product feature and the *other features*. We experiment with two strategies for assigning values to the VFI positions:

- **Binary Polarity (BP)**: For each sentence in the review, the position of the feature to which the sentence has been assigned is increased or decreased by 1, depending on whether the sentence was predicted as positive or negative.
- **Probability of Polarity (PP)**: Similar to the previous one, but the feature position is increased or decreased by the probability of the polarity assigned to the sentence by the polarity classifier.

For instance, the review shown in Table 1 will produce the following VFI when the MSF heuristic and the BP strategy are used, and the set of features in Figure 1a is considered:  $[-1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]$ . In this review, the sentence “*I liked the location...*” is labeled by the polarity classifier as positive, and assigned to the feature represented by the concept **location** (12th position in the VFI). Even if this sentence contains information related to concepts from other features (e.g. breakfast and buffet, from feature **breakfast**), location presents a higher salience and so the sentence is assigned to it, adding 1.0 to its vector position. Similarly, the sentence “*Parking on weekends...*” is labeled as positive and assigned to the **street** feature (14th position in the VFI). The sentence “*We got to the room...*” is labeled as negative and assigned to the feature **room** (1st position in the VFI), decreasing this position in 1.0. In turn, the sentences “*I’d recommend this hotel...*” and “*But hey... it’s not...*” are both assigned to the feature **hotel** (4th position in the VFI) but, since the first one is classified as positive and the last one is classified as negative, their scores are neutralized. Finally, the sentence “*Staff is nice except for...*” is assigned to the **staff** feature (9th position in the VFI) with a negative intensity, and the sentence “*Anyways, everybody else was nice*” is assigned to the **other features** (the last position in the VFI) with a positive intensity.

The VFI is finally used as the input to a machine learning algorithm that classifies the review into different rating categories.

## 4 Evaluation

### 4.1 Evaluation Setup

We use the *HotelReview* collection described in Section 2 for evaluating the method. This collection contains 1000 reviews manually labeled within two different sets of categories: *[Good, Fair, Poor]* and *[Excellent, Good, Fair, Poor, Very Poor]*. To determine the best ML algorithm for review rating, several Weka classifiers were trained and evaluated using 10-fold cross validation. We only show the results of the three best performance classifiers: a logistic regression model (*Logistic*), a support vector machine (*LibSVM*) and a functional tree (*FT*). Furthermore, since the sentences in the reviews in the *HotelReview* corpus are labeled with either a positive or negative polarity, depending on whether they appear in the *<PositiveOpinion>* or *<NegativeOpinion>* section of the review, we use these labeled sentences to train the polarity classifier described in Section 3.3, again using 10-fold cross validation.

We test the system using different hotel feature sets. As explained in Section 3.1, these sets depend on three parameters: the set of reviews, the similarity measure and the similarity threshold used to build the graph. We have experimented using 50, 1000 and 1500 reviews randomly selected from *booking.com*; three different similarity measures (Lesk [16], Jiang & Conrath [18], and Lin [19]); and various similarity thresholds from 0.1 to 0.5. We have found that the best results are achieved when the Jiang & Conrath similarity is used and the similarity threshold is set to 0.25. Thus, we only report here the results for the three feature sets obtained using this similarity and threshold: **Feature set 1**, which is built from 50 reviews and consists of 24 features and 114 concepts; **Feature set 2**, which is built from 1000 reviews and consists of 18 features and 330 concepts; and **Feature set 3**, which is built from 1500 reviews and consists of 18 features and 353 concepts.

We also compare our system with two state-of-art approaches. The first one is the SVM over bags of unigrams approach presented in Pang et al. [4]. Let  $u_1, \dots, u_m$  be a set of  $m$  unigrams that can appear in a review. Let  $f_i(r)$  be the number of times  $u_i$  occurs in the review  $r$ . Then,  $r$  is represented by the review vector  $R = f_1(r), \dots, f_m(r)$ . The set of  $m$  unigrams is limited to unigrams appearing at least 4 times in the 1000-review corpus. In spite of its simplicity, this approach turned out to be more effective than more complex approaches using features such as bigrams, part-of-speech tagging and word positions, and might actually be considered somewhat difficult to beat. The second approach is just the Carrillo de Albornoz et al. [9] algorithm for sentence polarity prediction presented in Section 3.3. In order to work with reviews rather than sentences, the whole text in a review is considered as a unique sentence.

### 4.2 Results

We first examine the effect of the product feature set on the review classification. Table 2 shows the accuracy for three Weka classifiers in a 3-classes task (i.e. *Good*,



*Fair* and *Poor*), using the three feature sets presented in the previous section. For these experiments, we use the *Binary Polarity* strategy for assigning values to the VFI attributes, as explained in Section 3.4.

**Table 2.** Average accuracies for different classifiers, using different feature sets and sentence-to-feature assignment strategies

Method	Feature Set 1			Feature Set 2			Feature Set 3		
	<i>MCF</i>	<i>ACF</i>	<i>MSF</i>	<i>MCF</i>	<i>ACF</i>	<i>MSF</i>	<i>MCF</i>	<i>ACF</i>	<i>MSF</i>
Logistic	69.8	67.7	69.8	70.4	67.4	<b>70.8</b>	69.1	67.4	70
LibSVM	69	67.1	69.2	69	67.8	<b>69.2</b>	68.8	67.7	69
FT	66.8	64.2	66.8	66.3	65.2	<b>68.6</b>	68.4	65.8	68.4

As shown in Table 2, the *Feature set 2* reports the best results for all classifiers. However, the accuracy differs little across different feature sets, which seems to indicate that increasing the number of reviews used for extracting the features does not necessarily improve the accuracy of rating prediction. As a result of this finding, we use the *Feature set 2* in the remaining experiments.

We also aim to determine which of the three heuristics for sentence-to-feature assignment produces the best outcome (see Section 3.2). As it may be seen from Table 2, the *MSF* heuristic reports the best results for most algorithms and feature sets, but the *MCF* also reports very close accuracies. In contrast, the *ACF* heuristic produces significantly worse results. Although intuitively any information about a product feature in a sentence should be taken into account by the classifier, these results seem to indicate that only the main feature in each sentence provides useful information for the rating prediction task. On the other hand, we have observed that the heuristics *MCF* and *MSF* produce very similar sentence-to-feature mappings, which leads to similar classification results.

We next examine the use of the *Probability of Polarity* strategy for assigning values to the VFI (see Section 3.4). As shown in Table 3, the use of this strategy improves the average accuracy for all ML algorithms. The best performance (71.7%) is achieved by *Logistic*, increasing its accuracy by as much as 0.9% beyond that of the *Binary Polarity* strategy (Table 2). We presume that this probability of polarity somehow captures the degree of negativity/positivity of a sentence, which results in useful information for the classifier, since it is clearly not the same to say “*The bedcover was a bit dirty*” than “*The bedcover was terribly dirty*”. Finally, it may be also observed that the results produced by the system in all ML techniques significantly outperform those of the *Pang et al.* [4] and *Carrillo de Albornoz et al.* [9] approaches.

We finally tested our approach in a 5-classes prediction task (e.g. *Excellent*, *Good*, *Fair*, *Poor* and *Very Poor*). The results (Table 4) considerably decrease with respect to the 3-classes task, but are still significantly better than those achieved by the *Pang et al.* [4] and *Carrillo de Albornoz et al.* [9] approaches. This result was expected, since it is a more difficult task. However, to find out further reasons for this decrease, we examined the confusion matrix and discovered that most classification errors come from *Good* and *Poor* instances which are classified

**Table 3.** Average results for different classifiers in the 3-classes prediction task

Method	Acc.	Good		Fair		Poor	
		Pr.	Re.	Pr.	Re.	Pr.	Re.
Logistic	<b>71.7</b>	77.3	82.5	58.6	46.6	74	81.7
LibSVM	69.4	73.6	83	57.2	38	71.1	81.7
FT	66.9	73	76.9	50.8	43.2	71.4	76.5
Carrillo de Albornoz et al. [9]	66.7	71.7	82.7	48.7	32.9	70.4	78.5
Pang et al. [4]	54.2	63.6	61.8	38.7	39.7	58.1	58.5

**Table 4.** Average results for different classifiers in the 5-classes prediction task

Method	Acc.	Excellent		Good		Fair		Poor		Very Poor	
		Pr.	Re.	Pr.	Re.	Pr.	Re.	Pr.	Re.	Pr.	Re.
Logistic	<b>46.9</b>	52.6	65	39.9	30.5	38.3	38.5	41.7	40	58.5	60.5
LibSVM	45.3	52.3	68	33.1	20.5	37.4	36.5	37.3	40.5	59.8	61
FT	43.7	49	59.5	27.6	18.5	37.6	37	39.7	35.5	55.1	68
Carrillo de Albornoz et al. [9]	43.2	51.4	81.5	38.7	23	40.9	13.5	31.6	55.5	57.8	42.5
Pang et al. [4]	33.5	59.7	46	26.3	39	29.9	26	26.5	28	34.8	28.5

as *Excellent* and *Very Poor* respectively. We also checked that most problematic instances correspond to borderline cases where the final classification involves some degree of subjectivity from human taggers. For instance, the two following reviews are classified in the corpus as *Very Poor* and *Poor* respectively, but have been considered by different judges to entail a similar degree of negativity:

- *Only the location was a positive aspect. The room was very small, in the basement of the hotel and we could hear people walking around all night long. The shower and the toilet closets were really uncomfortable and small as well.*
- *Very near transport to London central. On arrival the apartment was very smelly of old spicy food and standard of cleanliness was poor. Entrance to apartment (hallway) needs a paint job.*

## 5 Discussion and Conclusions

Our experimental results show that our feature-driven approach to product review rating performs significantly better than previous approaches, which confirms our intuition that the different product features have different impact on the user opinion about a product. Our approach succeeds in identifying salient features which can be easily obtained from a relatively small set of product reviews and are quite independent of the reviews used to extract them. We speculate that this indicates that users are concerned about a relatively small set of product features which are also quite consistent among users. Also, the identification of salient features is carried out without previous knowledge, so the

system may be easily ported to other domains. We have also observed that the differences between the various Weka classifiers are not marked, which suggests that the proposed data representation properly captures the salient product features and the user opinions about them.

On the other hand, since we use the polarity classifier presented in Carrillo de Albornoz et al. [9] to quantify the opinion of the users about each feature, the error of this classifier increases the error of our system. To estimate the effect of error propagation, we repeat the experiments reported in Table 3, but using the sentence polarity categorization as given in the review (i.e. all sentences within the tags *<PositiveOpinion>* are considered positive and all sentences within the tags *<NegativeOpinion>* are considered negative). As a result, we improve accuracy to 84% for *Logistic*, 83% for *LibSVM* and 81.9% for *FT*. This improvement seems to indicate that a good amount of sentences are incorrectly classified. We believe the error in this classifier mainly comes from: (1) mislabeled instances in the training set (e.g. the sentence “*Anyway, everybody else was nice*” in the review shown in Table 1 is incorrectly placed in the *<NegativeOpinion>* section), (2) very frequent spelling errors in the reviews and (3) the presence of *neutral* sentences that do not express any opinion but are necessarily classified as positives or negatives. We also plan to adapt the polarity classifier to take into account the features when evaluating the polarity in order to estimate how this could affect the rating classifier.

However, most classification errors come from reviews in which it is not possible to assign any sentence to any feature and, as a result, all sentences are assigned to the *other features*. This situation occurs when the review only deals with secondary features, but especially when the method is not able to determine the focus of the sentences, so that it cannot decide which feature a sentence is talking about. For example, the review “*Dirty. Stinky. Unfriendly. Noisy.*” presents 4 sentences that are assigned to the *other features*, but should be assigned to different features if the method could identify the objects/features to which the four adjectives refer to. The same occurs in the sentence “*Anyway, everybody else was nice*” from the review in Table 1, where the system does not know that ‘everybody’ refers to the hotel staff. This problem has been previously noted by Pang et al. [4] and Turney [5] and is regarded as a kind of co-reference problem. We will address this issue in the near future.

To end with, in the long term future we plan to apply the method to deal with documents in Spanish language, using the Spanish version of WordNet available in EuroWordNet.

## Acknowledgments

This research is funded by the Spanish Government through projects TIN2009-14659-C03-01 and TSI-020100-2009-252, and the FPU program; and the Comunidad Autónoma de Madrid and the ESF through the IV PRICIT program.

## References

1. Wiebe, J.M., Bruce, R.F., O'Hara, T.P.: Development and use of a gold-standard data set for subjectivity classification. In: Proc. of ACL, pp. 246–253 (1999)
2. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proc. of ACL, pp. 271–278 (2004)
3. Kim, S.M., Hovy, E.: Determining the sentiment of opinions. In: Proc. of COLING, pp. 1367–1373 (2004)
4. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using Machine Learning techniques. In: Proc. of CoRR (2002)
5. Turney, P.D.: Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In: Proc. of ACL, pp. 417–424 (2002)
6. Esuli, A., Sebastiani, F.: Determining term subjectivity and term orientation for opinion mining. In: Proc. of EACL, pp. 193–200 (2006)
7. Brooke, J.: A semantic approach to automated text sentiment analysis. PhD thesis, Simon Fraser University (2009)
8. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity: an exploration of features for phrase-level sentiment analysis. *Computational Linguistics* 35(3) (2009)
9. Carrillo de Albornoz, J., Plaza, L., Gervás, P.: A hybrid approach to emotional sentence polarity and intensity classification. In: Proc. of 14th CoNLL, pp. 153–161 (2010)
10. Martineau, J., Finin, T.: Delta TF-IDF: An improved feature space for sentiment analysis. In: Proc. of 3rd AAAI Conference on Weblogs and Social Media (2009)
11. Carenini, G., Ng, R.T., Pauls, A.: Multi-document summarization of evaluative text. In: Proc. of EACL (2006)
12. Titov, I., McDonald, R.: A joint model of text and aspect ratings for sentiment summarization. In: Proc. of ACL/HLT, pp. 308–316 (2008)
13. Kim, S.M., Hovy, E.: Automatic identification of pro and con reasons in online reviews. In: Proc. of COLING/ACL, pp. 483–490 (2006)
14. Hu, M., Liu, B.: Mining opinion features in customer reviews. In: Proc. of AAAI (2004)
15. Plaza, L., Díaz, A., Gervás, P.: Automatic summarization of news using wordnet concept graphs. *IADIS International Journal on Computer Science and Information System V*, 45–57 (2010)
16. Lesk, M.E.: Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In: Proc. of SIGDOC (1986)
17. Yoo, I., Hu, X., Song, I.Y.: A coherent graph-based semantic clustering and summarization approach for biomedical literature and a new summarization evaluation method. *BMC Bioinformatics* 8(9) (2007)
18. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. In: Proc. of International Conference Research on Computational Linguistics (1997)
19. Lin, D.: An information-theoretic definition of similarity. In: Proc. of ICML, pp. 296–304 (1998)

# Modeling Answerer Behavior in Collaborative Question Answering Systems

Qiaoling Liu and Eugene Agichtein

Emory University, Atlanta, USA  
{qiaoling.liu,eugene}@mathcs.emory.edu

**Abstract.** A key functionality in Collaborative Question Answering (CQA) systems is the assignment of the questions from information seekers to the potential answerers. An attractive solution is to automatically recommend the questions to the potential answerers with expertise or interest in the question topic. However, previous work has largely ignored a key problem in question recommendation - namely, whether the potential answerer is likely to *accept* and *answer* the recommended questions in a *timely manner*. This paper explores the contextual factors that influence the *answerer behavior* in a large, popular CQA system, with the goal to inform the construction of question routing and recommendation systems. Specifically, we consider *when* users tend to answer questions in a large-scale CQA system, and *how* answerers tend to choose the questions to answer. Our results over a dataset of more than 1 million questions draw from a real CQA system could help develop more realistic evaluation methods for question recommendation, and inform the design of future question recommender systems.

## 1 Introduction

A key step in Collaborative Question Answering (CQA) systems is matching the posted questions to the best answerers who can contribute the needed information. In the existing systems such as Yahoo! Answers or Naver, this step is performed by the answerers manually, who choose which questions to answer based on widely varying and often subjective criteria[20]. However, this often leads to inefficiencies, redundancies, and often delayed or poor quality answers, which in turn degrades the experience for the question submitters.

An attractive solution is to automatically *recommend* the questions to the potential answerers, usually based on the expertise and/or past performance of these users for similar questions (e.g., [16,9,5,13]). However, previous work has largely ignored the key problem in question recommendation - namely, whether the potential answerer is likely to accept and answer the questions recommended to them in a timely manner. That is, even if the question is on a topic of past interest to the answerer, they may not have the opportunity or interest in answering the question *at recommendation time*.

This paper addresses this gap by exploring the contextual factors that influence the *answerer behavior* in a large, popular CQA system, with the goal to inform the construction of real-time, online question routing and recommendation

systems that also take into account the behavior of real answerers. Specifically, we consider the following research questions:

1. *When* do users tend to answer questions in a web-scale CQA system?
2. *How* do users tend to choose the questions to answer in CQA?

Our overall approach is to analyze the real answering behavior of a large group of Yahoo! Answers users, collected for more than 1 million questions over a period of one month in early 2010. Specifically, for the first research question, we analyzed both the *overall* and *user-specific* temporal activity patterns, identifying stable daily and weekly periodicities, as well as not previously observed *bursty* patterns of activity in the individual answer sessions of many users. We exploit this observation to perform a novel *session-based* analysis of the answerer activity. For the second research question, we analyze the factors that may affect the users' decisions of which questions to answer. These factors include the question category (topic), the question position in the list shown to users, and the surface patterns in the question text. We confirmed previous findings that users have "favorite" categories that attract most of their contributions, but interestingly the decisions for most of the users *within* a category are determined more by the *rank position* of the question in the list of available questions, than any other factors such as the text or the provenance of the question itself.

As far as we know, this is the first reported large-scale analysis of *answerer behavior* on session level. Our results identify new temporal patterns of contributor participation in CQA, and shed light on how the participants make minute-by-minute decisions during their online sessions. Our results could help develop more accurate answerer behavior and prediction models; allow the development of more realistic evaluation methodology for question recommendation; and inform the design of future question recommender systems.

In the next section, we overview the related work. Then, Section 3 describes our dataset in more detail, and explores the temporal patterns on *when* users tend to answer questions. Section 4 then discusses our findings of *how* the answerers tend to choose which questions to answer. Section 5 summarizes our results and concludes the paper.

## 2 Related Work

Collaborative Question Answering (CQA) systems are attracting increasing research effort in information retrieval and HCI communities (e.g., [2,14,11,20,16,7]). A key to the success of CQA systems is to provide askers with efficient and helpful service, by minimizing the time that askers need to obtain good answers for their questions. There are generally three ways to better achieve this goal. First, the large volumes of existing content in CQA systems can be reused to satisfy an asker's information need, based on effective retrieval of relevant questions and answers to the information need [14,7,6,4,22].

Secondly, an attractive approach to improve the answer quality for CQA askers is to route the new questions to experts on the topic of the question,

which has been an active area of research. For example, Jurczyk et al. [16] formulated a graph structure for CQA systems and applied a web link analysis algorithm to discover authoritative users in topical categories. Liu et al. [19] cast the expert finding problem as an IR problem, by viewing a new question as a query to retrieve the user profiles as “documents”, and tested several language models for expert ranking. Bouguessa et al. [5] focused on automatically discriminating between authoritative and non-authoritative users by modeling users’ authority scores as a mixture of gamma distributions for each topic. Beyond the CQA context, there has also been extensive work on expert finding in online forums such as [23,24].

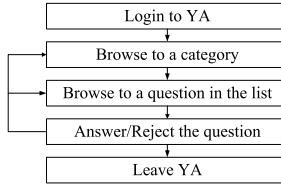
To further reduce the response time to new questions in CQA, additional question routing methods that consider user interests have been proposed [9,21,17,13,24]. For example, Guo et al. [9] developed a probabilistic generative model to obtain latent topics for questions and users, and incorporated both the topic-level and term-level information for recommending new questions to potential answerers. Qu et al. [21] applied PLSA to capture user interests in terms of topics based on their answering history, and Liu et al. [17] employed an integration of the language model and the Latent Dirichlet Allocation model for measuring the relationship between an answerer and a question, which also considered user activity and authority information. Horowitz et al. [13] addressed the question routing problem in a real-time CQA system by considering both user interest and social connectedness.

A third way to reduce the waiting time for CQA askers is simply to encourage more answerers, which depends on better understanding of the answerer behavior. Some previous work has been done on understanding user behavior in CQA [1,20,11,18,10,3]. For example, Adamic et al. [1] analyzed the content properties and user interaction patterns across different Yahoo! Answers categories. Gyöngyi et al. [11] studied several aspects of user behavior in Yahoo! Answers, including users’ activity levels, interests, and reputation. Guo et al. [10] studied the patterns of user contributions in knowledge sharing-oriented Online Social Networks including a question answering social network. Nam et al. [20] investigated the motivation of top answerers in Naver - a popular Korean CQA system, including altruism, learning, competence and points. Liu et al. [18] explored the effects of an answerer’s Web browsing context on the effectiveness of CQA systems. Aperjis et al. [3] studied the speed-accuracy tradeoff faced by CQA askers, i.e., maximizing the information accuracy while minimizing the waiting time.

As far as we know, ours is the first analysis of the session-level patterns in the answerer behavior - which could provide useful information for improving all of the question recommendation methods above.

### 3 Temporal Patterns in Answerer Behavior

This section first describes the CQA dataset used throughout this paper. It then shows the aggregate patterns of *when* answerers tend to answer questions, which confirms previous findings and validate our dataset construction. Then, we focus on the novel contribution of this paper, namely modeling the *individual* answerer



**Fig. 1.** Basic question answering process in Yahoo! Answers

**Table 1.** Dimension of the Yahoo! Answers dataset. The USER20 dataset focuses on answerers with at least 20 answers, which is used in the rest of the paper.

	Questions	Answers	Best Answers	Answerers	Askers	Users
ALL	1,056,945	4,734,589	1,056,945	433,902	466,775	726,825
USER20	933,746 (~88%)	3,319,985 (~70%)	751,633 (~71%)	45,543 (~10%)	419,395 (~90%)	437,493 (~60%)

activity within a single answer session. These analysis could help question recommender systems by suggesting *when* to begin recommending questions to a user in the first place, and *how many* questions to recommend to a user.

### 3.1 The Yahoo! Answers Setting and Data

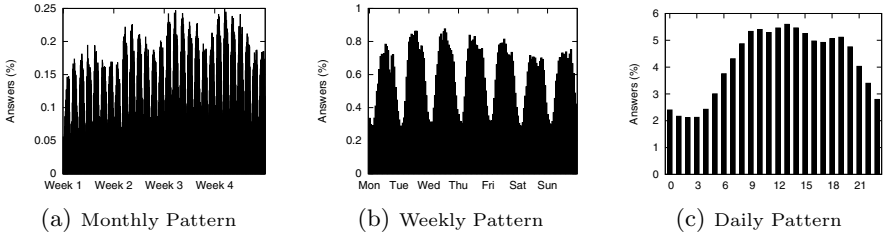
For this study we chose the Yahoo! Answers (YA) website, as a large-scale, popular, and representative example of a CQA system. To clarify our terminology and the subsequent descriptions, we briefly summarize the basic question answering model in YA, which we believe is representative of many other CQA sites. Figure 1 illustrates this process. After logging into the YA site, answerers can choose a category of interest to them to browse (including the root category “All categories”). Then they are shown a list of questions in that category among which they may answer some and skip others. This process is repeated when answerers browse to another category, until they eventually leave the site.

To construct the dataset, we crawled about 1M questions and 4.7M answers covering the top 26 categories and 1659 leaf categories in YA, as of May 2010. Since inactive users reveal less information of their answering behavior, our analysis focuses on *active answerers* who posted at least 20 answers during the period of time. This subset, called USER20, includes 45,543 answerers, accounting for about 10% of all answerers but 70% of all answers and best answers. Table 1 presents the statistics of the dataset in more detail.

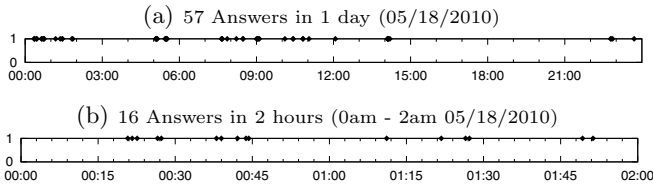
### 3.2 Aggregate Temporal Pattern Analysis

First, we analyze the overall temporal patterns of answering activities in Yahoo! Answers with the strategy used in [10]. We bin all the answers by hours, aggregate answers in the same hours by months/weeks/days, and then normalize the number of answers in each hour by the total number of answers. Based on our dataset, the answer activities in YA demonstrate strong monthly, weekly





**Fig. 2.** Temporal patterns of answer activities in YA, showing the percentage of answers in the same hours aggregated by (a) months; (b) weeks; (c) days



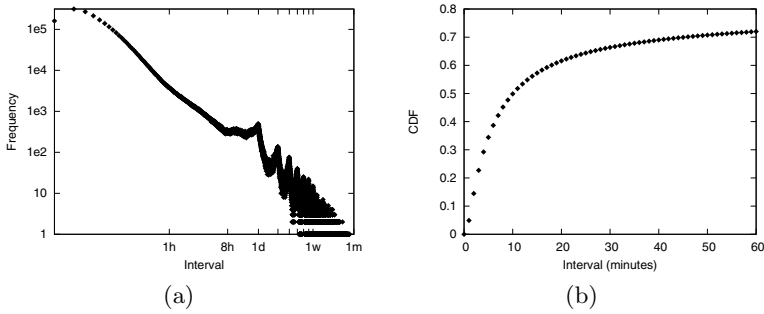
**Fig. 3.** Example answering behavior for an active user over 1 day (a) and over a period of 2 hours (b)

and daily patterns as shown in Figure 2. From Figure 2(a), we can see that the number of answers across the whole month is increasing, which indicates the growing popularity of YA. Figure 2(b) shows that the number of answers during the weekday is higher than that on the weekends, with Tuesdays and Wednesdays being the most active weekdays. Based on Figure 2(c), there tend to be three peak times in a day for answering questions, 10:00, 13:00, and 19:00 (YA server time). The least active time for answering questions is 2:00-3:00 AM. These results are similar to those described in [10], but with a time shift in the daily pattern possibly due to a different time zone used in their study.

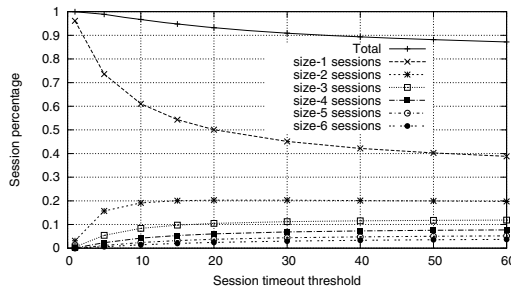
### 3.3 Burstiness of Individual Answering Activities

We now explore the temporal patterns of answering activity for individual users. We found that users tend to post bursts of answers within short answering sessions, and then “disappear” for relatively long periods of inactivity. For example, Figure 3(a) illustrates the answer activities of an example user. The user answered 57 questions that day; however, the answering was not distributed uniformly, but was concentrated in relatively short *bursts*. To provide a better intuition, we plot the user’s answering activities over a period of two hours, shown in Figure 3(b). We can see that some intervals between two successive answers are short (e.g. less than 3 minutes), but others are long (e.g. around 30 minutes), which presumably correspond to breaks between the answer sessions.

There may be two reasons for the long intervals between answers: it could be that it took the user a long time to provide the answer to a difficult question, or



**Fig. 4.** The (a)Frequency and (b)Cumulative Distribution of the intervals between two successive answers for all active users



**Fig. 5.** The change of session percentage with different session timeout thresholds

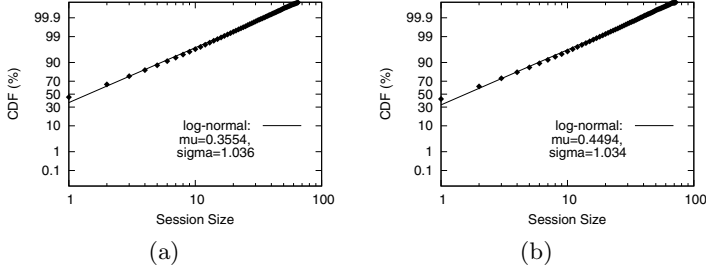
that the user left Yahoo! Answers to do something else (a more likely scenario). Therefore, we define the continuous answer activities of a user as an *answer session* of the user. Understanding the number of questions that a user would answer continuously within a single answer session would be helpful for designing question recommender systems, e.g. how many questions to recommend to a user.

To detect answering session boundaries, we adapt some of the methods proposed to determine Web search session boundaries (e.g., [8]). In our setting, the time gap between the successive answers was chosen as the most intuitive metric. We report the distribution of intervals between two successive answers of a user, shown in Figure 4(a). As we can see, the frequency of intervals less than 8 hours long, forming a roughly power-law-like distribution. However, there are seven secondary peaks, corresponding to intervals of one to seven days. We further “zoom in” to consider the intervals of one hour, shown in Figure 4(b), which shows that for over 70% of the cases, users post the next answer within 1 hour after the current answer.

Based on this observation, we apply a timeout threshold to detect the session boundaries. If the interval of two successive answers is larger than the threshold, they belong to different sessions, and to the same session otherwise. We use the methods in [12] to determine the optimal session timeout threshold, i.e., analyzing the effect of different session timeout thresholds on the proportions

**Table 2.** Answering session statistics for varying timeout values

Threshold	Session size	Session size( $\geq 2$ )	Session duration	Answer time	Gap duration
30m	2.89 $\pm$ 3.53	4.45 $\pm$ 4.16	26.5m $\pm$ 27.7m	7.68m $\pm$ 6.70m	19.1h $\pm$ 33.8h
40m	3.13 $\pm$ 3.86	4.69 $\pm$ 4.48	32.2m $\pm$ 34.7m	8.73m $\pm$ 8.44m	20.6h $\pm$ 34.8h

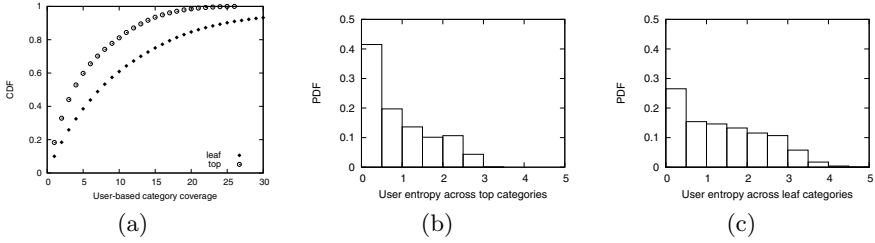


**Fig. 6.** The Cumulative Distribution (CDF) of session sizes based on the session timeout threshold of 30 minutes (a) and 40 minutes (b). The x axis is in log scale, and the y axis is in normal probability scale. Log-normal functions were fit to each CDF, with the parameters shown on the corresponding plot areas.

of sessions with different sizes. Figure 5 shows the results. The proportion of 1-size sessions decreases quickly with the increase of session timeout threshold until 30 minutes, while the proportion of sessions with size 3-6 increases. After that, increasing timeout threshold has negligible impact on proportions of these sessions, especially when the session timeout threshold is larger than 40 minutes. Therefore, the session timeout threshold should be between 30 and 40 minutes.

Table 2 shows the session statistics computed based on the two thresholds for session detection. As we can see, the average session size is around 3 for both session threshold values. This means that users answer 3 questions in a session on average, providing guidance for designing real-time question recommender systems, e.g. three or more questions can be recommended to a user. To explore the average time that users spend on posting an answer, we also compute the average session duration. A session duration is computed as the time between the posting of the first answer and the last answer in a session. For sessions with size  $n \geq 2$  and duration  $d$ , we can then compute the average answer time  $t$  as  $t = \frac{d}{n-1}$ . The results are shown in Table 2. As we can see, the average answer time appears to be about 8 minutes for both session threshold values (which also includes the time needed to choose the next question to answer).

To understand better the properties of answer session size, we show the distribution of session sizes in Figure 6. Regardless of the specific session timeout value, the distribution agrees well with the log-normal distribution, which is a line on the "normal probability" (y axis) vs log (x axis) plot. The CDF of a log-normal distribution is  $\Phi(\frac{\ln x - \mu}{\sigma})$ , where  $\Phi$  is the CDF of the standard normal distribution. The best-fit line is specified by the equation  $y = \frac{x - \mu}{\sigma}$  with the parameters reported in the plot area.



**Fig. 7.** The Cumulative Distribution (CDF) of user-based category coverage, which is the number of categories in which a user has posted answers across the entire dataset duration. The hollow circles represent user-based category coverage for top categories, and solid diamonds represent the leaf categories (a); The distribution of user entropy across all top (b) and leaf (c) categories: lower entropy indicates user activity focused on fewer categories.

In summary, the analysis above first focused on the answerer behavior in the aggregate (weekly and daily), and largely confirms previous findings, thus validating our data collection method. We then considered session-based behavior of *individual* answerers, and identified a novel *bursty* behavior of the answerers.

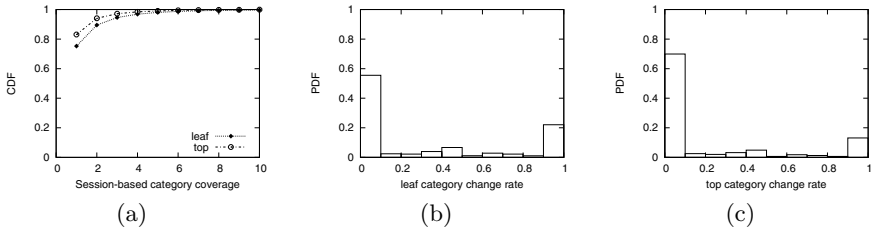
## 4 Understanding How Answerers Choose Questions

Having analyzed *when* users would like to answer questions, we now explore *how* they tend to choose the questions to answer. Based on the simplified answering process shown in Figure 11, we explore several factors that may affect the users’ decisions of which questions to answer, including question category (topic), the question’s rank in the list shown to users, question text, and the users’ previous answering history profile.

### 4.1 Question Category Effects

Browsing a category is a first step of an answering process in YA. Users can choose any category to browse, from top categories to leaf categories. If a category is not chosen explicitly, the root category “All categories” is used by default. To explore the effect of question category on users’ choices of which question to answer, we first compute the category coverage of users. The category coverage of a user is the number of different categories in which the user has posted answers. The results shown in Figure 7(a) confirm that some users answer questions in more than one leaf categories within the same top category. Moreover, we can see that more than 90% of users post answers in less than 30 leaf categories (out of 1659 leaf categories present in our dataset).

Next, we explore how focused the answers are across different categories, using the entropy measurement introduced by Adamic et al. [11]. The entropy of a user is defined as  $H = -\sum_i p_i \log(p_i)$ , where each  $i$  means a category covered by answers of the user and  $p_i$  means the percentage of answers of the user in that



**Fig. 8.** (a) The Cumulative Distribution Function of session-based category coverage, which is the number of categories in which a user has posted answers in a single answer session. The hollow circles (solid diamonds) represent session-based category coverage for top (leaf) categories. (b)(c) The Probabilistic Distribution Function of session-based category change rate for leaf(b) and top(c) categories, which is the percentage of two successive answers in different categories posted by a user in a single answer session. Note that the session timeout threshold of 30m is used here.

category. The results are shown in Figure 7(b) and 7(c). We can see that users tend to be relatively focused to answer questions primarily on a handful of topics.

For real-time question recommender systems, it is also very important to know whether a user would like to answer questions in different categories within a single session. Therefore, we also compute the session-based category coverage of users. The session-based category coverage of a user in a session is the number of different categories in which the user has posted answers in the session. The results are reported in Figure 8(a). As we can see, for around 70% of cases, the users post questions in just one leaf category in a single session.

To explore more about how users would change categories during his single answer session, we also compute the change rate of categories, shown in Figure 8(b) and 8(c). We can see that in most cases they tend not to change throughout an answer session; however, in some cases they change at every chance. Understanding the user preference on category changes in a single session can be very helpful for improving the user experience in question recommender systems.

The above analysis shows that categories play an important role in deciding users' choices of which questions to answer, as they tend to be focused rather than diverse regarding the topics. In a single session, most users prefer to answer questions in only very few categories and to answer the next question in the same category.

## 4.2 Question Rank Effects

According to the basic answering process shown in Figure 11, after choosing a category, the users will see a list of questions – by default, arranged in the order of most recent arrival. Then, the user will answer one or more questions in the list. We posit that the users tend to examine the questions in order of listing and answer them in order of the examination. This examination hypothesis has extensive support from the web search result examination literature.

Therefore, we propose the following simple, yet surprisingly accurate model of answerer behavior that simply follows the order of the posted questions.

**Ordered Question Examination Model (OQE):** *The Answerer repeatedly examines the questions in the order presented in the Category list (normally, in reverse order of arrival, most-recent first), and answers one of the top- $K$  questions in the list - and then goes back and repeats.*

To verify this OQE model, we need to know the questions and their order that the answerers saw before choosing a question and posting an answer. However, it is difficult to recreate the exact list of questions that the answerers saw, so we *approximate* the list based on the known characteristics of the YA site and the externally available data.

First, we represent an answer by a tuple  $A(u_A, q_A, c_A, t_A)$ , which means the answer  $A$  is posted by user  $u_A$  for question  $q_A$  in category  $c_A$  at time  $t_A$ . Similarly, we represent a question by a tuple  $Q(u_Q, c_Q, t_Q)$ , which means the question  $Q$  is posted by user  $u_Q$  in category  $c_Q$  at time  $t_Q$ . Then, for each answer  $A(u_A, q_A, c_A, t_A)$  of the user  $u_A$ , we create a ranked list of questions in the category  $c_A$  that are posted before the time  $t_A$ , ordered by their recentness, most recent first. More formally, the list with respect to  $A(u_A, q_A, c_A, t_A)$  can be represented as

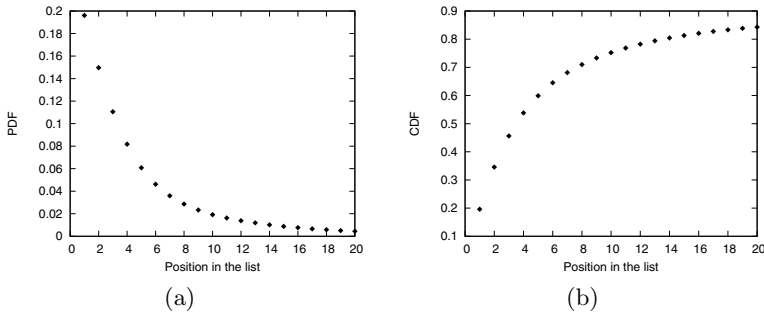
$$L_A = [Q_i(u_{Q_i}, c_{Q_i}, t_{Q_i}) \mid c_{Q_i} = c_A \wedge t_{Q_i} < t_A \wedge \forall j > i, t_{Q_j} < t_{Q_i}]$$

Note that in real scenarios, the answerers may browse any category from top to leaf. However, for simplicity, we just assume that answerers always browse to leaf categories before they answer questions. Also, we do not count in the user's time for submitting the answer  $A$  which will shift the estimated questions in  $L_A$  slightly, compared to the actual list. In addition, considering that YA shows 20 questions by default, and many answerers do not bother to click to the next page, we focus on the sublist  $L_{A,20}$  containing the top 20 questions in the estimated list  $L_A$ . Then, based on the list  $L_{A,20}$  for answer  $A(u_A, q_A, c_A, t_A)$ , we want to check whether the question  $q_A$  is in  $L_{A,20}$ . If yes, we are also interested in the rank of the hit, that is the  $i$  such that  $Q_i = q_A$ . This indicates that after the user  $u_A$  browsed to the category  $c_A$ , she chose to answer the question ranked at the  $i$ th ( $1 \leq i \leq 20$ ) position in the list shown to her.

Figure 9 shows the distribution of the rank positions of the chosen questions. As we can see, the higher a question is ranked, the greater probability it is answered. In addition, while only top 20 questions in the list are considered, the OQE model achieves a recall of 0.84. This means that for 84% of the cases, users just choose questions from the first page they see to answer.

### 4.3 Question Text Effects

Next, we try to explore beyond the question rank, to understand how the question text affects users' choices of which questions to answer. So we performed an experiment that learns to find the target question  $q_A$  in the list of  $L_{A,20}$  based on question text features. We use the learning-to-rank framework for this task. Given a list of questions  $L_{A,20}$  seen by a user, we derive features representing the



**Fig. 9.** The Probability Distribution Function(a) and Cumulative Distribution Function(b) of the positions in the list seen by a user, containing a question that was selected by the user to answer

**Table 3.** Features (50 total) used in the experiment

Position Information (4 total):
* The position where the question is in the list $L_{A,20}$ ;
* The delay of the question since it was posted until seen by the user.
* The deviation of the above 2 feature values from the average values of the user.
Similarity (5 total):
* The similarities between the question and user profile against the 4 fields and the whole profile.
Visual Quality (16 total):
* The length of question subject/content.
* The punctuation, capitalization and spacing density of question subject/content.
* The deviation of the above 8 feature values from the average values of the user.
History (4 total):
* The number of prior answers for the question seen by the user.
* The number of prior questions asked by the question asker.
* The deviation of the above 2 feature values from the average values of the user.
Keywords (21 total):
* A vector of length 20 representing whether this question contains the 20 most frequent terms in popular questions (i.e. questions with more than 20 answers).
* The number of 1s in the above vector.

associated information (e.g. question text, user’s answering history) to predict which question will be answered by the user.

Guided by reference [2], we design features according to five layers: position information about questions, question-user similarities, visual quality of questions, popular keywords in questions, and history information about the questions. The complete list of features is shown in Table 3.

To make our experiments feasible, we randomly selected 1000 out of the 45,543 active users to build the dataset for this experiment. For each user, the first half of her answers is used to build her user profile. Then, we use the next 1/4 of her answers as training data, and the last 1/4 of her answers as testing data for training and testing the ranker. The resulted dataset contains 15,226 answers and 304,434 questions for training, and 14,721 answers and 294,361 questions for testing.

We use Lucene<sup>1</sup> in our experiment to compute the similarity features between a user profile and the question. Each user profile is indexed as a document with

<sup>1</sup> <http://lucene.apache.org/>

**Table 4.** The performance of learning-to-rank approach for predicting the chosen question. A \* indicates significance at  $p < 0.05$ , and \*\* indicates significance at  $p < 0.01$ . The p-values are computed using paired t-tests (one-tailed distribution).

	P@1	Improvement over baseline
Baseline(whether the question is at position 1)	0.2445	0%
Pos(4)+Sim(5)	0.2496	+2.1%**
Pos(4)+Vis(16)	0.2438	-0.3%
Pos(4)+His(4)	0.2427	-0.8%
Pos(4)+Key(21)	0.2461	+0.6%
Pos(4)+Sim(5)+Vis(16)	0.2498	+2.1%*
Pos(4)+Sim(5)+His(4)	0.2512	+2.7%**
Pos(4)+Sim(5)+Key(21)	0.2539	+3.8%**
Pos(4)+Sim(5)+Key(21)+Vis(16)	0.2526	+3.3%**
Pos(4)+Sim(5)+Key(21)+His(4)	0.2533	+3.6%**
Pos(4)+Sim(5)+Key(21)+His(4)+Vis(16)	0.2542	+4.0%**

four fields: the content of her answers; and the title, content, and category of the questions she answered. Then we treat a question (including the title, content and category) as 5 queries against the 4 different fields as well as the whole user profile. The 5 scores returned are used as the question-user similarity features. All the features are normalized by linear scaling to unit range. After computing all the features, we then apply a learning-to-rank algorithm, SVM<sup>rank</sup> [15], to rank the questions.

Since our goal is to find the target question  $q_A$  in the list of  $L_{A,20}$ , we evaluate the results by P@1. The results are shown in Table 4. As we can see, the baseline by only checking whether the question is ranked at position 1 achieves the P@1 of 0.24. This means around one fourth of cases, users answer questions ranked top 1 in the list they see. Although position features dominate the performance, including the additional text features provides a slight, but statistically significant improvement of 4% ( $p < 0.01$ ) over the simple position-only OQE model. Therefore, while the question text does affect answerers for choosing questions, the effect of the question text is not as large as that of category and rank.

## 5 Conclusions and Future Work

This paper explored the contextual factors that influence the *answerer behavior* in a large, popular CQA system, with the goal to inform the construction of real-time, online question routing and recommendation systems. Specifically, we considered *when* users tend to answer questions in a large-scale CQA system, and *how* answerers tend to choose the questions to answer. Our analysis could help develop more realistic evaluation methods for question recommendation, and provide valuable insights into answerer behavior.

In future work, we plan to study the answerer behavior in the more proactive experiments (instead of the passive observation performed in this study), and to perform deeper investigation of the answerer behavior, e.g. by analyzing the differences in activity of the extremely active “top contributor” users.



## References

1. Adamic, L.A., Zhang, J., Bakshy, E., Ackerman, M.S.: Knowledge sharing and yahoo answers: everyone knows something. In: WWW, pp. 665–674 (2008)
2. Agichtein, E., Castillo, C., Donato, D., Gionis, A., Mishne, G.: Finding high-quality content in social media. In: WSDM, pp. 183–194 (2008)
3. Aperjis, C., Huberman, B.A., Wu, F.: Harvesting collective intelligence: Temporal behavior in yahoo answers (January 2010), <http://arxiv.org/abs/1001.2320>
4. Bian, J., Liu, Y., Agichtein, E., Zha, H.: Finding the right facts in the crowd: factoid question answering over social media. In: WWW, pp. 467–476 (2008)
5. Bouguessa, M., Dumoulin, B., Wang, S.: Identifying authoritative actors in question-answering forums: the case of yahoo! answers. In: KDD (2008)
6. Cao, X., Cong, G., Cui, B., Jensen, C.S., Zhang, C.: The use of categorization information in language models for question retrieval. In: CIKM, pp. 265–274 (2009)
7. Cao, Y., Duan, H., Lin, C.Y., Yu, Y., Hon, H.W.: Recommending questions using the mdl-based tree cut model. In: WWW, pp. 81–90 (2008)
8. Gayo-Avello, D.: A survey on session detection methods in query logs and a proposal for future evaluation. *Inf. Sci.* 179(12), 1822–1843 (2009)
9. Guo, J., Xu, S., Bao, S., Yu, Y.: Tapping on the potential of q&a community by recommending answer providers. In: CIKM, pp. 921–930 (2008)
10. Guo, L., Tan, E., Chen, S., Zhang, X., Zhao, Y.E.: Analyzing patterns of user content generation in online social networks. In: KDD, pp. 369–378 (2009)
11. Gyöngyi, Z., Koutrika, G., Pedersen, J., Garcia-Molina, H.: Questioning yahoo! answers. In: Proc. of the 1st Workshop on Question Answering on the Web (2008)
12. He, D., Göker, A.: Detecting session boundaries from web user logs. In: Proc. of the BCS-IRSG 22nd Annual Colloquium on Information Retrieval Research (2000)
13. Horowitz, D., Kamvar, S.D.: The anatomy of a large-scale social search engine. In: WWW, pp. 431–440 (2010)
14. Jeon, J., Croft, W.B., Lee, J.H.: Finding similar questions in large question and answer archives. In: CIKM, pp. 84–90 (2005)
15. Joachims, T.: Training linear svms in linear time. In: KDD, pp. 217–226 (2006)
16. Jurczyk, P., Agichtein, E.: Discovering authorities in question answer communities by using link analysis. In: CIKM, pp. 919–922 (2007)
17. Liu, M., Liu, Y., Yang, Q.: Predicting best answerers for new questions in community question answering. In: Chen, L., Tang, C., Yang, J., Gao, Y. (eds.) WAIM 2010. LNCS, vol. 6184, pp. 127–138. Springer, Heidelberg (2010)
18. Liu, Q., Liu, Y., Agichtein, E.: Exploring web browsing context for collaborative question answering. In: IIX, pp. 305–310 (2010)
19. Liu, X., Croft, W.B., Koll, M.: Finding experts in community-based question-answering services. In: CIKM, pp. 315–316 (2005)
20. Nam, K.K., Ackerman, M.S., Adamic, L.A.: Questions in, knowledge in?: a study of naver’s question answering community. In: CHI, pp. 779–788 (2009)
21. Qu, M., Qiu, G., He, X., Zhang, C., Wu, H., Bu, J., Chen, C.: Probabilistic question recommendation for question answering communities. In: WWW (2009)
22. Suryanto, M.A., Lim, E.P., Sun, A., Chiang, R.H.L.: Quality-aware collaborative question answering: methods and evaluation. In: WSDM, pp. 142–151 (2009)
23. Zhang, J., Ackerman, M.S., Adamic, L.: Expertise networks in online communities: structure and algorithms. In: WWW, pp. 221–230 (2007)
24. Zhou, Y., Cong, G., Cui, B., Jensen, C.S., Yao, J.: Routing questions to the right users in online communities. In: ICDE, pp. 700–711 (2009)

# Clash of the Typings

## Finding Controversies and Children’s Topics Within Queries

Karl Gyllstrom and Marie-Francine Moens

Katholieke Universiteit Leuven, Belgium  
{karl.gyllstrom,sien.moens}@cs.kuleuven.be

**Abstract.** The TadPolemic system identifies whether web search queries (1) are controversial in nature and/or (2) pertain to children’s topics. We are incorporating it into a children’s web search engine to assist children’s search during difficult topics, as well as to provide filtering or mitigation of bias in results when children search for contentious topics. We show through an evaluation that the system is effective at detecting kids’ topics and controversies for a broad range of topics. Though designed to assist children, we believe these methods are generalizable beyond young audiences and can be usefully applied in other contexts.

**Keywords:** controversy detection, children’s search.

## 1 Introduction

Consider the following claims about three topics in popular culture: (1) the recent *Twilight* book/film series is controversial as to whether or not it is sexist; (2) the subject of *capitalism*, though often discussed in mature news media, is likely to be interesting to many children; and (3) there is disagreement about whether *Nas*, an American rap artist, is a member of the *illuminati*. If you share the view of the authors, you will likely find these claims to be surprising, obscure, or even nonsensical. Yet, much data supports them: the query “twilight is sexist” has been issued to Google 190,000 times, “twilight is not sexist” has been issued 141,000 times, “capitalism for kids” has been issued 3,620,000 times, and “nas is/is not illuminati” 206,000 and 126,000 times, respectively<sup>1</sup>. Though they may not be obvious to us, many aspects of topics are evident in the way users seek information. In this work, we explore the use of queries to identify both controversial and children’s topics. To this end, we have developed the TadPolemic system, which identified the above examples, among many others.

There are many potential applications of this system, but we mention here a few that we plan to incorporate into a children’s search engine that we are building. First, TadPolemic can inform search engines that are designed to provide assistance or supervision to child users, who may require special treatment when searching the web. For example, if we detect that a query does not pertain

---

<sup>1</sup> As determined via the Google Suggest feature [\[2\]](#), described later.

to typical children’s topics, we may assume it to be of advanced nature (e.g., calculus), and alert the system to provide greater assistance to the child (studies have called for such support, e.g., [6]). Controversial topics, for which children are still forming personal positions, may need extra care by the system to reduce the volume of biased information, as children may be less capable than adults at filtering information and making informed judgement of the material [10]. Next, TadPolemic’s entity detection can create an extension to existing kids’ topic listings, such as that of DMOZ [1], to enable users to browse by topic, as children often prefer to do [4].

The importance of providing such assistance – and protection – to young users cannot be overstated. A 2005 study revealed the prevalence of computer use in US households: 77% of children aged 5-6 had used a computer, 42% of whom had used websites, and 22% of whom were able to browse to websites unassisted [5]. A UK study reports that a surprisingly large number of children sometimes access the web with no parental supervision: as many as 68% for children aged 5-7, and 84% for children aged 8-15 [11]. Indeed, children are capable web users, and systems should embrace young audiences rather than assume vigilant supervision. However, we stress that although our work is designed to assist children, we believe it is generalizable beyond young audiences and can be usefully applied in other contexts.

To address the detection of both kids’ entities and controversies, TadPolemic applies a simple, query-side approach in detecting the topical nature of queries. Specifically, frequently-issued queries are used as a measure of community sentiment toward various topics. As an example of the former, appending the terms “for kids” to a query (e.g., transforming “science puzzles” to “science puzzles for kids”), then determining that the new query is frequently issued, is an indication that the topic is interesting to children; on the other hand, the low frequency of a query such as “multivariate calculus for kids” indicates that the subject of “multivariate calculus” is unlikely to be appealing to children.

We specify 5 advantages of our query-side approach. (1) It is simple and easily adoptable. (2) Queries have a potentially faster time-to-discovery over a possible content-based approach that must await web articles being written. For example, a sudden news event about a novel topic may trigger a spike in queries about that topic, which would be immediately available to a query-side system, while a content-side approach would have to diligently crawl new articles within which to identify controversies. As we show later, the use of queries to detect kids’ and controversial topics may have a temporal advantage over content-based approaches. (3) Queries are potentially more reflective of the general public’s sentiments. Whereas articles, such as Wikipedia, must generally fit a neutral standard, there are no such limitations on queries; hence, they may be more revealing of users’ feelings. (4) Queries are the most compact representations of users’ information interests, unlike web pages which contain dramatically more information that must often be filtered, cleaned, or otherwise reduced. Finally, (5) systems such as Google’s and Yahoo!’s suggestion features demonstrate that the access of related queries (which we exploit) can be accomplished in real-time, allowing

TadPolemic the benefit of operating at interactive speed. In fact, this quality enables TadPolemic to be implemented as a search engine layer that responds at query time.

## 2 Kids’ Entity Detection

### 2.1 Overview

We refer to a *KidsEntity* as a subset of query terms that represent a topic of interest to children, and the objective of TadPolemic is to discover KidsEntities from queries. For example, for the query “when was Mickey Mouse created?”, TadPolemic discovers “Mickey Mouse” as a KidsEntity. For this, we use a query-based approach that applies the knowledge of the community. Specifically, we use the Google Suggestion (GS) service [2], which provides query suggestions that are based on queries that other users issue. The interaction with GS works as follows: TadPolemic sends a query to the service, which responds with a set of 0-10 queries that are recommended completions of the query, as well as the frequency of each suggested query being issued to Google’s web search. Conceptually, this serves as a coherence check for a query; if it is frequently issued, it is more likely that the query is meaningful [9]. The task of KidsEntity detection is as follows:

1. For query  $q_i$  comprising terms  $t_i \in q_i$ , find each subsequence  $s_{\langle i,k \rangle}$  of  $k$  adjacent terms from 1 to the query length.
2. For each subsequence  $s_{\langle i,k \rangle}$ , form a *KidsQuery* by appending the terms “for kids” to the subsequence’s terms.
3. This KidsQuery is issued to GS, and the suggestions (if any) are checked for the presence of the query itself (i.e., the query is a suggestion for itself).
4. The longest subsequence  $s_{\langle i,k \rangle}$  for which its KidsQuery is in the GS is used as the KidsEntity for  $q_i$ .

For example, the query “who is Mickey Mouse” would generate the subsequences “who”, “is”, “Mickey”, “Mouse”, “who is”, “is Mickey”, “Mickey Mouse”, “who is Mickey”, “is Micky Mouse”, as well as the full query itself. Of these subsequences, “Mickey Mouse” is the only one for which its KidsQuery “Mickey Mouse for kids” is in the GS suggestions.

**Related work.** Entity detection within queries has not been studied extensively. Paşca describes a method where query templates (e.g., “how much does X cost”) are discovered by processing the surrounding text from a large number of queries containing a particular term [12]. Our work adopts a similar template-based approach, though our focus is different: rather than attempt to identify templates through which similar entities can be discovered, we seek to identify binary properties – e.g., being for kids – and, in the case of controversy (described later), the nature of that property (i.e., the particular dimensions along which disagreements align).

## 2.2 Evaluation

The detection of KidsEntities should be both accurate (i.e., they actually correspond to kids’ topics) and have a high coverage (i.e., they are detected in cases where queries pertain to kids’ topics). This section is divided into the separate evaluation of these criteria, but we begin with our experimental setup. Let  $D_{kids}$  be the set of topics on DMOZ [1] within the *Kids and Teens* main category, which contains a nested hierarchy of children’s topics as well as links to child-appropriate web pages within those topics. Let  $D_{adult}$  be the set of topics on DMOZ not within the Kids and Teens category. We filtered both  $D_{kids}$  and  $D_{adult}$  to the *Science*, *Arts*, and *Society* subcategories, then collected all of the subcategories within these two sets of topics by collecting the name of any topic appearing as a directory within the hierarchy. For example, DMOZ lists *Kids\_and\_Teens/Sports* and *Kids\_and\_Teens/Sports/Basketball*, from which we would draw *Sports* and *Basketball* as topics. Conceptually,  $D_{kids}$  represents a set of topics that are more likely to be for children, or to have child-friendly facets. Conversely,  $D_{adult}$  represents a set of topics that are less likely to meet this criterion (note that this distinction does not imply being adult-oriented).

**Accuracy.** Let  $Q_{all} = D_{kids} \cup D_{adult}$ , comprising all the topics contained in either  $D_{kids}$  or  $D_{adult}$ , used as queries. Each query in  $Q_{all}$  was run through TadPolemic’s entity detection process, generating a query set  $Q_{TP} \subseteq Q_{all}$  comprising queries for which a KidsEntity exists. For each query  $q \in Q_{TP}$ , we checked the source ( $D_{kids}$ ,  $D_{adult}$ ) from which it was derived, allowing us to measure the extent to which the various sources included KidsEntities. As reported in Table 1,  $D_{kids}$  includes a substantially larger proportion of KidsEntities than  $D_{adult}$ ; this serves as a validation, in that queries determined to be for children by TadPolemic are more likely to have explicit child labels by DMOZ.

We extended upon these findings to include an assessment of the child-friendliness of pages produced by web searches using these queries. This contributes a more applied assessment, as actual web searches and pages are used. As child-friendliness labels are not generally available for the web (as they are in DMOZ), we assessed child-friendliness by using demographic information made available by the Alexa database [2]. For a given site, Alexa may list the distribution of visitors who have children, relative to the general population, on a scale of -2 to 2, with -2 being much less likely to have children than the general population, 2 being much more likely, and 0 being similar to the general population. Our belief was that child friendly sites are more likely to be visited by households with children than households without children. We confirmed this by comparing the Alexa scores between a large sample of 5899 pages from  $D_{kids}$  and 1695 pages from  $D_{adult}$  which showed the average  $D_{kids}$  score to be statistically significantly higher ( $\mu_1 = -0.17$ ,  $\mu_2 = -0.43$ , t-test p-value  $\ll 0.0001$ ).

---

<sup>2</sup> <http://alexa.com>

**Table 1.** Inclusion of  $Q_{TP}$  within various sources.  $\Delta$  indicates difference between sources’ proportions, reported as p-value of Fisher’s test.

Source	$\cap Q_{TP}$	Total	Ratio	$\Delta$
$D_{kids}$	1132	1923	0.59	
$D_{adult}$	2448	18529	0.13	$\ll 0.001$

**Table 2.** Alexa ratings.  $\Delta_{prev}$  indicates difference with previous row’s mean Alexa rating, reported as p-value of Student’s t-test.

Source	$\in$		#	KidsQuery		Regular Query	
	$Q_{TP}$	$D_{kids}$		$\bar{x}$	$\Delta_{prev}$	$\bar{x}$	$\Delta_{prev}$
$Kids_1$	Yes	Yes	1132	0.37	N/A	-0.47	N/A
$Kids_2$	Yes	No	2448	0.28	$\ll 0.01$	-0.48	0.90
$Kids_3$	No	Yes	791	-0.02	$\ll 0.01$	-0.44	0.03
$Non_-$	No	No	16081	-0.32	$\ll 0.01$	-0.61	$\ll 0.01$

For each query in  $Q_{all}$  we issued a web search<sup>3</sup>, collecting 5 results, for which we looked up the Alexa ratings where available. In addition, we executed a search for each query using its KidsQuery variant (i.e., by appending “for kids” to it). We created 4 categories of queries for this examination:  $Kids_1$  include queries from  $D_{kids}$  that were identified to have a KidsEntity by TadPolemic ( $D_{kids} \cap Q_{TP}$ );  $Kids_2$  have a KidsEntity, but were not in  $D_{kids}$ , while  $Kids_3$  include queries from  $D_{kids}$  without a KidsEntity;  $Non_-$  include queries that were neither found to have a KidsEntity, nor were in  $D_{kids}$ . This serves to compare the quality of DMOZ and TadPolemic labels. Table 2.2 depicts the average Alexa score of the top 5 search results for each query. The differences among sources in terms of Alexa ratings for KidsQueries are significant in each case. In the case of the regular (non-altered) query, the first 3 variants perform similarly. From the data we conclude that a query having a KidsEntity or appearing in  $D_{kids}$  are both strong indicators of that query’s pages’ child-friendliness ratings, and, in the case of the KidsQuery variants, having a KidsEntity is a *stronger* indicator of Alexa rating than being in  $D_{kids}$ . TadPolemic’s labels have similar quality to DMOZ, though TadPolemic can be applied to new topics, while DMOZ is limited to the categories created by the site authors. This adds evidence that the detection of KidsEntities can help orient web searches to children’s web pages.

**Coverage.** In this section, we study the use of TadPolemic on actual user queries to demonstrate how it might perform in natural contexts. For this purpose, we used the children’s query log proposed by Duarte et al. [7], to which we refer as  $Q_{AK}$ . The query log is a subset of the AOL query log<sup>4</sup> that is likely to pertain to children’s queries, based on the landing site of the query and various other properties of the search session. Though not proven that the queries are from or on behalf of children, for our purposes it suffices as a useful approximation.

For each query  $q_i \in Q_{AK}$  we used TadPolemic to identify a KidsEntity if available. Of 2332 queries, KidsEntities were detected in 2119 (90%), providing

<sup>3</sup> We report results from Google web search, though results from Yahoo! and Bing did not significantly affect our results.

<sup>4</sup> We understand that the use of this query log is controversial due to privacy issues. The subset identified by Duarte et al. was constructed with care to avoid exposing potentially sensitive personal data [7].

further evidence of a relationship between the detection of KidsEntities and the child-appropriateness of queries. We then studied the queries from the side of web search results to validate the detected KidsEntities. We had two goals: first, to determine whether the results for queries containing a KidsEntity are more child-oriented, and second, to determine whether the KidsEntities extracted from TadPolemic are semantically meaningful representations of the query. First, we examined the child-friendliness of results. For each  $q_i \in Q_{AK}$ , we ran a Google web search on  $q_i$  and drew the top 5 results, which we refer to as  $SR_i$ . For each result  $r_j \in SR_i$ , we looked up the Alexa children’s rating, and averaged the value of the 5 ratings across the  $SR_i$ . Of the 2108 queries with KidsEntities and ratings, the mean Alexa rating was 0.824, while the mean rating among the 212 queries without KidsEntities was 0.383. This difference is significant (p-value  $\ll 0.0001$  by Fisher’s test), echoes our previous findings, and reinforces the fact that KidsEntities are more likely to yield pages suitable for children.

Regarding the second goal, we examined the topical-similarity between a query and its KidsEntity to ensure that, in addition to being more suitable for children, the KidsEntity retained the semantic value of the original query (i.e., extracting the KidsEntity from a query did not substantially change that query’s meaning). For each result  $r_j \in SR_i$ , we looked up the popular *del.icio.us* tags for that page, where available, using the API call `urlinfo`<sup>5</sup>, adding the tag to the query’s tag set  $T_{SR_i}$ . If the KidsEntity  $k_{q_i}$  for  $q_i$  was in  $T_{SR_i}$ , the search result was considered relevant, and  $k_{q_i}$  was therefore considered to be a topically-relevant representation of  $q_i$ . Of 1768 queries for which a tag existed within their web results, 1273 had a matching tag (72%), indicating a strong topical overlap. Note that our use of *del.icio.us* is an approximation of relevance, as there are not human relevance assessments of the queries and web pages used in this experiment.

## 3 Controversy Detection

### 3.1 Overview

Controversy detection uses a related approach to that of the KidsEntity detection. The approach applies the frequency of what we refer to as *claim queries*, or queries of the form “X [is/are/was/were] Y”, which can provide insight into the community’s sentiments – and disagreements – on popular web search topics. Given a topic  $T_i$ , we create a set of *claim queries* by appending, individually, the verbs “is”, “are”, “was”, and “were”, to create four query variants. For example, given the topic *war* we would create the queries “war is”, “war are”, etc. These queries are then dispatched to the GS service, and the list of suggestions are examined. For each suggestion  $s_k$  in this list, we draw the terms  $t_{s_k}$  appearing after the query’s verb (e.g., from the query “global warming is real”, we pull the term “real”), and add these terms to a set of *claim terms*  $C_{T_i}$  for  $T_i$ . Next, we create a set of *negation terms*  $C'_{T_i}$ : for each term  $t_{s_k} \in C_{T_i}$ , we create its antonyms

<sup>5</sup> <http://www.delicious.com/help/json>

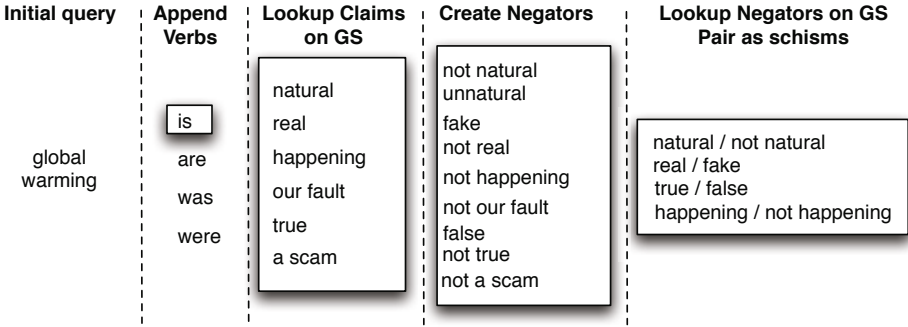


Fig. 1. Controversy lookup

by (1) Wordnet [8] lookups, and (2) creating negating terms by prepending the word “not” to the term (e.g., “real” becomes “not real”). These negating terms are used to construct *anti-queries* (e.g., “global warming is *not real*”), and we remove those not appearing within GS. For each query/anti-query pair, we create a *schism*  $Z_j$  that is composed of a source term  $t_{s_k} \in C_{T_i}$  and its negation term  $t'_{s_k} \in C'_{T_i}$ . See Figure 1 for a depiction of this process.

We treat the existence of a schism for a topic as evidence that the topic is (at least partially) controversial. Of course, controversy is a complex quality, and the nature and number of schisms should both be considered when assessing the depth of the disagreement. In fact, we later describe ways to identify highly contentious topics based on the presence of certain terms within their schisms. An important aspect of our work is not only that controversy is detected, but that we detect the particular dimensions along which disagreements align. Finally, an approach similar to the KidsEntity detection could be used for controversy detection on more complex queries (e.g., from the query “what is global warming” we could extract “global warming” as a popular entity, for which controversy detection is run).

**Related work.** The problem of controversy ranking on Wikipedia was explored by Vuong et al. as a means to identify pages of more significant dispute [13]. Their approach studies the reversal of modifications to articles – particularly when the reversals and modifications are generated by established contributors for whom such events are rare. Our approach is complementary in that we study claims made within queries, although we share interest in Wikipedia as a source of controversy and platform for testing the detection thereof.

### 3.2 Evaluation

We explore the validity of our controversy detection through the use of Wikipedia as both an explicit reference of controversial topics via labels, as well as an implicit reference via the content of articles and their discussion pages.



**Topic detection.** We created a list of topics to use as input for TadPolemic's controversy detection using popular Wikipedia topics; we selected the article titles of the 3000 most frequently viewed Wikipedia articles during an hour of August 25th, 2010<sup>6</sup>, which we refer to as  $T_{wiki}$ . Of these, TadPolemic labeled a subset  $V_{TD}$  as controversial. For comparison we used the list of controversial topics available on Wikipedia<sup>7</sup>, which we refer to as  $V_{wiki}$ <sup>8</sup>. This page contains a listing of Wikipedia articles that, for various reasons, experience a large degree of edit conflicts, and represent topics for which disagreement and controversy likely exist. We identified 384 topics in  $V_{TD}$ , 277 topics in  $V_{wiki}$ , and 100 topics in both, meaning 36% of the topics in  $V_{wiki}$  were identified by TadPolemic, while 74% of the topics discovered by TadPolemic were not in  $V_{wiki}$ .

**Comparison of topic qualities.** Though TadPolemic effectively identified many controversies contained within  $V_{wiki}$ , we further studied the topics in  $V_{wiki}$  not identified by TadPolemic and observed some limitations with these topics. First, many pertain to issues that may be obscure or otherwise unpopular. Very few of these topics had a *claim query* (only 28, or 15%), meaning that TadPolemic could not draw any useful information. We measured the relative unpopularity of these topics (as compared to those found by TadPolemic) by comparing the query frequencies for topics within  $V_{TD}$  and  $V_{wiki}$ , which are reported in Table 3 (frequency columns). The differences are quite pronounced, and each source is significantly different from the others. From this we conclude that the topics in  $V_{wiki}$  that were not identified by TadPolemic tend to be more obscure; this would make them less likely to appear as common queries, a property upon which TadPolemic relies, and therefore less likely to be issued by users.

A related problem is the timeliness of articles. For example, since TadPolemic is based on queries, it is theoretically more adaptive to novel controversies than Wikipedia, as Wikipedia must await discussion by editors. We measured this effect, with the hypothesis that topics detected by TadPolemic, yet not appearing in  $V_{wiki}$ , would be more recent. This is depicted in Table 3 (recency columns): indeed, controversies discovered by TadPolemic are more recent than topics listed by Wikipedia as controversial. We would recommend the combined use of both TadPolemic and Wikipedia to maximize coverage of controversial topics.

**Contentious issues.** We isolate certain schisms as *contentious*, or involving disagreement about topics that are polarizing or highly sensitive. We accomplish this by simply checking the schisms for the presence of a preselected set of sensitive terms, including *right*, *wrong*, *true*, *false*, *guilty*, *innocent*, *safe*, *dangerous*, *legal*, *illegal*, and *evil*. We call the set of topics containing these terms  $V_{hot}$ . We feel that this set is more important to capture, since the potential sensitivity among the audience is higher.

<sup>6</sup> Dumps of traffic are available at <http://dammit.lt/wikistats/>

<sup>7</sup> [http://en.wikipedia.org/w/index.php?title=Wikipedia:List\\_of\\_controversial\\_issues&oldid=386446018](http://en.wikipedia.org/w/index.php?title=Wikipedia:List_of_controversial_issues&oldid=386446018)

<sup>8</sup> We removed from  $V_{wiki}$  any topic that was not contained in  $T_{wiki}$ .

**Table 3.** Frequency and recency of topics from various sources.  $\Delta_{prev}$  reports the difference of means between a row and its preceding row, reported as p-value of a Student’s t-test.

Source	#	Frequency		Recency	
		$\bar{x}$ frequency	$\Delta_{prev}$	$\bar{x}$ age (days)	$\Delta_{prev}$
$V_{TD} - V_{wiki}$	284	233452813.38	N/A	2887.691	N/A
$V_{TD} \cap V_{wiki}$	100	192684800.00	0.42	3017.863	0.093
$V_{wiki} - V_{TD}$	177	60927305.08	0.0035	3249.621	$\ll 0.0001$

**Table 4.** Difference of inclusion between  $V_{hot}$  and  $V_{TD} - V_{hot}$ , reported as p-value of Fisher’s test.

Source	$S \cap V_{wiki}$	#	Ratio	$\Delta$
$V_{hot}$	38	59	0.644	$\ll 0.0001$
$V_{TD} - V_{hot}$	65	269	0.242	

**Table 5.** Proportion of topic pages containing controversial terms. Difference reported as p-value of Fisher’s test.

Source	# contr	# total	Ratio	$\Delta_{prev}$
$V_{hot}$	64	93	0.688	N/A
$V_{TD}$	217	386	0.562	0.0344
$T_{wiki}$	1418	2949	0.481	0.0029

First, we compared  $V_{hot}$  to  $V_{wiki}$ . Our hypothesis was that topics of more severe disagreement would be more likely to appear in Wikipedia’s controversial list. As depicted in Table 4, this is indeed the case:  $V_{hot}$  are substantially more likely to be included in  $V_{wiki}$ . Next, we continued with a simple, content-oriented approach. For each page in  $T_{wiki}$ , we checked the contents of the page for occurrences of the terms “controversy” or “controversial”. We compared the prevalence of pages with these terms between  $V_{hot}$  and  $V_{TD}$ , with the result depicted in Table 5. The prevalence of these controversial terms is significantly higher in topics which TadPolemic has detected to be controversial, which are significantly higher than Wikipedia pages in general (as indicated by the prevalence for  $T_{wiki}$ ).

**Schism detection.** The approach we took in this experiment was to evaluate whether the schisms identified by TadPolemic were also visible in the editor discussions on Wikipedia about the topic. We believed that a particular schism (e.g., whether global warming is real or not) would manifest in the commentary about changes to the article, as users may more aggressively revise or delete text about these issues and document the modifications. For each topic  $t_i \in V_{TD}$ , we drew the text comments of the 3000 most recent edits<sup>9</sup> and placed them into a single document for the topic, which we then indexed into a Lemur text index [3]. For each topic  $t_k \in V_{TD}$ , we selected the terms from its schisms, creating schism-set  $S_{t_k}$ . Using this schism-set, we created a query set  $Q_{merged}$ , containing a query  $q_k$  for each  $S_{t_k}$  by merging the unique terms of  $S_{t_k}$  together. Next, we

<sup>9</sup> For example, the edit-history of the *global warming* topic is [http://en.wikipedia.org/w/index.php?title=Global\\_warming&action=history](http://en.wikipedia.org/w/index.php?title=Global_warming&action=history).

**Table 6.** Success rate and search results of queries in the two scenarios. Position ratio indicates the average region in which the correct result is found; for example,  $Q_{merged}$  queries tend to appear in the first quartile of results.

Method	succeeded	failed	Success Ratio	$\bar{x}$ position (matches)	$\bar{x}$ results	Position ratio
$Q_{merged}$	269	108	0.714	27.710	112.724	0.246
$Q_{indiv}$	1172	1211	0.492	34.514	84.572	0.408

created query set  $Q_{indiv}$ , containing a query  $q_m$  for each schism  $s_m \in S_{t_k}$  composed only of the terms within the particular schism. These queries were then issued to the Lemur index<sup>10</sup>, retrieving a list of documents corresponding to the Wikipedia edit-history of the topic. Conceptually, this evaluation identifies links between schisms detected by TadPolemic and discussions regarding potential disagreement on Wikipedia. The number of queries for which the document was returned is depicted in the left columns of Table 6, revealing that 71% of merged schisms retrieved the topic from which they were derived. We also examined the positions of the correct results within these query results. Ideally, a schism should be a strong match for the discussion text of a Wikipedia article, as indicated by it appearing at a better (lower) position. Statistics about the positions of the correct result within query’s results are depicted in the 4 right columns of Table 6. We observe that the TadPolemic schisms identified for a topic are generally matched to the Wikipedia discussion topics to which they correspond. Note that this experiment is quite coarse, as it assumes that schism terms are mentioned as comments in Wikipedia discussion pages (as they often are not). However, these results show that some degree of connection exists.

## 4 Concluding Remarks

In our evaluations, we presented a large number of findings that we summarize here:

1. The percentage of queries with KidsEntities appearing in  $D_{kids}$  is high, and significantly higher than the queries appearing in  $D_{adult}$  (Table 1).
2. For a query, there is a strong connection between the existence of a KidsEntity within it and a higher child-appropriateness rating for its search results (Table 2.2). This is nearly as strong as the topic being manually labelled as child-oriented ( $D_{kids}$ ).
3. The connection in (2) is stronger than the also-positive connection between a query being in  $D_{kids}$  and its child-appropriateness rating (Table 2.2), when using the KidsQuery variant.
4. For actual queries, the connection between a query having a KidsEntity and child-appropriate ratings was strong (Section 2.2).
5. For actual queries with children’s landing pages, the prevalence of KidsEntities was high and query results tended to include pages pertaining to those KidsEntities (Section 2.2).

<sup>10</sup> Lemur was configured to use KL-divergence and default values.

6. 36% of the topics in Wikipedia’s controversy list were detected, while only 26% of the topics TadPolemic identified as controversial were in the list (Section 3.2), and contentious topics detected by TadPolemic were even more likely to be listed on Wikipedia’s list (Table 4).
7. The topics detected by TadPolemic were more popular and newer than those on the list that were not detected (Table 3); those that were not had low (15%) incidence of claim queries.
8. The Wikipedia pages for topics determined by TadPolemic to be controversial were more likely to contain terms like “controversy”, and this was more pronounced with topics identified to be contentious by TadPolemic (Table 5).
9. The schisms identified by TadPolemic were correlated with the discussion pages of their Wikipedia topics (Table 6).

We connect these findings into the following conclusions: the KidsEntities detected by TadPolemic are accurate (1-3) and have a broad coverage in actual web queries (4-5). The controversies detected by TadPolemic are accurate (6, 8), and potentially more broad, popular, and timely than those specified by Wikipedia’s controversial list (7). Evidence suggests that the schisms detected are accurate (9).

Though TadPolemic is still in a formative stage, we believe much of the technology could be simply implemented in a usable system. GS is designed to execute at interactive rates, specifically to offer suggestions as the user types a query. In terms of requests per user query to the service, our system generates a volume that is comparable to the volume that a typical Google search user would generate. For example, the KidsEntity extraction on the query “who is Mickey Mouse” would generate 10 requests to GS (each subsequence of 1 to 4 terms), while a user typing the query on the Google website would generate 19 requests (once per character entered). The controversy detection requires a similarly manageable number of requests. In this respect, an implementation of TadPolemic in an online search engine could be as simple as a thin layer between the user and a web search engine. The simplicity of our approach – a single URL call to Google – is an asset in this regard.

**Limitations.** Due to the inherent difficulty of evaluating kids’ entities and controversies, we used some comparison data sets that are imperfect. We emphasize that the use of the Wikipedia controversy list is a very coarse approximation of a gold standard. The inclusion of pages within this list is subject to the presence of “edit-wars”, which are unlikely to occur for the vast majority of topics for which some disagreement exists. Furthermore, inclusion on this page is a ephemeral matter, and topics may enter and exit as disagreements are mediated via Wikipedia’s community. Similarly, the use of Wikipedia articles containing the term “controversy” is also coarse; the presence or absence of the term “controversy” is an extremely simple test, and subject to the peculiarities of Wikipedia’s structure (e.g., many large topics are distributed among many linked articles, only one of which may contain discussion of controversies pertaining to the topic). Our approach also assumes that the particular schisms will be mentioned directly within the comments, though in practice this is not nearly comprehensive. Still, the connections we identified are a sign that there is reasonable overlap. Finally,

the use of *del.icio.us* tags as relevance assessment suffers the limitation of sparsity in number and variety of tags; though it has the advantage of providing easy, fast, and cheap relevance information. Despite the flaws of these comparisons, we believe that our results characterize a system that is effectively performing the function that we intended it to perform.

On the other hand, we perceived the use of human assessment to also be sensitive to errors. The reason is that (1) controversy is a largely subjective matter, and (2) our system identified controversy in an extremely large number of topics, many of which we were not initially familiar with (though brief research confirmed their existence), and included schisms ranging from obscure (whether *oxygen* is flammable, and whether *Nas* (American rapper) is a member of the illuminati) to the very recent (whether *twitter* is useful, whether the films *Toy Story 3* and *The Last Airbender* were good or bad). Nonetheless, we consider human evaluation to be an essential future direction of our work.

## References

1. ODP – Open Directory Project, <http://www.dmoz.org/>
2. Query Suggest FAQ, <http://labs.google.com/intl/en/suggestfaq.html>
3. The Lemur Project (2001-2008), <http://www.lemurproject.org>
4. Bilal, D.: Draw and tell: Children as designers of web interfaces. *ASIST* 40(1), 135–141 (2003)
5. Calvert, S.L., Rideout, V.J., Woolard, J.L., Barr, R.F., Strouse, G.A.: Age, Ethnicity, and Socioeconomic Patterns in Early Computer Use. *American Behavioral Scientist* 48(5), 590–607 (2005)
6. Druin, A., Foss, E., Hutchinson, H., Golub, E., Hatley, L.: Children's roles using keyword search interfaces at home. In: *CHI 2010*, pp. 413–422. ACM, New York (2010)
7. Duarte Torres, S., Hiemstra, D., Serdyukov, P.: An analysis of queries intended to search information for children. In: *IiX 2010*, pp. 235–244. ACM, New York (2010)
8. Fellbaum, C.: *WordNet: An Electronic Lexical Database*. Bradford Books (1998)
9. Gyllstrom, K., Moens, M.-F.: A picture is worth a thousand search results: finding child-oriented multimedia results with collAge. In: *SIGIR 2010*, pp. 731–732. ACM, New York (2010)
10. Hirsh, S.G.: Children's relevance criteria and information seeking on electronic resources. *J. Am. Soc. Inf. Sci.* 50(14), 1265–1283 (1999)
11. Ofcom. UK children's media literacy (March 2010), [http://stakeholders.ofcom.org.uk/market-data-research/media-literacy/medlitpub/medlitpubrssi/uk\\_childrens\\_ml/](http://stakeholders.ofcom.org.uk/market-data-research/media-literacy/medlitpub/medlitpubrssi/uk_childrens_ml/)
12. Paşca, M.: Weakly-supervised discovery of named entities using web search queries. In: *CIKM 2007*, pp. 683–690. ACM, New York (2007)
13. Vuong, B.-Q., Lim, E.-P., Sun, A., Le, M.-T., Lauw, H.W.: On ranking controversies in wikipedia: models and evaluation. In: *WSDM 2008*, pp. 171–182. ACM, New York (2008)

# Are Semantically Related Links More Effective for Retrieval?

Marijn Koolen<sup>1</sup> and Jaap Kamps<sup>1,2</sup>

<sup>1</sup> Archives and Information Studies, University of Amsterdam, The Netherlands

<sup>2</sup> ISLA, University of Amsterdam, The Netherlands

**Abstract.** Why do links work? Link-based ranking algorithms are based on the often implicit assumption that linked documents are semantically related to each other, and that link information is therefore useful for retrieval. Although the benefits of link information are well researched, this underlying assumption on why link evidence works remains untested, and the main aim of this paper is to do exactly that. Specifically, we use Wikipedia because it has a dense link structure in combination with a large category structure, which allows for an independent measurement of the semantic relatedness of linked documents. Our main findings are that: 1) global, query-independent link evidence, is not affected by the semantic nature of the links, and 2) for local, query-dependent link evidence, the effectiveness of links *increases* as their semantic distance *decreases*. That is, we directly observe that links between semantically related pages are more effective for ad hoc retrieval than links between unrelated ones. These findings confirm and quantify the underlying assumption of existing link-based methods, which sheds further light on our understanding of the nature of link evidence. Such deeper understanding is instrumental for the development of novel link-based methods.

**Keywords:** Links, Semantic Relatedness, Effectiveness, Wikipedia.

## 1 Introduction

Link-based ranking algorithms such as spreading activation [2], relevance propagation [19], HITS [8] and SALSA [12], use the assumption that linked documents have related content. For example, Picard and Savoy [16] say “the implicit reasoning made in spreading activation (SA) technique is the following: a link from a document  $d_1$  to a document  $d_2$  is *evidence* that their content is similar or related, such that if  $d_1$  is relevant to a given request,  $d_2$  may also be relevant.”

So far, this assumption has remained implicit, because it is hard to measure the semantic relatedness of linked documents independent from the feedback of a retrieval system given a search query. Wikipedia allows us to explicitly measure the semantic relatedness of linked documents independently, and, with the INEX Wikipedia Ad Hoc test-collections since 2006, also allows us to study its impact on the effectiveness of link evidence for retrieval. Kamps and Koolen [7] found that Wikipedia links behave very much like links on the larger Web. Wikipedia being a part of the Web, we expect our findings to be generally applicable.

The algorithms mentioned so far are all query-dependent methods, but there are also query-independent methods, which might rely less on the semantic nature of links. In our analysis, we make a distinction between algorithms that use global, query-independent evidence, such as PageRank [15], and local, query-dependent algorithms such as HITS, which use the links between a subset of documents retrieved for a given topic in a query-dependent way. In this paper, we use the term local link evidence to refer to the links between the top 100 results for a given query.

Najork [14] showed that for large-scale Web retrieval, SALSA is more effective than PageRank, indicating that local link evidence is more effective for retrieval than global link evidence. A similar observation was made by Kamps and Koolen [7], who compared the effectiveness of global and local link degrees on Wikipedia ad hoc retrieval. These findings suggest that local link evidence reflects not only document importance, but also topical relevance [9]. Note that the query-dependent set of links is a proper subset of the global link structure. Evidently, some, but not all, links are useful for retrieval. This confronts us with the question:

- Are links between semantically related documents more effective for ad hoc retrieval than links between unrelated ones?

Wikipedia has a complex category structure, providing us with a hierarchical semantic classification of the articles. Thus, we can see whether a link connects two documents in the same category—in which case there is a clear semantic aspect to the link—or between two documents belonging to very different categories. We can also use category hierarchy to measure the semantic distance between two documents, which gives us a more fine-grained measure of semantic relatedness.

By filtering links based on their distance—removing the longest semantic distance links or the shortest semantic distance links—we can study the impact of the semantic nature of links on the effectiveness of link evidence. But filtering not only changes the semantic nature of link evidence but also the quantity. We therefore compare semantic filtering of links against a random filter. This leads to the more specific research questions:

- How is the link structure related to the categorical organisation in Wikipedia?
- How does semantic filtering of links affect the impact of link evidence on retrieval?

The rest of this paper is organised as follows. We first discuss related work in Section 2, and describe the category structure and look at the semantic relatedness of documents in Section 3. In Section 4 we address the issue of measuring semantic relatedness using the Wikipedia category structure. We then analyse how linked documents in the global and local link graph are distributed over the semantic relatedness measure in Section 5. Then, in Section 6 we describe experiments with filtering links using the category structure and finish with conclusions in Section 7.

## 2 Related Work

Link-based ranking algorithms like spreading activation [2], relevance propagation [19] and HITS [8] all use the implicit assumption that linked documents tend to be related to each other and therefore, that link information is potentially useful for retrieval. Consider the expansion step of the HITS algorithm: Starting with the highest ranked results for a query, this set is expanded by pages connected to those results to make sure the most important authorities on the search topic are included in the expanded set. The first part of this assumption was confirmed by [4], who showed that links on the Web tend to connect pages with topically related content.

The benefits of link information for information retrieval have been well-researched. A recent, large-scale evaluation of well-known algorithms such as PageRank [15], HITS and SALSA [12] was conducted by Najork [14]. On a large Web crawl and some 28,000 queries, he found that any link-based algorithm, including simple in-degree counts clearly outperform a random ordering of the same results. Link information is useful for ranking documents. However, these results do not explain why link information is useful.

Kurland and Lee [10, 11] showed that generating links based on document similarity can help improve ad hoc retrieval effectiveness. Assuming that document similarity reflects some kind of semantical relation between documents, this result shows that links between semantically related documents are effective for retrieval. However, it does not show they are effective *because* they connect semantically related documents.

Measuring the semantic relatedness (SR) of documents can be done in many different ways. A good overview of SR methods can be found in [1]. For our purposes, the work of Strube and Ponzetto [20] is relevant, because they used the Wikipedia category and link structures to measure word relatedness, and found that path-based measures using the category hierarchy perform well. The effectiveness of simple path-based measures on the Wikipedia category structure for SR is supported by Zesch and Gurevych [21].

## 3 Wikipedia Category Structure

We use the INEX 2006 Wikipedia collection [5], consisting of over 650,000 documents. The Wikipedia category structure is more or less hierarchical—categories are linked to each other via hypernym/hyponym relations but can have multiple parent categories—and allows us to determine how semantically related two documents are, even when they are not assigned to the same category, based on distances between categories.

In Wikipedia, anyone can edit the category structure, and there is no standard way to create such taxonomies of categories: one person could introduce several intermediate levels between two categories where another would introduce none or only a few. Some of the relations are even cyclic in the sense that two categories can subsume each other. However, we assume that distances at the extreme



**Table 1.** Link degree and category size statistics of the Wikipedia collections

	Description	min	max	mean	median	stdev
<i>Category</i>	# articles	0	4,534	16.82	4	56.87
	# children	0	1,581	1.69	0	8.55
	# parents	0	55	1.69	2	1.17
	distance	1	23	7.29	7	1.58
<i>Article</i>	# categories	1	41	2.20	2	1.64

ends of the distribution—the shortest and longest distances—can respectively be interpreted as semantically related and unrelated.

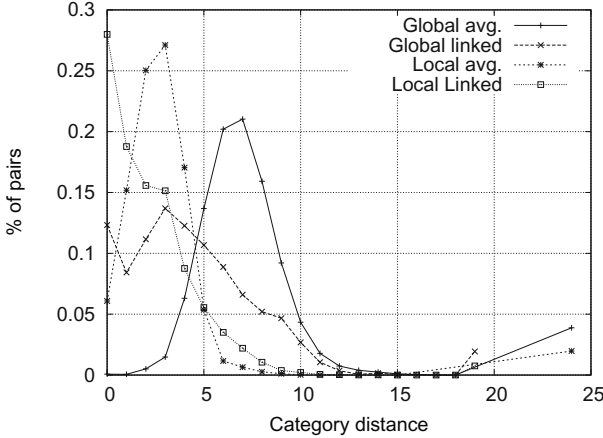
Some statistics on the category structure are given in Table 1. The category structure of the INEX 2006 Wikipedia collection contains 86,024 distinct categories. The top category in the hierarchy is called CATEGORIES, and almost all categories are connected to this top category via sub-category relations. There are 75,601 categories that contain articles and 10,423 categories that contain no articles but have only sub-categories. The mean number of articles per category is 16.82, but the median is lower (4), showing that the distribution is skewed. The mean number of parent and child categories is 1.69, but the median numbers of parent and child categories are 2 and 0 respectively. Thus, most categories are leaves in the category structure, connected to at least 2 broader categories. All articles in the collection are assigned to at least one category, with a mean (median) of 2.2 (2).

## 4 Measuring Semantic Relatedness

Given that Wikipedia is a collection of interrelated topics, we can view the category structure as a taxonomy of concepts and use methods from computational linguistics to measure SR. The easiest way is to make a distinction between a pair of documents belonging to the same category and a pair of documents belonging to different categories, and say that the former pair is *semantically similar* whereas the latter pair is not. To give insight into how links in Wikipedia are related to the category structure, we adopt a path-based measure that simply counts the number of edges along the shortest path between two concept nodes [17, 18]. The rationale behind this is that “the shorter the path from one node to another, the more similar they are” [18] and “the relatedness of two words is equal to that of the most related pair of concepts they denote” [1].

We opt for the path-based measure using the category hierarchy because it is simple, has proven to be reasonably effective in semantic relatedness evaluations [20, 21], and, as we will see in the next sections, is sufficient for our purpose of studying the impact of SR on the effectiveness of link evidence for retrieval. The category distance between two documents  $d_a$  and  $d_b$  is the minimum of the category distances between the categories of  $d_a$  and  $d_b$ :

$$dist_{\text{cat}}(d_a, d_b) = \min_{c_i \ni d_a, c_j \ni d_b} dist_{\text{cat}}(c_i, c_j)$$



**Fig. 1.** Distribution of category distances between documents

where  $c_i \ni d_a$  are the categories to which  $d_a$  is assigned. The distance  $dist_{cat}(c_i, c_j)$  between the two categories  $c_i$  and  $c_j$  is defined as:

$$dist_{cat}(c_i, c_j) = dist_{cat}(c_i, lso(c_i, c_j)) + dist_{cat}(c_j, lso(c_i, c_j)) \quad (1)$$

where  $c_i$  and  $c_j$  are two categories,  $lso(c_i, c_j)$  is the lowest super-ordinate (the lowest super category) of  $c_i$  and  $c_j$  and  $dist_{cat}(c_i, lso(c_i, c_j))$  is the number of steps up the hierarchy from category  $c_i$  to  $lso(c_i, c_j)$ .

When we consider only categories connected to the top category CATEGORIES, the average shortest distance between two categories is 7.29 (median 7) and the maximum is 23. What is the average category distance between two pages? We randomly sampled one million pairs of documents and computed the shortest category distance between them. The distribution of category distances is shown in Figure 1 (the solid line, Global average). The distribution of the global pairs is roughly normally distributed, with a peak at distance 7, with 21% of the documents pairs. The bulk of the document pairs are at a category distance of 4–10, and very few document pairs are semantically close to each other. The right-most data points represent document pairs belonging to unconnected parts of the category structure. Among the pairs that are connected via the category structure the average distance is 6.61, which is slightly below the average distance between two categories, which is 7.29. We also computed the category distance between all document pairs in the local top 100 documents for the 221 topics. This resulted in 1,093,950 document pairs. The distribution has roughly the same shape but is shifted towards the smaller distances. In the top 100 retrieved documents, 6% of the document pairs share at least 1 category (distance 0), the most frequent distance is 3 and almost all pairs have a distance less than 6. Among the pairs that are connected via the category structure, the average distance is 2.56. The documents in the top 100 results are more semantically related to each other than in the overall collection.

## 5 Links and Categories

Now that we have chosen a method to measure semantic relatedness, we look at how the link structure is related to semantic relatedness. Again, one of the main assumptions underlying algorithms like HITS and relevance propagation [19] is that links are a signal that two documents are topically related to each other. But perhaps not *all* linked documents are topically related to each other. The Wikipedia category structure provides a manually created semantic organisation of the Wikipedia articles, with which we can quantify how related two articles are. How is the link structure related to the categorical organisation in Wikipedia? We look at the shortest category distance between linked articles. The distribution of links over shortest category distance is given in Figure 11 and is shown both globally and locally over the top 100 retrieved results. The local top 100 results are based on the 221 Ad Hoc topics and associated relevance judgements of the INEX Ad Hoc test-collections of 2006–2007 [6, 13]. The baseline retrieval system is described in the next section.

In the global link structure, around 12% of the links connect two articles sharing at least one category—from here on referred to as *within-category links*, as opposed to *cross-category links*, which connect documents that share not a single category. The most frequent distance is 3 steps, above which the frequency gradually drops to almost 0 at 12 steps. There is a small peak again at the end, for the links between articles assigned to unconnected categories.

Linked documents tend to be more semantically related to each other than randomly paired documents and share a category much more often. The median category distance of the linked documents is 4 while the median of the randomly paired documents is 7. Among the linked documents that are connected via the category structure, the average distance is 4.04, compared to 6.60 for the randomly sampled pairs. *There is a clear relation between global links and semantic relatedness.* However, compared to the documents in the top 100, the linked documents share a category more often but are also more frequently separated by greater semantic distances. Within the top retrieved results, the global link evidence has a weaker semantic signal than the text evidence.

The category distance distribution over the local links is based on 63,435 links between the documents in the top 100 results of the 221 topics (5.8% of all possible pairs in the local sets). The local links show a very different distribution. Here, the 0 distance links are the most frequent and make up more than 25% of the link set, and the frequency drops monotonously over category distance, with almost no pairs beyond 8 steps. There is a small set of links between articles assigned to unconnected categories. This means there is a clear relation between local link evidence and SR. In the query-dependent link set we more frequently find links between articles that are semantically similar. This is not surprising, because each article appears in the local set because it shows similarity with the search query and therefore also with the other documents in the local set. However, the average distance of the linked document pairs is 2.22 while over the entire local set the average is 2.56. In the top 100 results of a given query,

the local links provide a stronger signal that two documents are semantically related than the text evidence.

How is the link structure related to the category structure? There is a clear relation between global links and SR. However, this semantic signal is weaker than the text evidence in the top retrieved documents. In the local set, pages that are linked tend to be more semantically related than pages that are not linked. Is the semantic nature of links also related to their effectiveness for information retrieval? This question is addressed in the next section.

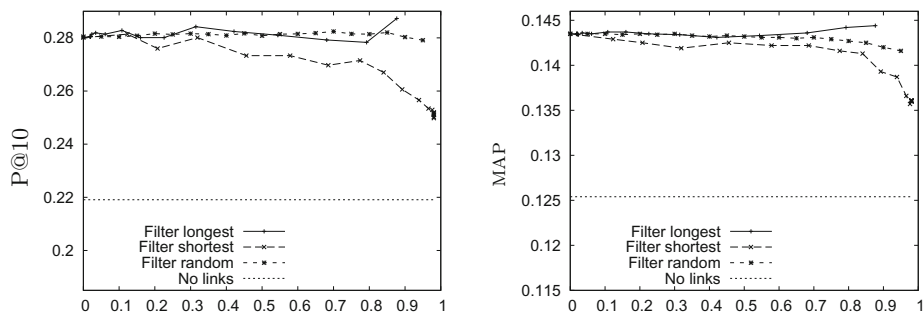
## 6 Semantic Relatedness and Effectiveness of Links

By zooming in on the top ranked retrieval results, we filter the link graph on the search topic and end up with links between semantically related pages. The global link graph contains the same links but also many more links between semantically unrelated pages. How is the impact of link evidence related to the semantic nature of links? We use the category structure to filter links and thereby control the semantic nature of link evidence. What happens to the impact of link evidence if we remove the within-category links? Does link evidence become less effective? What happens when we remove only the longest distance links?

Our baseline run is a standard language model run with linear smoothing ( $\lambda = 0.15$ ) and a document length prior  $P_{length}(d) = |d|/\sum_{d' \in D} |d'|$ , where  $d$  and  $d'$  are documents in collection  $D$ . The length prior promotes longer documents and improves MAP from 0.2561 to 0.3157.

To study the effectiveness of link evidence, we look at link degrees, which have proven to be very competitive compared to more complex algorithms like PageRank and HITS [7, 14], and are simpler to compute. As in [7], we concentrate on the top 100 results for each topic. We experimented with in-degrees, out-degrees and their union (treating links as undirected), and found that for global link evidence, the in-degree is more effective than out-degree or their union. For local link evidence, in- and out-degree are equally effective, but their union is more effective. As global, query-independent evidence, links are more effective in one direction, which suggests they provide evidence of document importance. As local, query-dependent evidence, links are effective in both directions, suggesting their evidence is symmetric, and might reflect semantic relatedness, which is symmetric as well. For lack of space, we restrict our discussion to the global in-degrees and the local union degrees.

To show how the semantic nature of links affects their impact on effectiveness, we use two filtering methods: one where we remove the shortest semantic distance links (the SD filter), effectively degrading the semantic nature of the link graph, and one where we remove the longest semantic distance links (LD filter), effectively improving the semantic nature of the link graph. We filter links based on the path length distance measure described above. In the first filtering step the SD filter removes the links at distance 0, in the second step the links at distance 1, etc. The LD filter first removes the links between pages unconnected to each other via the category structure. In the second step the LD filter removes links at the largest distance (18 steps, see Figure 1), etc.



**Fig. 2.** The impact of filtering links on the effectiveness of ranking the top 100 results by global in-degree. The x-axis shows the percentage of links removed.

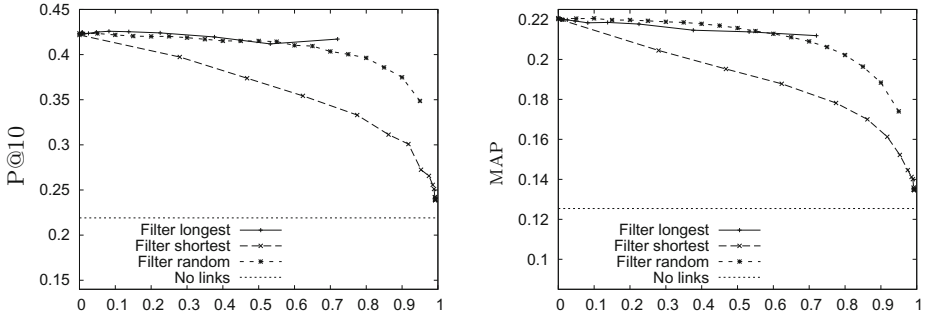
Note that by filtering we not only affect the semantic nature of the link graph, but also the link quantity. For comparison, we also look at the impact of randomly filtering links. We do this by assigning a random value between 0 and 1 to each page in the collection and sampling  $n\%$  of the pages by selecting all pages with a value below  $\frac{n}{100}$ . The degree distribution of an  $n\%$  sample is determined by the random assignment of the values, so repeating the experiment can result in different distributions. Therefore, the values reported are the averages over 20 iterations. If we randomly remove links from the graph, we would expect that the degrees change uniformly. That is, all pages are affected in the same way.

We look at the top 100 results retrieved by the text retrieval baseline and compare the ranking based on link evidence against a random ordering of documents. This shows whether link evidence has any potential value. The impact of filtering on the effectiveness of the global in-degrees is shown in Figure 2. The x-axis shows the percentage of links filtered.

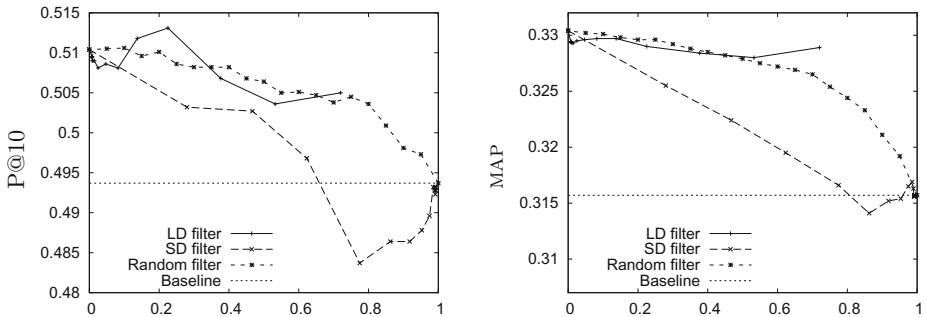
The left figure shows the impact on P@10. The Random and LD filters have little impact on the in-degrees, but removing the shortest distance links hurts performance. Performance stays well above that of random ordering though. On MAP (right figure) the impact of filtering is similar to the impact on P@10. The in-degree performance slightly improves with only the within-category links and drops with only the 10% longest distance links. From these observations we learn that filtering has little impact on the global degrees, probably because the link graph is very rich and the high-degree pages are very robust against filtering.

Although filtering does not improve performance for the in-degrees, we note that using global out-degrees (not shown) is not effective—no better than random—unless we filter out the longest distance links. A large number of links to semantically related pages signals that a page is a good hub for a particular topic.

The impact of filtering on the local degrees is shown in Figure 3. Note that without filtering, local links are far more effective than global links. Here, random filtering has a bigger impact. The local link graph is already filtered on the search topic and has far fewer links. Further filtering flattens the degree distribution even more. If we remove the shortest distance links first, performance drops faster



**Fig. 3.** The impact of filtering links on the effectiveness of ranking on local union degrees. The x-axis shows the percentage of links removed.



**Fig. 4.** The impact of filtering links on the effectiveness of ranking on local union degree and text evidence. The x-axis shows the percentage of links removed.

than with random filtering, while if we remove the longest distance links first, performance remains stable. The shortest semantic distance links are the more effective links. If we want to improve ad hoc search by exploiting link evidence, we need links between semantically related pages. Another important thing to note is that filtering on the category structure does not make local link evidence more effective. Zooming in on the highest ranked retrieval results already gets rid of most links between unrelated pages. Further filtering is not needed.

What happens to the performance of link evidence in combination with the content-based score when we filter links? There are many ways to combine content and link evidence (see, e.g. [3]). We experimented with several combination methods, such as using the log of the degrees or prior probabilities trained on the relevance data, instead of the degrees themselves. Although the impact of filtering is similar for the different methods, we found that the most effective combination is to multiply the document score from the baseline model by the local union degree plus 1 (so that documents with no links keep their original document score). That is, the final score is  $S(d) = S_{base}(d) \cdot (1 + \text{degree})$ , where  $S_{base}(d)$  is the baseline score. The results are given in Figure 4. The baseline

scores are the straight dotted lines. The local union degrees improve upon the baseline performance. With random filtering, both P@10 and MAP gradually drop as we remove more links. If we remove the SD links first, the improvement drops faster and the score even falls below that of the baseline. With the LD filter, the P@10 score fluctuates somewhat between 0.505 and 0.513, while the MAP score remains stable. With just the local within-category links, the improvement is the same as with all local links. Again, the links between the most semantically related documents are the most effective.

Note that filtering does not improve the effectiveness of local link evidence, which might be explained by the fact that the local link graph is already filtered on the search topic, which is a semantic filter in itself.

To summarise, global link evidence is very robust against filtering and its effectiveness seems unrelated to semantic relatedness. Local link evidence is more sensitive to filtering, partly because the graph is more sparse as is it already filtered on the search topic. But its effectiveness is directly related to the semantically relatedness of the linked documents.

## 7 Conclusions

This paper investigated the semantic nature of links, trying to answer whether links between semantically related pages are more effective for retrieval than links between unrelated ones. Our first research question was:

- How is the link structure related to the categorical organisation in Wikipedia?

Compared to a random sample of document pairs, linked documents tend to be more semantically related to each other and more often share a category, showing a clear relation between global links and semantic relatedness. However, within the top retrieved documents for a given query, the semantic signal of global link evidence is weaker than that of the textual evidence, providing an explanation why global link evidence is almost ineffective for topic relevance tasks. In the local set, pages that are linked tend to be more semantically related than pages that are not linked. Local link evidence is more clearly related to semantic relatedness and, even in the more topically focused set of top retrieved pages, links are a stronger signal that two pages are semantically related. This shows a difference in the semantic nature of global and local links. The semantic nature of link evidence changes as we zoom in on a subset of pages retrieved for a given query. Our second research question was:

- How does semantic filtering of links affect the impact of link evidence on retrieval?

Global incoming link evidence is robust against filtering links randomly or based on semantic distance, and only becomes less effective when the longest semantic distance links are left. The effectiveness of global link evidence is not determined by the semantic relatedness of linked documents. Local link evidence is

less robust against filtering, becoming less effective when we remove links. Effectiveness drops as the number of short distance links drops. The effectiveness of local link evidence is thus, at least partly, determined by the semantic relatedness of linked documents. The step from a global link graph to a local link graph works as a semantic link filter. Many of the links between semantically unrelated pages are removed. This is an essential step in making link evidence useful for ad hoc search. Our hypothesis that link evidence for topical relevance is symmetric hinges on the semantic relatedness of linked pages.

Finally, our main aim was to investigate:

- Are links between semantically related documents more effective for ad hoc retrieval than links between unrelated ones?

When the aim of link evidence is to identify important documents, links between semantically related documents are not more effective than links between unrelated ones. When we make link evidence sensitive to the context of the search topic, the role of link evidence shifts to identifying topically relevant documents, and here links between semantically related documents are indeed more effective than links between unrelated ones.

More generally, our findings confirm the assumption that (query-dependent) link information is effective for retrieval because it signals the semantic relatedness of linked documents. This adds to our understanding of why link evidence works, which can help in developing better link-based ranking methods.

We did this analysis in Wikipedia because its category structure allows an independent measurement of the semantic relatedness of linked documents. The fact that the impact of link evidence in our experiments is similar to the results of other studies (e.g. local versus global evidence, [14]), and that Wikipedia links behave like general Web links [7], offers support that these findings generalise to the larger Web and hyperlinks in general.

In future work, we will extend our analysis to a general Web corpus and experiment with generating links based on content similarity (as done by Kurland and Lee [11]). We will also investigate better ways of combining link and text evidence, and look at the impact of weighting instead of filtering links.

**Acknowledgments.** This work was generously supported by the Netherlands Organization for Scientific Research (NWO, grants # 612.066.513, 639.072.601, and 640.001.501).

## References

- [1] Budanitsky, A., Hirst, G.: Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics* 32(1), 13–47 (2006)
- [2] Cohen, P.R., Kjeldsen, R.: Information retrieval by constrained spreading activation in semantic networks. *Inf. Process. Manage.* 23(4), 255–268 (1987)
- [3] Craswell, N., Robertson, S., Zaragoza, H., Taylor, M.: Relevance weighting for query independent evidence. In: *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 416–423. ACM, New York (2005) ISBN 1-59593-034-5



- [4] Davison, B.D.: Topical locality in the web. In: Research and Development in Information Retrieval (SIGIR), pp. 272–279 (2000)
- [5] Denoyer, L., Gallinari, P.: The Wikipedia XML Corpus. SIGIR Forum. 40(1), 64–69 (2006)
- [6] Fuhr, N., Kamps, J., Lalmas, M., Malik, S., Trotman, A.: Overview of the INEX 2007 ad hoc track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) INEX 2007. LNCS, vol. 4862, pp. 1–23. Springer, Heidelberg (2008)
- [7] Kamps, J., Koolen, M.: Is Wikipedia link structure different? In: Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM 2009), pp. 232–241. ACM Press, New York (2009)
- [8] Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. *Journal of the ACM* 46(5), 604–632 (1999)
- [9] Koolen, M., Kamps, J.: What’s in a link? from document importance to topical relevance. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) ICTIR 2009. LNCS, vol. 5766, pp. 313–321. Springer, Heidelberg (2009)
- [10] Kurland, O., Lee, L.: Pagerank without hyperlinks: structural re-ranking using links induced by language models. In: SIGIR, pp. 306–313. ACM, New York (2005)
- [11] Kurland, O., Lee, L.: Respect my authority!: Hits without hyperlinks, utilizing cluster-based language models. In: SIGIR, pp. 83–90. ACM, New York (2006)
- [12] Lempel, R., Moran, S.: Salsa: the stochastic approach for link-structure analysis. *ACM Trans. Inf. Syst.* 19(2), 131–160 (2001)
- [13] Malik, S., Trotman, A., Lalmas, M., Fuhr, N.: Overview of INEX 2006. In: Fuhr, N., Lalmas, M., Trotman, A. (eds.) INEX 2006. LNCS, vol. 4518, pp. 1–11. Springer, Heidelberg (2007)
- [14] Najork, M.: Comparing the effectiveness of hits and salsa. In: CIKM, pp. 157–164. ACM, New York (2007)
- [15] Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project (1998)
- [16] Picard, J., Savoy, J.: Enhancing retrieval with hyperlinks: A general model based on propositional argumentation systems. *JASIST* 54(4), 347–355 (2003)
- [17] Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man, and Cybernetics* 19(1), 17–30 (1989)
- [18] Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 1995), pp. 448–453 (1995)
- [19] Shakery, A., Zhai, C.: A probabilistic relevance propagation model for hypertext retrieval. In: CIKM, pp. 550–558. ACM, New York (2006)
- [20] Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence (July 2006)
- [21] Zesch, T., Gurevych, I.: Analysis of the wikipedia category graph for nlp applications. In: Proceedings of the TextGraphs-2 Workshop (NAACL-HLT 2007), pp. 1–8 (April 2007)

# Caching for Realtime Search

Edward Bortnikov<sup>1</sup>, Ronny Lempel<sup>1</sup>, and Kolman Vornovitsky<sup>2,\*</sup>

<sup>1</sup> Yahoo! Labs, Haifa

{ebortnik,rlempel}@yahoo-inc.com

<sup>2</sup> Technion CS, Haifa

`kolman@cs.technion.ac.il`

**Abstract.** Modern search engines feature real-time indices, which incorporate changes to content within seconds. As search engines also cache search results for reducing user latency and back-end load, without careful real-time management of search results caches, the engine might return stale search results to users despite the efforts invested in keeping the underlying index up to date. A recent paper proposed an architectural component called CIP – the *cache invalidation predictor*. CIPs invalidate supposedly stale cache entries upon index modifications. Initial evaluation showed the ability to keep the performance benefits of caching without sacrificing much the freshness of search results returned to users. However, it was conducted on a synthetic workload in a simplified setting, using many assumptions. We propose new CIP heuristics, and evaluate them in an authentic environment – on the real evolving corpus and query stream of a large commercial news search engine. Our CIPs operate in conjunction with realistic cache settings, and we use standard metrics for evaluating cache performance. We show that a classical cache replacement policy, LRU, completely fails to guarantee freshness over time, whereas our CIPs serve 97% of the queries with fresh results. Our policies incur a negligible impact on the baseline’s cache hit rate, in contrast with traditional age-based invalidation, which must severely reduce the cache performance in order to achieve the same freshness. We demonstrate that the computational overhead of our algorithms is minor, and that they even allow reducing the cache’s memory footprint.

## 1 Introduction

Large commercial search engines report ever-growing query volumes, leading to tremendous computation load on the data centers that serve those queries. The query distribution is highly skewed and exhibits a power-law distribution [9]. This means that the query stream exhibits high locality of reference, whereby popular (“head”) queries are repeatedly submitted by multiple users in close temporal proximity. This motivates the caching of search results by the search engine upon computing them, for the purpose of serving those same results to subsequent query submissions. Even classical replacement policies were shown to achieve hit rates of around 30% [10] on representative query streams.

---

\* Research intern at Yahoo! Labs Haifa at the time of research.

Caches serve their purpose when many queries have their results returned from the cache, rather than being evaluated by the search engine’s back-end. However, by definition, cached results were computed at a time that precedes the submission of the query. If the underlying index of the search engine has changed from the time of the results’ computation, the returned cached results may be *stale*, i.e. they may be different than what a re-evaluation of the query would have produced. Stale search results are of a particular concern in certain corpora, e.g. news feeds and social media (blogs, twitter, etc.), especially given the efforts search engines are making to index such content with very low latency<sup>1</sup>. Indexing the content quickly is useless if that content is blocked from being served by an effective cache of “old” results. While it’s trivial to avoid serving stale results by simply not caching at all, that comes at a significant computational price.

Blanco et al. [3] introduced Cache Invalidation Predictors (CIP) as means by which search engines may keep the performance benefits of caching search results, while reducing the frequency of returning stale results. The key idea is to have a component that monitors the changing content of the index in order to selectively invalidate (and evict) cached entries corresponding to queries whose top- $k$  results are likely to have been affected by the content changes.

The contributions of this work are the following. We propose additional CIP schemes that are sensitive to varying query popularities and document update rates. This allows us to fine tune the prediction beyond the methods in [3], where uniform (query agnostic) time-to-live values were applied. We then evaluate the CIP framework in a realistic, coherent setting in terms of both data and system. Data-wise, this means running our CIPs over several weeks of all document updates ingested by – and all queries submitted to – the Yahoo! news search engine. (To the best of our knowledge, this is the first study of such scale that jointly explores a dynamic document corpus and the queries submitted to it). System-wise, this means applying CIP in conjunction with a dynamic cache of search results, observing the overall performance of the system and its sensitivity to various parameters. In contrast, Blanco et al. [3] evaluated the framework over a synthetic workload and without accounting for the effects of cache dynamics.

We show that the classical least-recently-used (LRU) policy is inadequate for providing fresh results. However, augmenting it with the CIPs offers a variety of tradeoff points between hit rate and freshness. Our policies achieve 97% to 99.5% fresh results. At the lower end of this range, the impact on the hit rate as compared with LRU is minor. The CIPs induce negligible overhead – 2 to 3 orders of magnitude less than the cache miss processing. The inter-process communication required by them can be kept low, without compromising the freshness significantly. In some settings, they allow shrinking the cache by 4 without sacrificing the performance.

---

<sup>1</sup> <http://googleblog.blogspot.com/2009/12/relevance-meets-real-time-web.html>

## 2 Related Work

Caching is a well-known optimization that has been successfully applied in many computing domains since the early 1970's [1]. Applications include microprocessors [8], OS paging [1], and many more. Caches exploit the locality of reference present in application request streams. They mask the latency of expensive operations by storing their results in a fast but bounded local memory from which future identical requests are served. A *cache hit* occurs when a request whose results are cached is submitted. A *cache miss* occurs when a request whose results are not stored is submitted. In this case, the system must compute the results for the request, and typically also stores them in the cache. When the cache is full, this required the results of some other request to be *evicted* from the cache. Cached entries are evicted according to a *replacement policy*, whose goal is typically to maximize the *hit rate* of the cache - the fraction of requests resulting in cache hits. Popular cache replacement policies have emerged, e.g. the LRU heuristic [1], which evicts the least recently used entry upon a cache miss. All applications in which the cache is a replica of a dynamic remote store face the problem of *cache coherency* – keeping the cache consistent with the updates [8].

The first to study caching of search results was Markatos [10], who experimented with classical cache replacement policies (e.g. LRU and variants). Follow-up papers explored differentiated handling of query classes in separate cache segments, for optimizing hit rates [6,9] or computational costs [7]. All these works assumed indices that are generated at relatively large intervals, and that remain read-only throughout their lifetime. In a setting when the underlying index changes often, special care must be taken to ensure that search result caches continue to serve fresh results. The solutions can be classified into two categories – *decoupled designs*, which invalidate cached entries periodically independently of index, and *coupled designs*, which introduce a backdoor channel between the indexer and runtime systems. In traditional search engine architectures, these two sides do not communicate, and share only a producer-consumer relationship with respect to the index. Cambazoglu et al [4] used a decoupled design in proactively refreshing cached search results. They set age limits to cache entries, and selectively refresh aging and popular entries by re-evaluating queries when idle backend cycles are detected. This approach both increases cache hit rates and reduces the average age of cache hits. However, it is suboptimal in a real-time setting, in which awaiting light-load windows is not an option. We now survey the solution by Blanco et al. [3] (a coupled design), and our contribution.

### 2.1 Cache Invalidation Predictors

A CIP [3] is an architectural component that maintains coherency between the search result cache and the underlying index. It selectively invalidates cached entries of search results it believes have become stale since entering the cache. The CIP is composed of two parts – a synopsis generator at the indexer side, that creates summaries of documents ingested by the search engine, and an invalidator at the runtime system side, that receives the synopses and predicts which of the

cached search results will become stale due to the index modifications. The effect of invalidating the cached results of a query is identical to their eviction by a replacement policy – the cache entry is released, and the next submission of the same query will be evaluated against the index. We extend their work as follows:

*Authentic workload.* Blanco et al. used revisions of English Wikipedia as the corpus, and approximated queries against that corpus by sampling 10 000 queries that were submitted to a Web search engine and led to a click on a result from Wikipedia. The experiment was conducted on epochs of the corpus, with each epoch being a daily snapshot of Wikipedia, and the same query set was processed against each epoch. In contrast, we process the full history and query log of the Yahoo! news search engine over several weeks. We preserve the original timing (and hence, the interleaving) of corpus documents (1.8M) and queries (21.6M).

*Realistic cache settings.* We examine CIPs over a dynamic cache of finite size, with a concrete replacement policy (in contrast, [3] ignored query and cache dynamics, by effectively assuming a static infinite cache containing exactly the set of queries). We study the interplay between CIP and dynamic caches of varying sizes that use the LRU replacement. We employ the standard hit rate metric to evaluate the performance, which was meaningless in the setting of [3].

*New CIP policies.* We propose CIPs that are sensitive to varying query popularities and document update rates. In [3] it was shown that in order to bound low staleness of search results, one must couple CIP with TTL (time-to-live), invalidating cached entries once they reach a certain age. The same TTL value was applied (per experiment) to all cache entries. Thus, it is agnostic to query and document arrival dynamics. We suggest an adaptive approach called *virtual clock*, which improves over the uniform age bounding.

### 3 Problem Definition and Metrics

Consider a dynamic document collection  $\mathcal{D}$ , which is being continuously updated by a stream of document modifications  $d_1, d_2, \dots$ . A modification  $d_i$  can be an addition of a new document, an update of an existing document, or a deletion thereof. Let  $t(d_i)$  denote the timestamp of  $d_i$ . Let  $\mathcal{D}_t$  denote the snapshot of  $\mathcal{D}$  after applying the sequence of modifications  $\{d_i | t(d_i) < t\}$ . Consider a stream of queries  $q_1, q_2, \dots$  over  $\mathcal{D}$ . The timestamp of query  $q_j$  is denoted  $t(q_j)$ . A query result is an ordered list of  $k$  top-ranking documents for this query, ordered by document score with respect to it. We define the *ground truth* of  $q_j$  to be the result of  $q_j$ , as evaluated over the collection  $\mathcal{D}_{t(q_j)}$ . A query result returned by the engine is called *fresh* if it is identical to the ground truth, and *stale* otherwise. The *stale rate* at time  $t$ , denoted  $\mathcal{S}_t$ , is the fraction of stale queries till  $t$ .

A search engine may serve part of its results from a cache, which holds the results of some previous queries. An event of serving the result from the cache is called *cache hit*, whereas an event of direct evaluation is called *cache miss*. The fraction of cache hits among all requests till time  $t$  is called *hit rate*, and is denoted  $\mathcal{H}_t$ . Note that while a cache hit may be either fresh or stale, cache misses

always lead to fresh results. We study the interplay between the long-term hit rate and stale rate values, namely,  $\mathcal{S} = \lim_{t \rightarrow \infty} \mathcal{S}_t$ , and  $\mathcal{H} = \lim_{t \rightarrow \infty} \mathcal{H}_t$ .

## 4 CIP Algorithms

We describe a collection of different CIP’s that lead to a variety of tradeoff points between the system’s hit rate and stale rate metrics,  $\mathcal{H}$  and  $\mathcal{S}$ . The choice of the balance point between  $\mathcal{H}$  and  $\mathcal{S}$  can be affected by multiple factors, e.g., the freshness requirements, or the computation and communication overhead.

*Synopsis Generation.* The goal of a synopsis generator is to create a compact representative footprint of the document. Synopsis structure is closely related to the search engine’s ranking function. In general, the synopsis includes information that might affect the document’s scoring, e.g. raw content (text, embedded links, etc.), metadata (timestamp, user tags, etc.), and computed rank features. We address TF-IDF scoring [2], and employ vectors of the documents’ top-scoring terms as synopses. The synopsis size (in terms, denoted  $\eta$ ) determines the communication bandwidth between the indexer and the runtime system, as these vectors are sent from the synopsis generator to the invalidator. Intuitively, the more information a synopsis carries, the more accurate the invalidator will be.

*Query Invalidation.* We adopt some simple invalidator design principles from [3]. First, all CIPs invalidate all cache entries that include documents that have been deleted (clearly, this heuristic is always correct). With respect to new or updated documents, we make a simplifying assumption that query results are only affected by document synopses that *match* the cached queries – e.g., in conjunctive models a synopsis matches a query if it contains all of the query’s terms. The **Basic** policy invalidates all entries corresponding to queries the synopsis matches, whereas a more refined algorithm, **ScoreProjector**, invalidates a query upon a synopsis match only if the synopsis’ score for this query exceeds that of the query’s lowest-ranking cached result. The need to compute the scores of the matching  $\langle \text{query}, \text{synopsis} \rangle$  pairs renders the second policy more expensive.

Both heuristics suffer from false negatives – they fail to detect that cached results have become stale, due to term statistics drift [3]. In order to fix this drawback, the previous work suggested augmenting the CIP with age-based invalidation – replacing cached entries whose age exceeds some fixed timeout. However, that algorithm is agnostic to query frequencies, which significantly impact the cost of keeping a stale query in the cache. We augment both **Basic** and **ScoreProjector** by an event-driven *virtual clock* approach, which invalidates entries that surpass a certain number of penalizing events, denoted  $\Delta$ .

**Basic** defines the virtual clock as the number of times an entry was served since being cached. In other words,  $\Delta$  in **Basic** simply limits the number of consecutive cache hits per entry. **ScoreProjector** tracks two kinds of events, starting from the first non-invalidating match: (1) the recurring requests for this query, and (2) the further non-invalidating matches. For example, if  $\Delta = 1$ , then **ScoreProjector** – after encountering the first document to match query

$q$  without surpassing the score threshold – will invalidate  $q$  after the earliest between its first hit and the next (second overall) synopsis that matches it. Note that two low-scoring matching synopses will cause  $q$  to be invalidated.

The rationale behind this policy is as follows. Once a query is matched by some synopsis but not invalidated due to a low score, it becomes “suspicious”. The more times its potentially stale result is served, the greater the impact on the overall stale rate might be. Independently of that, every additional non-invalidating match increases the suspicion about the query result’s freshness.

Note that virtual clock, which promotes the invalidation of popular queries, is complementary to cache replacement policies, which evict infrequent queries. These two approaches pursue profoundly different goals – the first reduces the stale rate, whereas the second increases the hit rate. We further show (Section 5.2) that the two heuristics address very different segments of the query space, and hence don’t compete for the same queries. We also demonstrate that suspicious query invalidation significantly reduces `ScoreProjector`’s overhead.

*Implementation.* There are three computational tasks that invalidators perform:

*Locating all cache entries matching a given synopsis.* To perform this efficiently, we implement an inverted index over the terms of the set of cached queries. Namely, for each term appearing in at least one cached query, we maintain the list of queries that include it. The small bags of words that constitute the cached queries play the role of documents in this inverted index. Each incoming synopsis acts as a query over this index - we look up the inverted lists corresponding to the synopsis’ terms, and output cached queries that are matched. This inverted index is updated upon cache-ins of fresh result sets, as well as upon cache-outs triggered either by the CIP or by the replacement policy.

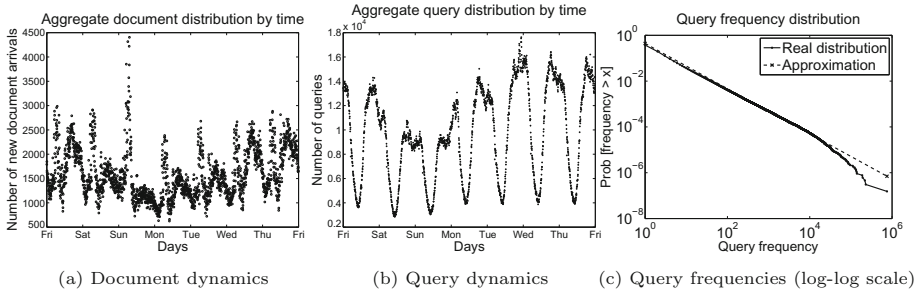
*Scoring the identified matches (ScoreProjector only).* We rely on the search engine’s ability to score a document with respect to a query. This operation, which in normal evaluation happens with a fixed query over many potential documents, is done here with a fixed synopsis over several potential queries.

*Locating all entries containing a given document id* (applied once a document is deleted). For each document that appears in at least one cached result set, we maintain the set of entries that have it among their top- $k$  results.

## 5 Evaluation

We evaluate the `Basic` and `ScoreProjector` predictors in a large-scale experiment that spans several weeks in the history of the Yahoo! news search engine. We use cache sizes including those typical for production environments, and study the interplay between the two CIPs and the least-recently-used (LRU) cache replacement policy, a de-facto caching standard [1]. We compare the performance of our algorithms to a combination of LRU with age-based invalidation. The comparison is in terms of the  $\mathcal{H}$  and  $\mathcal{S}$  metrics defined in Section 3. Visually, in the following plots, the  $(\mathcal{H}, \mathcal{S})$  curve of a policy that is superior within some operating area resides below and to the right of the curve of an inferior one.

Our experiments show that for large cache sizes (e.g., 256K entries), which are required for achieving high hit rates (above 80%), LRU replacement fails to



**Fig. 1.** Document and query statistics. (a) Week over week cumulative new document arrival dynamics, for the experiment’s duration. Each point stands for a 5-minute interval. (b) Week over week cumulative query submission dynamics, for the experiment’s duration. (c) Query frequency distribution, approximated by power-law distribution  $\log y = -0.9864 \log x - 0.3562$ .

provide fresh results for a vast majority of the queries. Different variations of augmenting LRU with simple age-based invalidation resolve the freshness issue, albeit at a high cost in terms of hit rates. For example, reducing the stale rate to 4% also reduces the hit rate by about 15%. Our CIP policies achieve far better results – e.g., one can reduce  $\mathcal{S}$  to 3% with negligible adverse impact on  $\mathcal{H}$ .

We examine the overhead of **Basic** and **ScoreProjector**, and show that they do not significantly burden the search engine’s runtime system. For example, in our experiments, the average number of cached entries that are matched per incoming document synopsis is about 15 for a **ScoreProjector** instance that achieves a stale rate of 2% and a close-to-optimal hit rate. Scoring that synopsis with respect to those queries is orders of magnitude cheaper than the average number of documents scored during query evaluation. Finally, a system designer willing to sacrifice some freshness for reducing the communication between the indexer and runtime systems, may use compact (e.g., 64-term) synopses.

## 5.1 Experiment Setup

We use the document history and the query log of the Yahoo! US news search engine, spanning several weeks in early 2010. During this period, about 1.8M new documents were added to the corpus. The amount of document updates was negligible, stemming from the append-only nature of the news feed. The lifetime of almost all documents is fixed, meaning that they are deleted after a few weeks. In a stable state, the document birth and death rates are equal, and the corpus size variations are insignificant – the index spans a few million documents. Figure 1(a) illustrates the process of document arrivals, aggregated by weekdays throughout the experiment’s duration. Each point represents a 5-minute interval. The distribution of arrival rates follows a daily recurring pattern, with occasional spikes due to breaking news.

The query log for the same period includes 21.6M queries. Most of these queries are short (on average, less than 3 terms). Figure 1(b) depicts the query



submission process, aggregated similarly to document arrivals. The workload is also periodic, with peak loads corresponding to daytimes in the US. The frequencies of unique queries follow a power-law distribution, similar to Web queries [9], albeit with a lighter tail (Figure 1(c)).

We use Lucene<sup>2</sup> as our search engine, employing conjunctive query semantics and Lucene’s default TF-IDF based ranking function for computing the top-10 results per query. The average number of documents matching a query was above 3800.

For each CIP instance, we replay the document history and the query log in parallel, while preserving the original event timing. Event types include document creation, update (rare) or deletion, and query submission. We monitor  $\mathcal{H}$  as the fraction of queries served from the cache, and  $\mathcal{S}$  as the fraction of queries that return results that are different from the ground truths.

## 5.2 Numerical Results

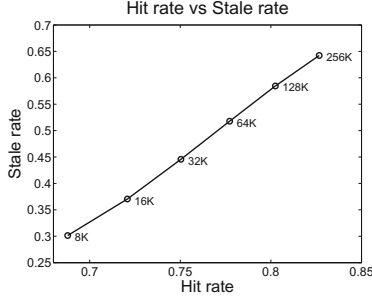
We start by exploring the baseline performance of LRU, for doubling cache sizes ranging from 8K entries to 256K entries. We start with an empty cache. The cache metrics statistically stabilize after a short warm-up period (a few hours for a 256K-entry cache – for smaller caches the stabilization takes shorter). Figure 2 depicts the  $(\mathcal{H}, \mathcal{S})$  tradeoff in this setting. The hit rate grows logarithmically with the cache size – namely, it increases approximately by 3% each time the cache is doubled. Growing the cache beyond 256K (production size in Web search [4]) produces diminishing returns. The 83% hit rate achieved for a cache of this size is higher than that reported in [4], due to smaller diversity of news queries compared to Web queries.

While LRU achieves impressive hit rates, it fails to provide fresh results over time. The bigger the cache is, the longer its entries live, and therefore, the more stale results are served. Moreover, as LRU rarely evicts popular queries, once a popular query’s result becomes stale, it will stay stale in the cache for an extended period, and will negatively affect the stale rate. For example, for a 256K-entry cache, above 65% of responses – a vast majority of hits – are stale.

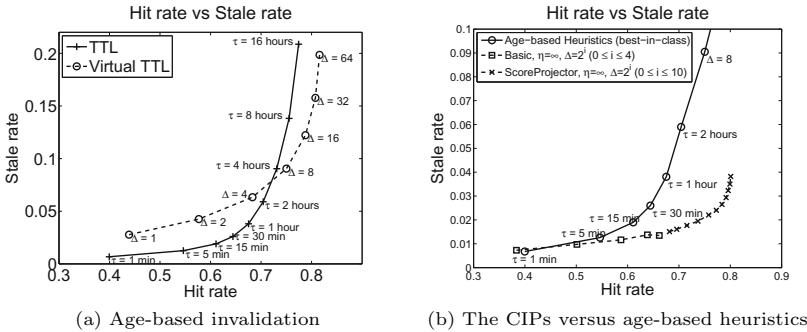
Coupling LRU with simple age-based invalidation provides a partial remedy for this problem. This policy proactively invalidates cache entries that achieve a certain age since the cache-in. This age can be measured either in real time units (*time-to-live*, or TTL), or in recurring requests for the query (*VirtualTTL*). Therefore, when a cached result ceases being fresh, its maximal impact on the stale rate is limited. Figure 3(a) depicts the comparison between the two algorithms, in terms of  $\mathcal{H}$  and  $\mathcal{S}$  values that the system converges to, for the cache size of 256K entries. Every policy instance corresponds to a single value either of the real age threshold (denoted  $\tau$ ), or a virtual threshold (denoted  $\Delta$ ). We limit our attention to instances that achieve stale rates below 20%. Smaller  $\Delta$ ’s lead to simultaneous reduction of the hit rate and the stale rate, and vice versa.

TTL achieves a better tradeoff between the system metrics for small thresholds (for which it simply overrides LRU). For example, for  $\tau = 1$  hour, the heuristic

<sup>2</sup> <http://lucene.apache.org>



**Fig. 2.** Hit rate  $\mathcal{H}$  versus stale rate  $\mathcal{S}$ , for the LRU policy without invalidation, on 8K- to 256K-entry caches. LRU achieves high hit rates but fails to prevent large stale rates.



**Fig. 3.** Hit rate  $\mathcal{H}$  versus stale rate  $\mathcal{S}$ , on a 256K-entry cache. (a) LRU with age-based heuristics: TTL versus **VirtualTTL**. The TTL policy is better for small thresholds, in which all cached queries are evicted quickly, whereas **VirtualTTL** is preferable for large thresholds, since it better adapts to the workload’s dynamics. (c) The **Basic** and **ScoreProjector** CIPs with complete synopses ( $\eta = \infty$ ), contrasted with best-in-class age-based heuristics.

returns 96.2% of fresh results, at the cost of reducing the hit rate to 67.5%. For large thresholds, **VirtualTTL** has enough time to adapt to the query stream, since it purges popular queries faster. In what follows, we compare a unified age-based invalidation policy built from the best-in-class instances of both heuristics, and compare it to the CIP algorithms on a 256K-entry cache.

We now turn to **Basic** and **ScoreProjector**. We study the impact of their parameters – synopsis size threshold  $\eta$  and virtual clock threshold  $\Delta$  – on the  $\mathcal{H}$  and  $\mathcal{S}$  metrics, as well as on their computational and communication overhead.

*Varying the virtual clock threshold  $\Delta$ .* Figure 3(b) depicts the comparison of multiple instances of **Basic** and **ScoreProjector** with the baseline age-based invalidation heuristic. We use complete synopses ( $\eta = \infty$ ), and a sequence of doubling virtual clock threshold values ( $\Delta = 2^i, 0 \leq i \leq 10$ ).

The **Basic** policy favors freshness over hit rate. **Basic**’s invalidator is stringent – all cached queries matching a new arriving document are evicted. The

CIP achieves higher freshness for the same desired hit rate than the document-agnostic age-based invalidation. Due to the invalidator’s rigidity, increasing the  $\Delta$  threshold beyond 16 has negligible impact. At the other extreme, very small  $\Delta$ ’s lead to performance close to that of TTL – low  $\mathcal{S}$  at the expense of low  $\mathcal{H}$ .

The **ScoreProjector** algorithm, which applies a more refined invalidator, is more convenient for optimizing the system’s hit rate. An instance that uses  $\Delta = 256$  achieves a hit rate of 80% (only 3% below the baseline LRU), and a stale rate of 3% – significantly lower than 19.8% incurred by the comparable age-based invalidator<sup>3</sup>. Larger  $\Delta$  values produce negligible returns. On the other hand, **ScoreProjector** with  $\Delta = 1$  induces a behavior similar to **Basic** without applying  $\Delta$ . Namely, the former invalidates a query upon the *first* penalizing event (either query hit or any kind of document match) occurring after a “suspicious” match that did not surpass the score threshold. The latter invalidates upon any match. This point partitions the applicability segments of our CIPs.

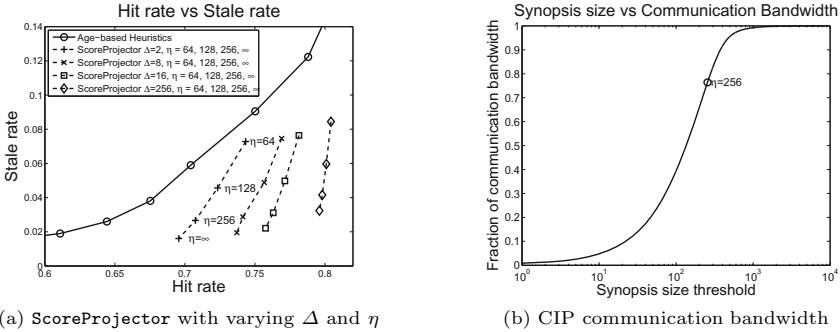
Summing up, the CIPs outperform the age-based invalidation in all points of comparison. Their advantage is most pronounced for high hit rates.

*Varying the synopsis size  $\eta$ .* Figure 4(a) depicts the performance of **ScoreProjector** for multiple synopsis size thresholds,  $\eta = 64, 128, 256$  and  $\infty$  (full synopses), and virtual clock thresholds,  $\Delta = 2, 8, 16$  and 256. Decreasing  $\eta$  leads to information loss, triggers fewer invalidations, and consequently, results in higher  $\mathcal{S}$  and  $\mathcal{H}$ . However, **ScoreProjector** outperforms the age-based invalidation heuristics even with  $\eta = 64$ . Figure 4(b) depicts the fraction of document terms employed in synopses, for varying values of  $\Delta$ . For example, using  $\eta = 256$  manifests in hit and stale rates that are quite similar to those attained by unbounded synopses, while reducing the latter’s communication bandwidth by approximately 25%.

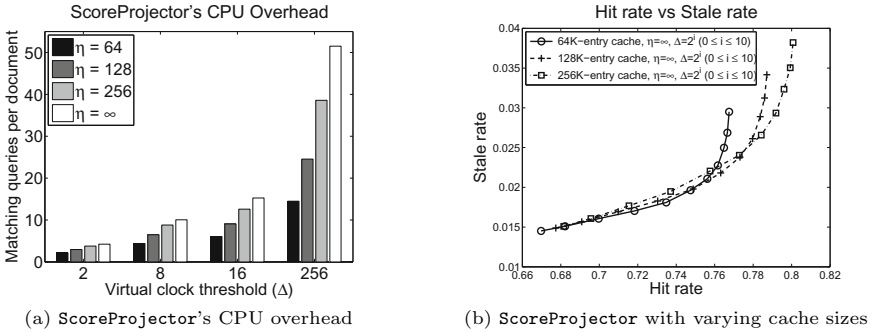
*Computational Overhead.* The previous experiments showed that **ScoreProjector** can be tuned to guarantee close-to-LRU hit rates. Hence, the price it incurs for freshness in terms of additional query evaluations due to increased misses is very small. We now further explore **ScoreProjector**’s computational overhead, to verify whether the CIP architecture is a practical solution. This overhead is dominated by computing the scores of queries matched by the incoming synopses of new documents. In Lucene, the score is roughly a scalar product between the term frequency and the inverse document frequency vectors over the matching terms (i.e., the entire query in the conjunctive model). Hence, the time for **ScoreProjector** to compute a synopsis’ scores with respect to the queries it matches is linear in the total number of terms in those matching queries.

Most of the queries in our dataset are short – 16% are one-term, and 38% are two-term. Short queries, in general, do not contribute much to the computational cost described above. Figure 5(a) depicts the average number of cached

<sup>3</sup> That is, **ScoreProjector** invalidates the most popular queries that turned “suspicious”. The LRU policy, which is complementary to the CIP, evicts only infrequent queries – mostly singletons.



**Fig. 4.** Impact of synopsis size on the CIP’s metrics and communication overhead. (a) Hit rate  $\mathcal{H}$  versus stale rate  $\mathcal{S}$ , for the **ScoreProjector** policy. We use varying synopsis size thresholds ( $\eta = 64, 128, 256$  and  $\infty$ ), and virtual clock thresholds ( $\Delta = 2, 8, 16$  and  $256$ ). (b) Fraction of maximal communication bandwidth (in synopsis terms) as function of  $\eta$ .



**Fig. 5.** **ScoreProjector**’s overhead and memory footprint. (a) Average number of matching queries, on a 256K-entry cache. We use varying virtual clock thresholds ( $\Delta = 2, 8, 16, 256$ ), and synopsis size thresholds ( $\eta = 64, 128, 256, \infty$ ). (b) Hit rate  $\mathcal{H}$  versus stale rate  $\mathcal{S}$ , with varying cache sizes (64K, 128K and 256K entries). We use complete synopses ( $\eta = \infty$ ) and running virtual clock thresholds ( $\Delta = 2^i, 0 \leq i \leq 10$ ).

queries matching a synopsis, for varying values of  $\Delta$  and  $\eta$ . We see that moderate  $\Delta$ ’s incur minor overhead. For example, the instance  $\langle \Delta = 16, \eta = \infty \rangle$ , which produces  $\mathcal{S} = 2\%$  and  $\mathcal{H} = 78\%$ , incurs on average 15 query matches per new document, on a cache of 256K entries. This is smaller by two orders of magnitude than the average number of documents ranked upon direct query evaluation in our dataset (over 3800, see Section 5.1).

The average number of invalidations per document deletion is less than 1.

*Memory Footprint.* Finally, we study the behavior of **ScoreProjector** with complete synopses on varying-size caches, using 64K, 128K and 256K entries respectively. Figure 5(b) depicts the three  $(\mathcal{H}, \mathcal{S})$  curves. As expected, achieving the optimal hit rate requires the maximal-size cache. However, large memory footprint is not needed for optimizing the stale rate. In low staleness settings,

the policy applies aggressive invalidation, which changes the dynamics between the CIP and the LRU. Most evictions are due to CIP-triggered invalidations, with the fraction of evictions performed by LRU ranging from 43% for large  $\Delta$ 's to less than 16% for small ones. Hence, the cache often remains underutilized, in contrast to the basic scenario in which it is managed entirely by LRU. For example, the 64K-entry cache is optimal if we need to maintain the stale rate below 2% – hit rates cannot be increased by larger caches at such operating points.

### 5.3 Discussion

In production search engines, the TF-IDF scoring employed by this experiment is one among many ranking signals. For example, real-time ranking of news may consider multiple recency features (e.g., [5]). In this setting, we foresee an even larger gap between the CIPs and age-based invalidators. Since TTL and VirtualTTL are agnostic to document features, they can achieve high freshness only through extremely aggressive timeouts, which have negative impact on  $\mathcal{H}$ .

One might consider extending ScoreProjector to incremental maintenance of cached query result sets, by speculatively inserting the high scoring new document into the result set instead of invalidating the entry. This approach may increase the hit rate, at the expense of result freshness, due to imprecision in comparing with historical scores. Our evaluation shows that the potential advantages of this extension are marginal (at least, for the studied dataset), since the hit rates achieved by ScoreProjector are close a system without any CIP.

## 6 Conclusions

Real-time indexing of the searched content forces the query result cache designers to face coherence problems. A recent work [3] suggested the cache invalidation predictors (CIP) framework as a remedy. CIPs judiciously invalidate the cached queries based on features of the arriving documents, and mitigate the tradeoff between the desired hit rate and stale rate metrics. We extended [3] in multiple ways. First, our work used authentic Web-scale workloads and standard system metrics. We studied production cache settings, including the LRU replacement policy. Finally, we presented new CIPs that better adapt to real workloads, and described the implementation details. Our algorithms outperform the traditional age-based invalidation policies by a vast margin. They achieve a hit rate close to pure LRU (approximately, 80%), while returning stale results for only 3% of queries. On the other extreme, the stale rate can be reduced to below 1% while retaining a hit rate of 40%. The computational overhead of CIPs is low – 2-3 orders of magnitude less than conventional query processing. The communication bandwidth can be reduced almost twofold, without compromising the invalidator precision significantly. We foresee that the CIP approach will be even more advantageous for ranking functions that employ deep document features.

## References

1. Aho, A.V., Denning, P.J., Ullman, J.D.: Principles of Optimal Page Replacement. *JACM* 18, 80–93 (1971)
2. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press / Addison Wesley, New York (1999)
3. Blanco, R., Bortnikov, E., Junqueira, F., Lempel, R., Telloli, L., Zaragoza, H.: Caching Search Engine Results over Incremental Indices. In: *SIGIR*, pp. 82–89 (2010)
4. Cambazoglu, B., Junqueira, F., Plachouras, V., Banachowski, S., Cui, B., Lim, S., Bridge, B.: A Refreshing Perspective of Search Engine Caching. In: *WWW*, pp. 181–190 (2010)
5. Dong, A., Chang, Y., Zheng, Z., Mishne, G., Bai, J., Zhang, R., Buchner, K., Liao, C., Diaz, F.: Towards recency ranking in web search. In: *WSDM*, pp. 11–20 (2010)
6. Fagni, T., Perego, R., Silvestri, F., Orlando, S.: Boosting the Performance of Web Search Engines: Caching and Prefetching Query Results by Exploiting Historical Usage Data. *ACM Trans. Inf. Syst.* 24(1), 51–78 (2006)
7. Gan, Q., Suel, T.: Improved techniques for result caching in web search engines. In: *WWW*, pp. 431–440 (2009)
8. Handy, J.: *The Cache Memory Book*. Academic Press, London (1998)
9. Lempel, R., Moran, S.: Predictive Caching and Prefetching of Query Results in Search Engines. In: *WWW*, pp. 19–28 (2003)
10. Markatos, E.P.: On Caching Search Engine Query Results. *Computer Communications* 24(2), 137–143 (2001)

# Enhancing Deniability against Query-Logs

Avi Arampatzis, Pavlos S. Efraimidis, and George Drosatos

Department of Electrical and Computer Engineering,  
Democritus University of Thrace, Xanthi 67100, Greece  
{avi,pefraimi,gdrosato}@ee.duth.gr

**Abstract.** We propose a method for search privacy on the Internet, focusing on enhancing plausible deniability against search engine query-logs. The method approximates the target search results, without submitting the intended query and avoiding other exposing queries, by employing sets of queries representing more general concepts. We model the problem theoretically, and investigate the practical feasibility and effectiveness of the proposed solution with a set of real queries with privacy issues on a large web collection. The findings may have implications for other IR research areas, such as query expansion and fusion in meta-search.

## 1 Introduction

The Internet has gradually become the primary source of information for many people. More often than not, users submit queries to search engines in order to locate content. Considering the Internet as a huge library, web-search corresponds to a search within this library. While conventional library records are private under law, at least in the U.S., Internet users might be exposed by their searches.

Every time a user submits a query to a web search engine, some private information about the user and her interests might be leaked with the query. The query representing the interest will be saved in the engine's session-logs, or it may be intercepted by the Internet provider or any other site in the network path. Table 2 presents some queries, which—depending on culture, country laws, or corporation rules—may have privacy issues. Some of those queries may correspond to malicious intentions, but we will not distinguish. There is related ongoing research on web-log anonymization, where the use of fairly advanced techniques like token-base hashing [7] and query-log bundling [6] shows that the problem is by far not solved. Thus, it currently makes sense to investigate the issue also from the other side: *how users can protect themselves*.

In September 2006, AOL released a collection with search query-log data containing about 21 million web queries collected from about 650 thousand users over three months [11]. To protect user privacy, each real IP address had been replaced with a random ID number. Soon after the release, the first 'anonymous' user had been identified from the log data. In particular, the user given the ID 4417749 in AOL's query-log was identified as the 62-old Thelma [11]. Interestingly, this identification was based solely on the queries attributed to her ID. Even though AOL withdrew the data a few days after the privacy breach, copies of the collection still circulate freely online. The incident only substantiated what was already known: web search can pose serious threats on the privacy of Internet users.

There are some countermeasures a common user can take to protect her privacy. One is to submit the query anonymously by employing standard tools, like the TOR network or some anonymization proxy. This might seem as a step in right direction, but it does not solve the privacy problem. In the AOL incident, the origin of each query was hidden, since each IP address was replaced with a random ID. However, all queries originating from the same IP were assigned the same ID. This linkability between queries submitted by the same user, resolutely increased the leakage of personal data from her query set and led to the exposition of Thelma and possibly other users. Consequently, a further step would be to make the queries of a user unlinkable. To accomplish this, a user has to continuously change her IP address and to cancel out several other information leak issues that may originate elsewhere, e.g. cookies, and embedded javascript.

Alternatively or in parallel, a user can try to obfuscate her ‘profile’ by submitting some additional random queries. In this way, the real queries are hidden in a larger set, and the task of identifying the actual interests of the user is hindered to some extent. The TrackMeNot add-on [5] for the Firefox browser implements such a feature. Another interesting add-on is OptimizeGoogle which, among other features, trims information leaking data from the interaction of a user with Google. An interesting combination of anonymization tools is employed in the Private Web Search tool [12], which is also available as an (outdated) Firefox add-on.

An interesting approach was presented in [3], where a single-term query is mixed with a set of  $k - 1$  random terms. This approach achieves at most  $k$ -anonymity, which means that each keyword can be assumed to be the actual keyword with probability of  $1/k$ . In our view, the concept of  $k$ -anonymity provides a handy tool to quantify privacy. However, as it is applied in [3] it raises practical issues; the number of terms in a search query is bounded by a small number, for example, Google’s API allows a maximum of 32 keywords. The problem further escalates for multi-term queries, where the mixed query consists of  $k$  multi-term expressions. Another related work is the plausibly deniable search technique of [8] where a query is transformed into a canonical form and then submitted along with  $k - 1$  appropriately selected cover queries. A survey on issues and techniques for preserving privacy in web-search personalization is given in [13].

There is an important reason why the above tools and methods alone might be inadequate: in all cases, the query is revealed in its clear form. Thus, privacy-enhancing approaches employing proxies, anonymous connections, or  $k$ -anonymity, would not hide the *existence of the interest* at the search engine’s end or from any sites in the network path. In addition, using anonymization tools or encryption, the plausible deniability against the *existence of a private search task* at the user’s end is weakened.

Finally, there is also the related field of Private Information Retrieval (PIR). In PIR, the main problem addressed is to retrieve data from a database without revealing the query but only some encrypted or obfuscated form of it, e.g. see [18,9]. An interesting approach for private information retrieval that combines homomorphic encryption with the embellishment of user queries with decoy terms is presented in [10]. However, all these PIR methods have an important limitation: they assume collaborative engines.

In view of the limitations of the aforementioned approaches, we define the Query Scrambling Problem (QSP) for privacy-preserving web search as: Given a query for a web search, it is requested to obtain related web documents. To achieve this, it is



allowed to interact with search engines, but without revealing the query; the query and the actual interest of the user must be protected. The engines cannot be assumed to be collaborative with respect to user privacy. Moreover, the amount of information disclosed in the process about the query should be kept as low as possible.

To address QSP, we propose the QueryScrambler; in a nutshell, it works as follows. Given a query corresponding to the intended interest, we generate a set of *scrambled queries* corresponding loosely to the interest, thus blurring the true intentions of the searcher. The set of scrambled queries is then submitted to an engine in order to obtain a set of top- $n$  result-lists which we call *scrambled rankings*. Given the scrambled rankings, we attempt to reconstruct, at the searcher's end, a ranking similar to the one that the query would have produced, which we call *target ranking*. The process of reconstruction we call *descrambling*.

The novelty of the QueryScrambler is that it does not reveal the important terms of the exposing query, but it employs semantically related and less exposing terms. The amount of privacy gained can be controlled by users via a parameter which determines the minimum semantic distance between the intended query and each of the scrambled queries issued. In this respect, the QueryScrambler only protects the query against query-logs or sites in the network path. Thus, an adversary with knowledge of the method could potentially reverse the procedure getting to the actual interest, nevertheless, this is easy to fix. In practice, the QueryScrambler can—and should—be combined with other orthogonal methods, such as those mentioned earlier. Especially, adding random queries and/or querying via multiple proxies/agents can make adversarial descrambling nearly impossible.

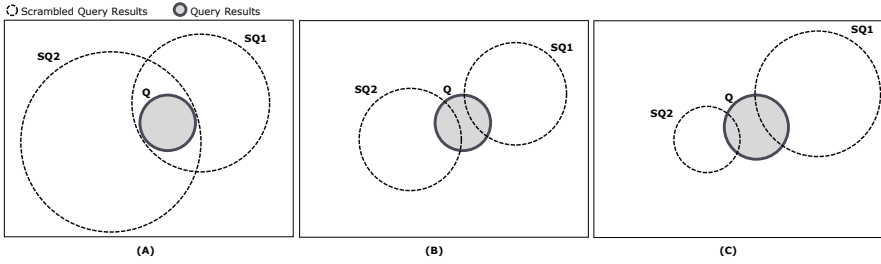
The QueryScrambler introduces an overhead over traditional web-search. We are currently not interested in its efficiency, as long as its requirements are within the reaches of current commodity desktop systems and retail Internet speeds. What we are interested in is its feasibility, focusing on the trade-off between privacy and quality of retrieved results. The method may be lossy, in the sense that the quality of results may degrade with enhanced privacy.

## 2 A Query Scrambler

The proposed QueryScrambler is based on a semantic framework (Section 2.2). First we discuss feasibility issues.

### 2.1 Theoretical and Practical Feasibility

There is no question of the theoretical feasibility of a near lossless QueryScrambler. Suppose we submit to the engine scrambled queries consisting of very frequent words, e.g. *near* stop-words. A few such scrambled queries could cover almost all the collection, which then could be downloaded to the user's site, indexed, and searched with the query. Accounting for the difference between the retrieval models, that of the engine's (usually proprietary) and that of the user's, a near-target or satisfactory ranking could be produced locally without revealing the user's target interest. In reality, such a procedure would be highly impractical or impossible for large web search engines.



**Fig. 1.** Results for two scrambled queries in relation to a query  $Q$ : (A) all results in a concept space of uniform density, (B) top- $n$  results in a uniform document space, (C) top- $n$  results in a non-uniform document space.  $Q$  represents all relevant results.

Having established the theoretical feasibility of near lossless solution to QSP with the procedure described above, what we are interested in is the trade-off between the *descrambled ranking quality* and the following three quantities: (1) *scrambling intensity*, i.e., the minimum semantic distance between the query and the set of scrambled queries, (2) *query volume*, in terms of the cardinality of the scrambled query set, and (3) *ranking depth*, i.e., the number of results returned by the engine for a scrambled query. The scrambling intensity represents the degree of hiding the true intentions; it should be given the highest priority and be kept high, affecting negatively the ranking quality. Query volume and ranking depth have the largest impact on the practical feasibility of the task; they should be kept low, affecting again negatively the ranking quality.

In practice, web search engines usually do not return the full set of results, but truncate at some rank  $n$ . For example, the Google API returns a maximum of top-1000 results per query. In this respect, we could eliminate the depth from the parameters by setting it to top-1000, a rather sufficient and practical value.

## 2.2 A Semantic Framework

Simplifying the analysis, let us assume that a query represents a single concept. Concepts more general to the query, i.e., *hyper-concepts*, would completely cover the query's concept, as well as other concepts. In this respect, some other query representing one of the hyper-concepts of the query would target more results than the query but include all results targeted by the query. Privacy for the query can be enhanced by searching for any of the hyper-concepts instead and then filtering the results for the query concept. Thus, queries representing hyper-concepts of the query can be used as scrambled queries (SQ).

Figure 1A depicts an idealized concept space. As an example consider a query  $Q$  representing the concept 'herpes' (the disease), but searching for the concept of 'infectious disease'. SQ1 could represent 'infectious disease'. SQ2 could represent 'health problem', a more general concept than this of SQ1 denoted by covering a larger area in the space. We assume that the space has a uniform concept density. Both SQ1 and SQ2 cover  $Q$  completely.

Trying to transform Figure 1A to a document space, some important issues come at play. First, concept retrieval via the bag-of-words paradigm is inherently noisy. Se-

mantic relations between keywords or phrases are seldom used. Thus, using concept names as keywords, e.g. using ‘infectious disease’ directly as SQ1, would count on 100% co-occurrence of this phrase on all documents containing the word ‘herpes’ in order to fulfill Figure 11A. Second, web search engines usually do not return the full set of results but truncate at some rank  $n$ . Third, document spaces are non-uniform, a direct result of the collection at hand not covering all concepts equally. Let us first consider an idealized uniform document space. The first issue would result to SQ1 and SQ2 circles not covering Q completely, with their centers positioned at slightly different areas (assuming keyword retrieval approximates well concept retrieval). The second issue would enforce equal circle areas for SQ1 and SQ2, denoting  $n$  results (assuming that both scrambled queries have  $\geq n$  results). These are depicted in Figure 11B.

Factoring in non-uniformity of the document space, i.e., the third issue, the picture changes to Figure 11C; the SQ2 area is denser than the area of SQ1, denoted by the reduced area covered by  $n$  results. The size of the Q area may also change, depending on the number of relevant results in the collection. Obviously, a single SQ would not cover all results corresponding to the query, so for a full coverage multiple SQs would have to be used.

### 2.3 Current Implementation

In order to generate scrambled queries representing hyper concepts of the query, we currently employ an ontology for natural language terms. The approach taken is a brute force one which does not involve deep semantic analysis of the query.

First, we perform a massive indiscriminate generalization taking all possible combinations of generalized query terms up to a certain higher conceptual level. Then, we apply a similarity measure to determine the distance between the query and scrambled queries; the further the distance, the better the privacy enhancement. In this respect, the similarity measure is ‘loaded’ with the task of classifying the scrambled queries into privacy levels, getting rid at the same time of generalized queries unsuitable to the task.

**Query Generalization.** As an ontology, we employ WordNet version 3.0 (2006). Initially, WordNet’s lemmatization process is applied to each keyword of the query, followed by stopword removal using the traditional SMART system’s English stoplist. Then, possible collocations are recognized by checking consequent query words against WordNet. All resulting *terms* (i.e. single keywords or collocations) go through part-of-speech (PoS) and sense disambiguation.

PoS and sense disambiguation cannot be performed well without enough contextual information, e.g. a complete sentence. Thus, we used a manual approach which gives the user more control over the whole procedure; the extra user effort is deemed insignificant in the big picture of privacy enhancement, considering also the fact that web queries consist of only 2 to 3 terms on average. The system finds all possible PoS for each term using Wordnet and prompts the user to select the proper one. Similarly, the user selects the proper sense.

Hyper-concepts for query’s terms are approximated via hypernyms and holonyms for nouns, and hypernyms for verbs. For each query term, a bag of related terms is generated following the hypernymy and holonymy relations in the ontology up to a minimum level of 2 or up to 3 if level 2 results to less than 300 scrambled queries.

The set of scrambled queries is the Cartesian product of those bags of words. Thus, accounting for collocations, scrambled queries have length equal to the query.

We do not generalize adverbs or adjectives since WordNet does not have similar relations, but keep them in scrambled queries. This does not seem to be a problem; adverbs and adjectives are unlikely to have privacy issues, since they are usually modifiers to verbs and nouns, respectively.

**Measuring Privacy Enhancement.** Several methods for determining semantic similarity between terms have been proposed in the literature. We apply the approach of Wu & Palmer [16] to estimate the semantic similarity between two terms. The method has been found to be among the best edge counting methods applied on WordNet [15], and it has been used widely in the literature, e.g. [14,17]. It measures the depth of the two concepts in the WordNet taxonomy as well as the depth of the least common subsumer (LCS), and combines these figures into a similarity score  $\text{sim}_{i,j}$ , where, for the task at hand, we will denote a query term with  $i$  and a scrambled query term with  $j$ .

The similarity between pairs of terms is used to calculate the similarity between each scrambled query and the query. Let SQ be a scrambled query. If  $q$  is the length of the query, then any SQ has also length  $q$ . Thus, there are  $q^2$  term(SQ)-to-term(query) similarities. For each scrambled query term  $j$ , what determines the privacy level is its max similarity with any of the query terms, i.e.,  $\max_i \text{sim}_{i,j}$ ; the larger the max, the lesser the privacy. Similarly, for a multi-term query what determines the privacy level is the least private term, justifying again the use of max. Thus, the similarity  $\text{sim}_{\text{SQ}}$  between the scrambled query and the query is  $\text{sim}_{\text{SQ}} = \max_j \max_i \text{sim}_{i,j}$ , where  $\max_j$  selects the most exposing scrambled query term with respect to the query terms.

The last measure is a very strict criterion for privacy. In the current implementation, considering that adverbs and adjectives appear in scrambled queries unchanged, the measure would return 1 denoting no privacy. In this respect, we relax the criterion by taking the average instead:

$$\text{sim}_{\text{SQ}} = \frac{1}{q} \sum_j \max_i \text{sim}_{i,j} \quad (1)$$

On the one hand, this implies that adverbs and adjectives reduce privacy, but not destroying it altogether. This reduction makes the measure safer from a privacy perspective. On the other hand, a too general scrambled query term would not artificially enhance too much the privacy of a multi-term scrambled query: too general terms are filtered out by limiting the paths on the ontology to 2 or 3 edges.

Table 1 shows all scrambled queries generated with the current query generalization method for the query ‘gun racks’, together with their similarities to the query as these are calculated by Equation 1.

## 2.4 Descrambling Ranked-Lists

Each scrambled query run on a search engine produces a scrambled ranking. We investigate two ways of reconstructing the target ranking from many scrambled rankings.

**Fusion.** A natural and efficient approach to reconstructing the target ranking would be to fuse the scrambled rankings. However, standard fusion methods from meta-search,

**Table 1.** All scrambled queries for the query ‘gun racks’

simsq	SQ	simsq	SQ
0.9442725	weapon system support	0.8736842	weapon system instrumentality
0.9442725	weapon support	0.8736842	weapon instrumentation
0.9442725	arm support	0.8736842	weapon instrumentality
0.9150327	instrument support	0.8736842	arm instrumentation
0.9111842	weapon system device	0.8736842	arm instrumentality
0.9111842	weapon device	0.8621324	device device
0.9111842	arm device	0.8503268	instrument instrumentation
0.8952206	device support	0.8503268	instrument instrumentality
0.8819444	instrument device	0.8433824	device instrumentation
0.8736842	weapon system instrumentation	0.8433824	device instrumentality

such as CompSUM, Borda Count, etc., may not be suitable: the scrambled rankings are results of queries targeting different, more general than the query, information needs.

Figure 11C depicts a document space, with the areas targeted by a query and two scrambled queries. The further from a query’s center, the deeper in the ranking. The results we are interested in appear deeper in scrambled rankings than their top ranks. To complicate things further, web search engines usually do not return scores. Thus, a fusion approach should be based solely on ranks and have a positive bias at deep or possibly middle ranks of scrambled rankings.

A simple method that may indirectly achieve the desired result is to fuse by the number of scrambled rankings an item appears in. Assuming that sets of top results of scrambled rankings, as well as sets of noisy results, would be more disjoint than sets of deep to middle results, such a fusion method would over-weight and rank at the top the common good results. We will call this *fusion by occurrence count* (FOC) descrambling. The method results to a rough fused ranking since it classifies items into  $v$  ranks, where  $v$  is the number of scrambled queries or rankings.

In order to determine whether Figure 11C corresponds well to the reality of the proposed scrambler, we will also fuse with Borda Count (BC). BC is a consensus-based electoral system which in its simplest form assigns votes to ranks as  $N - \text{rank} + 1$ , where  $N$  is the total number of items. Since  $N$  is unknown for web search engines, we set it to 1000, i.e., the depth of the scrambled lists. Then, votes per item are added for all rankings, and items are sorted in a decreasing number of total votes.

Note that BC results in a smoother ranking than FOC. Nevertheless, both approaches suffer from the low correspondence of ranks to relevance.

**Local Re-indexing.** Another approach to re-constructing a target ranking, which does not suffer from low correspondence of ranks to relevance and produces smoother rankings than FOC or BC, would be to recover item scores. This can be achieved by re-indexing the union of scrambled results at the user’s end, and running the query against a local engine. We will call this method *local re-indexing* (LR) descrambling.

Re-indexing such non-random subsets of a web collection would locally create different frequency statistics than these at the remote end. This may result in a ranking quality inferior to the target ranking, even if all target results are found by the scrambled queries and re-indexed. Furthermore, it is inefficient compared to the fusion approaches:

retrieving and indexing the union of results may introduce a significant network load, increased disk usage, and CPU load.

### 3 Evaluation

In order to evaluate the effectiveness of the QueryScrambler and how its quality trades off with scrambling intensity and scrambled query volume, we set up an offline experiment. First, we describe the datasets, the software and parameters, and the effectiveness measures used. Then, we present the experimental results.

#### 3.1 Datasets, Tools and Methods

The test queries were handpicked from real queries of the AOL query-log [11]. The full AOL dataset consists of 21 million queries from the AOL search (March–May 2006). Four human subjects independently selected a total of 95 queries which, in their opinion, may have required some degree of privacy. Table 2 presents a sample of the test queries; we will make their full set available online.

The ClueWeb09 dataset consists of about 1 billion web pages, in 10 languages, crawled in January and February, 2009. It was created by the Language Technologies Institute at CMU. It can be considered compatible with the test query set, since one of the many methods used to develop the ClueWeb09 employed queries sampled from the AOL query-log. As a document collection, we used the ClueWeb09\_B dataset consisting of the first 50 million English pages of the ClueWeb09 dataset.

The dataset was indexed with the Lemur Toolkit V4.11 and Indri V2.11, using the default settings of these versions, except that we enabled the Krovetz stemmer. We used the baseline language model for retrieval, also with the default smoothing rules and parameters. This index and retrieval model simulate the remote web search engine.

A local engine re-indexes, per query, the union of sets of results returned by the remote engine for all scrambled queries. For the local engine, we again used the Lemur Toolkit and Indri, but in order to simulate that a remote engine’s model is usually proprietary, we switched the local retrieval model to tf.idf. The items for re-indexing were extracted as term vectors directly from the remote engine’s index; this implies a common pre-processing (e.g. tokenization, stemming, etc.) across the remote and local engines.

There are several ways for measuring the top- $n$  quality of an IR system, e.g. precision and recall at various values of  $n$ , mean average precision (MAP), etc. These compare two top- $n$  lists by comparing them both to the ground truth, but this presents two limitations in the current setup. First, such measures typically give absolute ratings of top- $n$  lists, rather than a relative measure of distance. Second, in the context of the web, there is often no clear notion of what ground truth is, so they are harder to use.

**Table 2.** A sample of the 95 queries with possible privacy issues, handpicked from the AOL log

welfare fraud	post traumatic stress
rehabs in harrisburg pa	herpes
how to make bombs	lawyers for victims of child rape
hazardous materials	acute hepatitis
gun racks	police scanner

We are interested in the quality of the re-constructed ranking in terms of how well it approximates the target ranking, not in the degree of relevance of the re-constructed result-list. Although, this could still be measured indirectly as a percentage loss of a traditional IR measure (assuming ground-truth exists), e.g. MAP, we find more suitable to opt for direct measures of result set intersection and rank distance. In this way we will still measure the effectiveness even for queries poorly formulated for the information need, or information needs with near zero relevance in a collection. A simple approach to measure the distance between two top- $n$  lists  $\tau_1, \tau_2$ , is to regard them as sets and capture the extent of overlap between them. We measure the overlap with the following intersection metric (IM), which is based on the symmetric difference of the two lists:  $\text{IM}(\tau_1, \tau_2) = |(\tau_1 - \tau_2) \cup (\tau_2 - \tau_1)| / (|\tau_1| + |\tau_2|)$ . It lies in  $[0, 1]$ , with 1 denoting disjoint lists. For lists of the same size, IM equals 1 minus the fraction of overlap.

Traditional measures of rank distance (i.e., distance between two permutations), such as Kendall's tau distance or Spearman's rho, are not very suitable because our lists are truncated so they may rank different results. Thus, we use *Kendall's distance with penalty parameter  $p$* , denoted  $K^{(p)}$ , which is a generalization of Kendall's tau distance to the case of truncated lists.  $K^{(p)}$  was introduced in [4], where it was shown that it is not a metric in the strict mathematical sense, but still a near metric in the sense of satisfying a 'relaxed' triangle inequality. On the other hand, IM is a metric. A very important feature of the Kendall's distance with penalty parameter  $p$  is that it is a measure that can be applied even if the lists are obtained from very a large universe whose exact size might be unknown, thus it is suitable in the web retrieval context.

We evaluate with the averages of both measures on the test query dataset at top- $\ell$  for  $\ell = 50$  instead of  $n = 1000$ . We find top-50 to be more realistic for web retrieval than the top-1000 of traditional IR evaluations. In addition, this allows us to put our results somewhat in perspective with the  $K^{(0)}$  results for top-50 reported in [4] where rankings returned from different web search engines for the same query are compared to each other. In initial experiments, we found that  $K^{(0)}$  and  $K^{(0.5)}$  get values not too far away from each other. The authors in the last-mentioned study regard values of around 0.3 as 'very similar' rankings, while comparing a ranking fused from several engines to the individual rankings generated  $K^{(0)}$  distances between 0.3 and 0.8.

### 3.2 Experiments and Results

We run experiments for 3 levels of scrambling intensity and 3 levels of query volume. By looking into the sets of scrambled queries generated via the method described in Section 2.3, it seemed that a test query to scrambled query similarity of less than 0.70 results in extremely weak semantic relationship between the two. Consequently, we took the similarity intervals of  $(1, 0.7]$ ,  $(0.9, 0.7]$ , and  $(0.8, 0.7]$ , for low, medium, and high scrambling respectively. For scrambled query volume, we arbitrarily selected volumes in  $\{1, 10\}$ ,  $\{11, 25\}$ , and  $\{26, 50\}$ , for low, medium, and high volume respectively.

Where a combination of intensity and volume levels had 0 scrambled queries for a test query, we did not take that test query into account in averaging results. In such cases, search privacy for the query at the requested scrambling intensity and volume is not possible with the proposed method and other methods must be applied. Table 3 presents the number of test queries averaged per combination. In the parentheses, we

**Table 3.** # of test queries and (min, median, max) # of scrambled queries per scrambling/volume

		scrambling		
		low	med	high
volume	high	55 (27,50,50)	33 (29,50,50)	19 (26,50,50)
	med	72 (11,25,25)	62 (13,25,25)	30 (11,25,25)
	low	94 (3,10,10)	88 (1,10,10)	58 (1,10,10)

**Table 4.** Mean  $K^{(0.5)}$  and IM for FOC

		mean $K^{(0.5)}$			mean IM		
		scrambling			scrambling		
		low	med	high	low	med	high
volume	high	.980	.989	.998	.985	.992	.999
	med	<b>.961</b>	.978	.998	<b>.968</b>	.983	.999
	low	.962	.969	.993	.971	.977	.996

**Table 5.** Mean  $K^{(0.5)}$  and IM for BC

		mean $K^{(0.5)}$			mean IM		
		scrambling			scrambling		
		low	med	high	low	med	high
volume	high	.970	.981	.994	.978	.987	.996
	med	.944	.971	.994	.956	.978	.996
	low	<b>.927</b>	.958	.983	<b>.944</b>	.969	.988

further give the minimum, median, and maximum numbers of scrambled queries that the test queries had for the combination at hand. The combinations with the fewest test queries are the ones where a high volume was requested, especially at high scrambling; the proposed method can generate a limited number of scrambled queries. This can be a limitation of all ontology-based methods which statistical methods may not have.

Tables 4 and 5 present the mean  $K^{(0.5)}$  and IM (Section 3.1) for FOC and BC descrambling (Section 2.4) respectively. The best results are expected at the top-left corners of the tables for both measures, i.e. high-volume/low-scrambling, and are expected to decline with decreasing volume and/or increasing scrambling. The best experimental results are in boldface. In all experiments, the two measures appear correlated, in the sense that a better IM also implies a better ranking or  $K^{(0.5)}$ .

The best IM results correspond to an average intersection of only 2 or 3 results between fused and target top-50 rankings, for both fusion methods. In any case or measure, BC works better than FOC. This seems to be a result of the rougher ranking that FOC provides, since the results of the two methods become closer as volume increases. Results degrade with increasing scrambling, as expected, but also degrade with increasing volume. The later is due to the fact that larger volumes of scrambled queries presuppose larger degrees of scrambling even within the same scrambling interval.

Table 6 presents results for LR descrambling (Section 2.4); they are much better than the fusion descrambling results. The unexpected degradation with increasing volume appears again, but only at low or med scrambling. However, it is now more difficult to explain, and we can only speculate that it is a result of having biased global statistics in the local collection. Here, the best IM result corresponds to an average intersection of 7 to 8 results between descrambled and target top-50 rankings.

The task we set out to perform is daunting. Nevertheless, we get to the same 7 or 8 results of the top-50 of the plain query, without submitting its important keywords; we consider this a decent result. In principle, we may be uncovering relevant documents which do not contain any of the keywords of the plain query. However, this is difficult to measure without having the absolute ground-truth.



**Table 6.** Mean  $K^{(0.5)}$  and IM for LR

		mean $K^{(0.5)}$			mean IM		
		scrambling			scrambling		
		low	med	high	low	med	high
volume	high	.848	.898	.864	.891	.926	.906
	med	.832	.883	.901	.876	.915	.932
	low	<b>.812</b>	.870	.914	<b>.856</b>	.903	.940

**Table 7.** Mean number of the target top-50 results found by all scrambled queries

		scrambling		
		low	med	high
volume	high	11.1	9.7	7.5
	med	12.1	7.8	5.1
	low	<b>12.7</b>	8.0	4.3

In order to measure the quality of scrambled queries without the influence of de-scrambling, we can look at the number of the target top-50 results found by all scrambled queries combined. Table 7 presents these numbers, averaged over all test queries. The previously best result of 7 or 8 is now raised to almost 13. We see improvements of at least 40% and up to 100% all over the table. In other words, although the scrambled queries retrieve quite a few of the target top-50 results, local re-indexing can rank roughly half or two-thirds of those in the descrambled top-50. This is clearly due to having biased term frequency statistics in the local collection, and results can easily be improved by using a generic source of frequencies instead.

## 4 Conclusions

We introduced a method for search privacy on the Internet, which is orthogonal to standard methods such as using anonymized connections, agents, obfuscating by random additional queries or added keywords, and other techniques preventing private information leakage. The method enhances plausible deniability against query-logs by employing semantically more general queries for the intended information need. The key assumption is: the more general a concept is, the less private information it conveys; an assumption deemed true by example. We theoretically modeled the problem, providing a framework on which similar approaches may be built in the future.

The current implementation is based on a semantic ontology without using sophisticated natural language processing techniques or deep semantic analysis. It is arguably a brute force approach focusing on investigating the practical feasibility of the proposed method and the trade-off between quality of retrieved results and privacy enhancement. The proposed scrambling method gets up to 25% of the top-50 target results, at the ceiling of its performance. Obviously, there is a price to pay for privacy, i.e. a retrieval effectiveness loss. We investigated this trade-off in a system study; it should also be investigated in a user study in order to determine the levels of trade-off users find acceptable. Overall, the exercise demonstrated promising aspects and revealed important issues that future research should tackle.

There seems to be room for improving the method of generating scrambled queries. A thorough study of query transitions, from which one might be able to take ideas for improving the scrambled queries, is in [2]. Another direction to pursue is the fusion of loosely-related data such as results corresponding to queries targeting different but related topics. This may have further extensions for meta-search, or ad hoc retrieval via multiple queries. We have merely scratched the surface of a series of interesting aspects which beyond enhancing privacy may also prove useful for improving retrieval.

## Acknowledgments

We thank J. Kamps for providing access to ClueWeb09\_B, and S. Chatzichristofis for creating Figure 1. This research has received funding from the E.U. 7th Framework Programme [FP7 2007-2013] under grant agreement no 264226: SPace Internetworking CEnter—SPICE.

## References

1. Barbaro, M., Zeller, T.: A Face Is Exposed for AOL Searcher No. 4417749 (2006), <http://www.nytimes.com/2006/08/09/technology/09aol.html> (accessed June 2010)
2. Boldi, P., Bonchi, F., Castillo, C., Vigna, S.: From "Dango" to "Japanese Cakes": Query reformulation models and patterns. In: WI-IAT, pp. 183–190. IEEE Computer Society, Los Alamitos (2009)
3. Domingo-Ferrer, J., Solanas, A., Castella-Roca, J.: h(k)-private information retrieval from privacy-uncooperative queryable databases. *Online Inf. Review* 33(4), 720–744 (2009)
4. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. *SIAM J. Discrete Math.* 17(1), 134–160 (2003)
5. Howe, D.C., Nissenbaum, H.: TrackMeNot: Resisting surveillance in web search. In: *Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*, ch. 23, pp. 417–436. Oxford University Press, Oxford (2009)
6. Jones, R., Kumar, R., Pang, B., Tomkins, A.: Vanity fair: privacy in querylog bundles. In: *CIKM*, pp. 853–862. ACM, New York (2008)
7. Kumar, R., Novak, J., Pang, B., Tomkins, A.: On anonymizing query logs via token-based hashing. In: *WWW*, pp. 629–638. ACM, New York (2007)
8. Murugesan, M., Clifton, C.: Providing privacy through plausibly deniable search. In: *SDM*, pp. 768–779. SIAM, Philadelphia (2009)
9. Ostrovsky, R., Skeith, W.I.: A survey of single-database PIR: techniques and applications. In: Okamoto, T., Wang, X. (eds.) *PKC 2007*. LNCS, vol. 4450, pp. 393–411. Springer, Heidelberg (2007)
10. Pang, H., Ding, X., Xiao, X.: Embellishing text search queries to protect user privacy. In: *VLDB* (2010)
11. Pass, G., Chowdhury, A., Torgeson, C.: A picture of search. In: *InfoScale*. ACM, New York (2006)
12. Saint-Jean, F., Johnson, A., Boneh, D., Feigenbaum, J.: Private web search. In: *WPES*, pp. 84–90. ACM, New York (2007)
13. Shen, X., Tan, B., Zhai, C.: Privacy protection in personalized search. *SIGIR Forum* 41(1), 4–17 (2007)
14. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: *AAAI*, pp. 1419–1424. AAAI Press, Menlo Park (2006)
15. Varelas, G., Voutsakis, E., Raftopoulou, P., Petrakis, E., Milios, E.: Semantic similarity methods in wordnet and their application to information retrieval on the web. In: *WIDM*, p. 16. ACM, New York (2005)
16. Wu, Z., Palmer, M.: Verb semantics and lexical selection. In: *Proc. of the 32nd Ann. Meeting of the Assoc. for Computational Linguistics*, Las Cruces, New Mexico, pp. 133–138 (1994)
17. Yan, P., Jiao, Y., Hurson, A., Potok, T.: Semantic-based information retrieval of biomedical data. In: *SAC*, p. 1704. ACM, New York (2006)
18. Yekhanin, S.: Private information retrieval. *Commun. ACM* 53(4), 68–73 (2010)

# On the Contributions of Topics to System Evaluation

Stephen Robertson

Microsoft Research Cambridge  
ser@microsoft.com

**Abstract.** We consider the selection of good subsets of topics for system evaluation. It has previously been suggested that some individual topics and some subsets of topics are better for system evaluation than others: given limited resources, choosing the best subset of topics may give significantly better prediction of overall system effectiveness than (for example) choosing random subsets. Earlier experimental results are extended, with particular reference to generalisation: the ability of a subset of topics selected on the basis on one collection of system runs to perform well in evaluating another collection of system runs. It turns out to be hard to establish generalisability; it is not at all clear that it is possible to identify subsets of topics that are good for general evaluation.

## 1 Introduction

In some recent papers, it is claimed that certain topics are better than others for the evaluation of a collection of systems or system runs. Retrospective experiments show that, given a matrix of results on a set of topics and a collection of runs, evaluation results from certain individual topics or topic subsets are better correlated with, or better predictors of, the full results than other individuals or subsets. It is suggested that, given a set of topics, we can select a smaller subset which is sufficient for evaluation purposes, in that the results on the subset mimic or predict well the results on the full set.

This paper aims to extend the experimental investigations associated with this proposition, and to investigate the relations between two retrospective methods of identifying good topics for evaluation. The original intention was to go further and look at ways of predicting which topics might be good. However, in the event, the new experiments have cast serious doubt on the generality of the original proposition. In essence, although the basic premise concerning a given collection of runs is confirmed, the topics or topic sets that are good for evaluating one collection of runs are not necessarily good for another collection of runs. Thus the property of being a ‘good’ topic or topic subset for evaluation may not be an inherent property of the topic(s), or even of the topic(s) plus document collection, but may depend on the systems/runs as well.

This negative result is reported here on the grounds that negative (as well as positive) results *should* be published, and also because it is hoped that the effects observed will give further insights into the issues under investigation.

## 2 Two Retrospective Methods

In two recent papers, the matrix of evaluation results by system (or rather run) and topic has been used to identify which topics tell us most about the runs. The first method simply provides a per-topic measure of ‘hubness’, which is identified as a measure of how much that topic tells us about the comparative effectiveness of the runs. The second exhaustively explores subsets of the topic set, to identify the best subsets of any cardinality.

In what follows we assume that we have a standard set of evaluation results for a collection of runs, based on a TREC-style evaluation, using a set of topics, a corpus of documents, and relevance judgements. From this we extract a matrix representing topics against runs, of values of some single effectiveness metric  $M$ . In general the discussion is agnostic about the choice of  $M$ , although there may be reasons why it applies more naturally to some metrics than to others.

Mizzaro & Robertson [7] manipulate the matrix to produce a new matrix representing a directed bipartite graph, with nodes consisting of all the topics and all the runs. Every topic points to each run with a normalised version of  $M$ , indicating how good it thinks the run is, and every run points to each topic with a different normalised version of  $M$ , indicating how easy it thinks the topic is. The matrix is then analysed using the method of HITS [6], which gives each node a hubness score and an authority score. In the case of topics, the hubness score is interpreted as indicating how well each topic contributes to the overall evaluation of runs, and the authority score indicates how easy or hard the topic is. Dual interpretations are constructed for the hubness and authority of runs.

Guiver *et al.* [3] exhaustively explore different possible subsets of the topic set, and evaluate each subset according to how well it predicts evaluation on the full set of topics. The paper reports a series of experiments, with different metrics  $M$ , different measures of how good the prediction is, and various tests on generalisation, including testing on held-out topics and on held-out runs (the latter is discussed further below). One other conclusion of this paper was that good topic sets are not just made up of good topics: there is some complementarity effect, whereby a topic that is not so good on its own might be good in combination with other topics.

In general, both these papers find evidence of strong discrimination between topics, as to their value in evaluating runs. Subsequent work [4] has confirmed and extended this result; related work [1] uses different topics to evaluate different systems in a set. The generalisation experiments appear to indicate the possibility of identifying the good topics prospectively or predictively, so as to avoid expending scarce resources on judgements for less-useful topics.

The objectives of the work reported in this paper were initially: (a) to investigate the relation between the HITS method and the topic subsets method – in particular, whether the hubness of topics predicts their value in good subsets; (b) to expand the scope of the generalisation experiments; (c) to begin to seek methods of predicting which topics or topic sets are good. (a) relates to the question of individual topics versus topic sets: could an individual-topic-based measure predict (to any useful degree) the contribution of a topic to a set. (b) is motivated

by the acknowledged limitations of the generalisation experiments previously reported [3], which present several difficulties. (c) would be informed by (a), but follows fairly obviously from the hoped-for success of (b). However, in the event, the experiments for (b) cast some doubt on generalisability across collections of runs of the ‘goodness’ of topics or topic sets, so the focus of the present paper is on this generalisability question. (c) is left for future work.

## 3 Data

### 3.1 Collections, Topics and Runs

The data used for both the above papers, as well as by Voorhees & Buckley [9], was based on results from the TREC 8 Ad Hoc track; the same data are used again below. The TREC results are for 129 runs on 50 topics; following previous work, the 25% worst-performing runs (by MAP) are eliminated, leaving 96 runs. The document collection used is the one contained in TREC Disk 4 and 5, excluding the Congressional Record sub-collection.

However, in order to do a new comparison involving held-out runs, an additional set of results was obtained, on the same topics and documents. The new set consists of a series of 20 runs on the Terrier system, using four different weighting models (BM25, the Hiemstra language model, PL2 and TF\*IDF) and five different query versions (title, title+description, title+description+narrative, and two query expansion runs using pseudo relevance feedback)<sup>1</sup>. This new collection is referred to below as the Terrier data. It has some notable differences from the TREC runs: (a) all runs use the same initial parsing (including Porter stemming and stopword removal); (b) the collection constitutes a systematic matrix of certain variables (weighting schemes and forms of query); (c) all other system variables are held constant. MAP values range from 0.30 down to 0.16.

The TREC data, on the other hand, are very heterogeneous, a fact which has caused problems to some researchers in the past. One feature of the TREC data is the inclusion of 13 manual runs, ten of which constitute the ten best runs by MAP. It is well understood that manual runs exhibit some very different characteristics from automatic runs. In view of the differences uncovered below, and in an effort to understand them better, we have made up a third collection, which might be expected to be a little more comparable to the (fully automatic) Terrier runs, by eliminating the manual runs from the TREC data. We then follow the same procedure as above and eliminate the 25% worst-performing runs. This leaves us with 87 automatic TREC runs.

These differences in the collections of runs are a deliberate choice for the present experiments. If we find that the same topics are good for distinguishing different collections of runs, then this will constitute strong evidence that some topics or topic sets are better than others for evaluation in general. If there are characteristics of the collection of runs that affect which topic sets are good or bad, then we may expect such characteristics to play a role here, and to find that

<sup>1</sup> I am very grateful to Evangelos Kanoulas for providing these runs. They were made using Terrier 2.2.1, and are similar to those used by Kanoulas *et al.* [5]

the different collections of runs point to different subsets of topics. Below, the three collections of runs are referred to as TREC96 (mixed manual-automatic), TREC87 (automatic only) and Terrier.

Loosely, we could consider any of these three collections as a training set (from which to obtain a best subset of topics), and any other as a held-out test set (to be used only for testing). However, there is a significant asymmetry, in the way the relevance judgements were arrived at. The judgements used are the standard TREC qrels, which were generated in the usual way by judging pools of documents retrieved by the participants. This means that using TREC data for a training set and Terrier as a test set makes reasonable sense, in that it is a realistic scenario. Using Terrier as a training set and one of the TREC datasets as a test set makes much less sense, and clearly training on one version of TREC and testing on the other invites considerable overfitting. In addition, the number of runs in the Terrier data is relatively small – possibly too small for reasonable training results. For these reasons, we concentrate in the hold-out experiments on using one of the TREC collections for training and Terrier for test.

### 3.2 Evaluation Metrics and Goodness Measures

The basic evaluation metric used is average precision ( $AP$ ). However, in the present paper we transform  $AP$  using the logistic (logit) transform, following Cormack & Lynam [2]. There are two reasons for this. Like the log transform, or equivalently like using the geometric mean GMAP [8], this pays attention to hard topics in a way that ordinary MAP does not. Second, the normalisations used in the HITS analysis are not anomalous in any way (see [7]). The theoretical range of  $\text{logit}AP$  is  $(-\infty, \infty)$ . However, we have to avoid actual infinity values, so the transformation actually used is  $\log((AP + \epsilon)/(1 - AP + \epsilon))$ , where  $\epsilon = 10^{-5}$ . As future work, it would be useful to analyse to what extent the choice of metric affects the conclusions of this paper.

The Best Subsets paper used three different measures of how well a given subset of topics predicts the ‘true’ evaluation based on all topics: correlation across runs of the mean of the chosen metric; rank correlation (Kendall’s tau) of the run rankings produced by the full set and the subset; an error-based measure derived from work by Voorhees & Buckley [9]. In the present work, following Hauff *et al.* [4], we concentrate on the Kendall’s tau rank correlation measure. Some initial work had suggested that this measure might be more discriminative than the others, particularly when applied to the small Terrier dataset. However, this issue has not been explored systematically here, and again is left for future work. In respect of both this gap and the one mentioned in the previous subsection, we note that the best subset method [3] is expensive of computer time, even with the heuristic mentioned below. Both space limitations in this paper and time considerations for experiments have contributed to these limitations.

## 4 Data Analysis Methods

### 4.1 HITS

All three sets of results were separately analysed according to the HITS method [7]. In each case we obtain a vector of hubness values and a vector of authority values for all runs and all topics; each vector has  $s+t$  elements, corresponding to the  $s$  runs concatenated with the  $t$  topics. In the cited paper, various correlations involving these weight vectors were given. However, visual inspection shows that in each case there are order-of-magnitude differences between the  $s$  run weights and the  $t$  topic weights; these differences will have considerably distorted the reported correlations. We have therefore separated the system/run component from the topic component. The focus of this paper is solely on the topics – the only purpose of the systems is to help us discriminate between topics (we appreciate that this is the diametric opposite of the usual view!). Therefore in each case we consider only the topic part of the vector.

‘Authority’ of topics is interpreted as topic ease [7] (that is, topics to which runs tend to give high  $AP$  values are easy topics). ‘Hubness’, on the other hand, measures the topic’s ability to distinguish between runs – a high hubness topic is one which discriminates the good and bad runs well. High hubness topics might therefore be considered as candidates for a selected smaller *set* of topics for evaluation purposes. Note however that this is a per-topic measure, without regard to the other topics. It should therefore be seen as indicating which individual topics might be good for evaluation, but without any notion of whether they complement other topics in the set.

### 4.2 Best Subsets

Next we submit both sets of results separately to the Best Subsets method [3]. For each cardinality ( $n = 1, \dots, 50$ ) we exhaustively discover the subset of  $n$  topics which best predicts results on the full set of 50 topics. We also find a ‘worst’ subset and the average of random subsets of this cardinality. In the cited paper, a variety of system effectiveness metrics is used; we restrict ourselves to the logit  $AP$  metric, which was not included in the original paper. As indicated above, of the three measures of the goodness of a subset proposed in the previous paper, we mainly use Kendall’s tau rank correlation measure.

Also we make use of the heuristic methods described in the cited paper when an exhaustive examination of all possible subsets would be prohibitive. This depends on the cumulative nature of the best and worst sets over cardinalities. Although the best set at cardinality  $n$  is not normally simply the best set at cardinality  $n - 1$  with one extra topic, there is often much correlation between them. The heuristic is to avoid considering every possible cardinality  $n$  set by restricting the analysis to those that involve replacing at most  $h$  topics in the  $n - 1$  set. That is, we consider every subset of cardinality  $n$  which overlaps with the best  $n - 1$  set by at least  $n - 1 - h$ . Here we use  $h = 3$ . This heuristic also allows us to run a simple greedy algorithm, by setting  $h = 0$  – this ensures that the set for cardinality  $n$  is chosen by simply adding the best single new topic to

the best set for cardinality  $n-1$ . It is likely that a realistic topic selection strategy would take a simple greedy approach, and we would like to know whether this is likely to seriously limit the achievable correlation.

In the present paper we use the purely retrospective version of the best subsets method, without any hold-out sets of topics or runs (but see below for the comparison between the different collections of runs). However, we introduce one further development. Much of the previous paper was based on graphs of goodness measure against cardinality of the subset, with three lines, ‘best’, ‘average’ and ‘worst’. The ‘best’ and ‘worst’ lines represent the single best/worst subset of each cardinality; the ‘average’ line represents an average over a large sample (10,000) of subsets at each cardinality. In order to get a handle on statistical significance, we add two further lines, also based on the same large sample. These are the 95% confidence lines for 1-sided tests – i.e. the 5 and 95 percentiles of the sample distribution of the measure in question, at each cardinality. Thus when we have a candidate ‘best’ set of topics for each cardinality (say from the hubness analysis), we can plot the actual goodness of these sets on the same graph. If the new line falls above the 95% confidence line, we may claim that the candidate sets are significantly better than random subsets of the topics.

### 4.3 Comparing Methods and Collections of Runs

As indicated, the two methods ask somewhat different questions and therefore get somewhat different forms of result. However, in a straightforward way, we can regard the hubness method as giving us a potential first approximation to the best subsets method. In particular, we rank topics in descending order of hubness, and construct a single candidate subset of cardinality  $c$ , consisting of the top  $c$  ranked topics, for each  $c$ . We also generate candidate worst subsets by working in the reverse order. These candidate subsets are then compared to the (retrospectively) best and worst subsets of the same cardinality, and to the 5 and 95 percentiles. We note that this is itself a retrospective test, in that both methods start from the same original evaluation data. However, the following experiments address this issue.

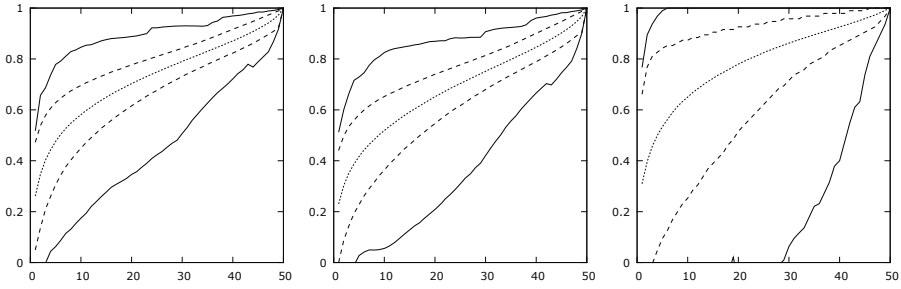
Using the HITS method, we correlate the hubness of topics according to the three collections of runs. Using the best subsets method, we ask how well the best (or worst) subset according to one collection of runs predicts the full result on another. Finally, we take the candidate sets according to the hubness analysis on one collection of runs, and examine how well they work according to the best subsets analysis on another collection.

## 5 Results

### 5.1 HITS

The HITS paper [7] reported that hubness was generally positive; this (it was argued) is to be expected: a topic with negative hubness would be one on which the poor runs generally perform better than the good runs. In the present experiments on the 96 TREC runs, just 3 topics exhibit negative hubness, but this





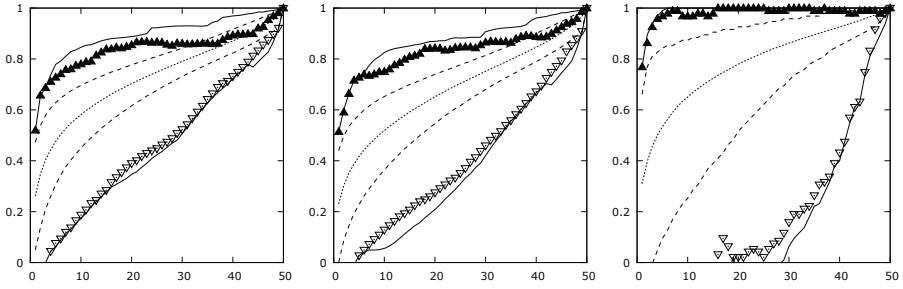
**Fig. 1.** Rank correlation values (Kendall tau) over cardinalities for  $\text{logit}AP$ :  $y$ -axis is tau,  $x$ -axis is subset size. The top and bottom lines represent the best and worst subsets respectively, the middle line represents the average of randomly chosen subsets, and the two remaining lines represent the 95 and 5 percentiles of the distribution of the metric over randomly chosen subsets (TREC96 runs on left, TREC87 runs centre, Terrier runs on right).

might be random noise. However, there is an increase for the TREC87 data (7 topics), and the numbers are higher again on the 20 Terrier runs (12). This may again be random noise (because of the smaller number of runs), but could also reflect some genuine differences. In particular, for the Terrier data, it could be an effect of the inclusion of comparable runs with and without query expansion. Since we know that the effect of query expansion is very dependent on the topic, it could be that the query expansion runs do better overall but significantly harm some topics, as has been observed in the past. These topics (likely often the same for all 8 QE runs) would then point in the opposite direction from the majority of topics for all QE *v.* non-QE comparisons. So negative hubness could be a genuine effect. This would have some consequences for the notion of a good subset of topics. We return to this effect in the discussion below.

### 5.2 Best Subsets

Results for the (retrospective) best subsets method are similar to those reported in the Best Subsets paper [3]. A representative graph, similar to those in the cited paper but with the addition of the second and fourth lines, is shown as Figure 1. This shows the rank correlation in the run rankings induced by  $\text{logit}AP$ , between the entire set of topics and subsets of cardinality  $n$  (on the  $x$  axis). The five lines represent the best, upper 95%, average, lower 95%, and worst subsets at each cardinality. We note that the margin between the best and the upper 95% line, which is where we are looking for significance, is not large.

The results for the Terrier runs show that the best subsets for a given cardinality are substantially better than the corresponding results for the TREC runs. This is probably a result of the smaller number of runs (20 versus 87 or 96), but may also reflect the fact that the Terrier data is a systematic full matrix of variants of a small number of system parameters, with others held constant. Results for the TREC87 data look very similar to the TREC96 data. Results for



**Fig. 2.** Tau values over cardinalities for subsets selected by a greedy algorithm: the same graphs as Figure 1, with two additional lines. Solid triangles are the best greedy results, open triangles are the worst greedy results. (TREC96 runs on left, TREC87 runs centre, Terrier runs on right).

other measures of goodness of fit of the subset and full set results are similar to those reported previously [3], although it appears from the present results that the Kendall tau measure may be more discriminative than the others.

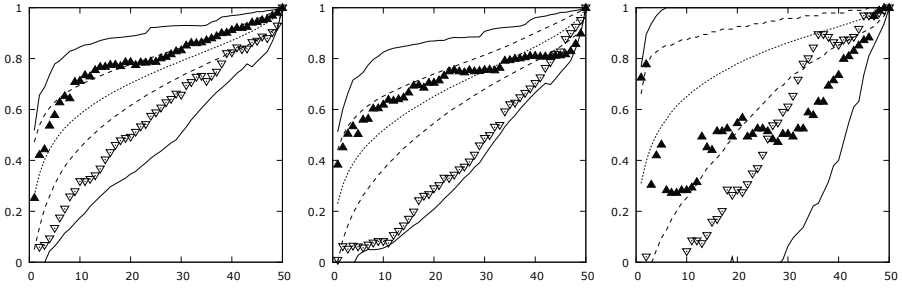
We also run a simple greedy algorithm, as discussed in section 4.2. Figure 2 shows the same graphs as Figure 1 with the addition of the best and worst greedy algorithm results. We see that the best greedy algorithm does reasonably well particularly at the beginning, but (at least for the TREC datasets) eventually falls down close to the level of the average subset.

### 5.3 Comparing Methods

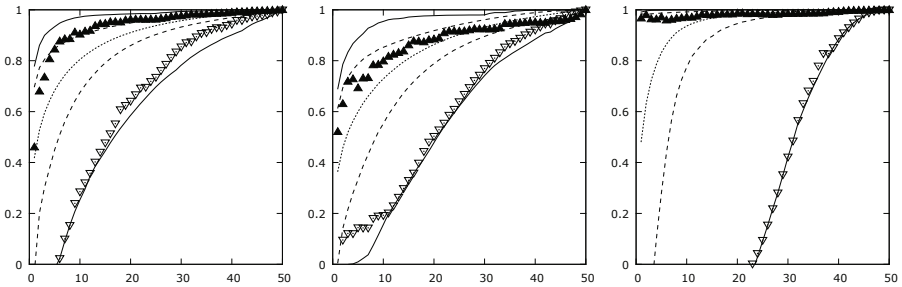
As described above, we use the ranking induced by the hubness measure to choose candidate subsets of each cardinality, and compare these to the best subsets. We also include, as "worst", candidate subsets chosen by ranking the topics in reverse hubness order. Figure 3 shows the same three graphs as Figure 1 with the addition of lines representing the candidate subsets (in each case chosen using the hubness measure on the same collection of runs as the test).

We see that the extent to which good subsets can be predicted by the hubness measure is limited. In the case of the two TREC datasets, at least in the early part of the graph the best sets are well above average and the worst sets well below, though the best sets struggle to get above the 95% line. In the case of the Terrier data, the results are chaotic, with the 'best' sets often worse than the 'worst' sets. Performing the analysis the other way around, we look at the average rank by hubness of the best sets at cardinality 10. These are 22.7 (TREC96; 25 (TREC87); 19.3 (Terrier). If there were no association at all, the expected value of this average rank would be 25.5. These results suggest barely a flicker of signal. Furthermore, we note that in the case of TREC87, the best set at cardinality 10 includes two topics with negative hubness.

It appears that whatever the hubness measure measures, it is somewhat different from what is required to identify good subsets of topics in the best subsets sense. There may be several reasons for this. We may note that if we change the



**Fig. 3.** Tau values over cardinalities for candidate subsets from hubness ranking. Again, solid triangles represent the best hubness ranking, open triangles the worst. (TREC96 runs on left, TREC87 centre, Terrier runs right).

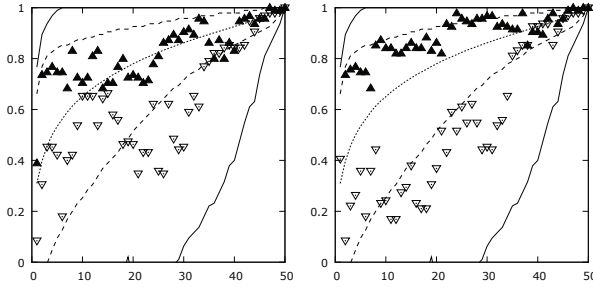


**Fig. 4.** Correlations over cardinalities for candidate subsets from hubness ranking: now the  $y$ -axis is correlation values. (TREC96 runs on left, TREC87 centre, Terrier runs right).

measure of goodness to the correlation of mean logit $AP$  values (as opposed to the rank correlation of runs), the results get very much better (see Figure 4). This is not really surprising: the hubness weight of a topic derives from the contribution of that topic to distinguishing between runs on the basis of the mean logit $AP$  value. It appears that despite this result, the hubness measures fails to determine adequately the topic's contribution to rank ordering of runs.

#### 5.4 Comparing Collections of Runs

As indicated above, what we hope to find is that the same topics or topic sets (at least to some degree) are good for the comparative evaluation of both collections of runs. This question was not asked in the original HITS paper [7]; in the Best Subsets paper [3] an attempt was made to answer this question, by dividing the TREC runs randomly into two parts, performing the analysis on one part, and evaluating the resulting best-worst sets on the other part. The analysis provided some evidence of generalisation, but with qualifications: in particular, the collection of TREC runs includes a number of runs that are essentially minor variants of the same system. (A second experiment, splitting the topics instead of



**Fig. 5.** Tau values over cardinalities for best runs from TREC data (TREC96 on left, TREC87 on right), when applied to the Terrier data

the runs, also provided some supporting evidence of generalisation.) The present experiment, with a completely new collection of runs and a new partition of the original collection, is intended as a more rigorous test of generalisation.

We first consider the HITS analysis. We measure the pairwise correlations of the vectors of hubness weights of topics derived from the three collections of runs (for a given metric). These range from -15% to +19%. This looks like a very negative result. For comparison, we look at the corresponding correlations for authority (i.e. topic ease) weights. Here the correlations are high: 92% to over 99%. In other words, the three collections of runs agree almost perfectly on which topics are easy and which are hard – but disagree completely on which topics are good indicators of system effectiveness.

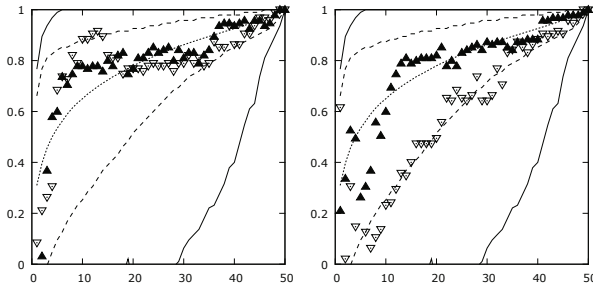
Next we consider the Best Subsets analysis. In Figure 5, we show the results of applying the best/worst subsets from the TREC data to the Terrier data. It does look as though there may be some signal here, particularly from TREC87, but there is clearly also a lot of noise. A few cardinalities of ‘best’ achieve significance, but some others are little better than random, and some worse. It does appear to be slightly easier to predict bad subsets than good ones; unfortunately this is not a very useful possibility!

Finally, we attempt to use the hubness measure derived from the TREC data (either collection) to predict good subsets for the Terrier data. Here the results are almost random – see Figure 6.

## 5.5 Discussion

The main conclusion is inescapable. Out of the 50 TREC topics, some individual topics and some subsets are good at indicating which of the 96 TREC runs are the most effective. The same statement can be made about the 87 TREC automatic runs, or the 20 Terrier runs. But neither the individual topics nor the subsets are the same in the three cases, particularly not between the TREC and Terrier collections. In general, a subset of topics that is good for distinguishing one collection runs is likely to be mediocre at distinguishing another collection, and in particular may well not be significantly better than a random subset.

In addition, the issue of the complementarity of good sets is somewhat complex. Guiver *et al.* [3] report that choosing a good subset is not just a matter



**Fig. 6.** Tau values over cardinalities for candidate subsets from hubness ranking taken from the TREC datasets (TREC96 on left, TREC87 on right), applied to the Terrier dataset

of choosing good individual topics; topics may complement each other to make a good subset. This observation is given more force by the observed negative hubness values, reported in section 5.1, and the observation in 5.3 that in one of the collections of runs, the best subset of 10 topics includes two with negative hubness.

## 6 Conclusions

The hope of this work was that we would be able to identify smaller sets of topics than are currently commonly used, which would nevertheless provide good enough measurements of system effectiveness to distinguish between runs. In contrast to much recent work on topic sampling, the idea was that the selection of particular topics would be a fruitful line to pursue.

In the event, the experiments described here have cast serious doubt on the possibility of making such selections in a way that would generalise across collections of runs. A set of topics that (together with a corpus of documents) is good for distinguishing reliably among one collection of systems or system runs may be poor at distinguishing reliably among another collection. The property of being ‘a good set of topics for evaluation’ is somewhat more subtle than our initial hypothesis suggested, and (these results suggest) there can be little hope of identifying such good sets without reference to the particular collection of runs that we are trying to evaluate.

Many experiments remain to be done. Those reported here are limited to one evaluation metric and one measure of goodness of prediction. Also, the original Best Subsets paper determined that there are other very good subsets (close to the best) but with rather different membership. It is possible that taking into account some other property, rather than simply identifying the best, might yield subsets with more generalisability. The right hand graph of Figure 5 suggests that there is indeed some signal, despite the absence of statistical significance in most cases. Thus the door is not fully closed to the topic-selection approach. However, it is very clear that generalisability is a crucial issue, and future researchers in this area would do well to pay it substantive attention.

**Acknowledgements.** Many thanks to Stefano Mizzaro and the reviewers for insightful comments on an earlier draft of this paper.

## References

1. Cattelan, M., Mizzaro, S.: IR evaluation without a common set of topics. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) ICTIR 2009. LNCS, vol. 5766, pp. 342–345. Springer, Heidelberg (2009)
2. Cormack, G.V., Lynam, T.R.: Statistical precision of information retrieval evaluation. In: SIGIR 2006, pp. 533–540. ACM Press, New York (2006)
3. Guiver, J., Mizzaro, S., Robertson, S.: A few good topics: Experiments in topic set reduction for retrieval evaluation. *Transactions on Information Systems* 27(4) (2009)
4. Hauff, C., Hiemstra, D., de Jong, F., Azzopardi, L.: Relying on topic subsets for system ranking estimation. In: CIKM 2009, pp. 1859–1862. ACM, New York (2009), <http://doi.acm.org/10.1145/1645953.1646249>
5. Kanoulas, E., Pavlu, V., Dai, K., Aslam, J.A.: Modeling the score distributions of relevant and non-relevant documents. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) ICTIR 2009. LNCS, vol. 5766, pp. 152–163. Springer, Heidelberg (2009)
6. Kleinberg, J.: Authoritative sources in a hyperlinked environment. *Journal of the Association for Computing Machinery* 46(5), 604–632 (1999)
7. Mizzaro, S., Robertson, S.: HITS hits TREC — exploring IR evaluation results with network analysis. In: SIGIR 2007, pp. 479–486. ACM Press, New York (2007)
8. Robertson, S.: On GMAP – and other transformations. In: CIKM 2006, pp. 78–83. ACM Press, New York (2006)
9. Voorhees, E., Buckley, C.: The effect of topic set size on retrieval experiment error. In: SIGIR 2002, pp. 316–323. ACM Press, New York (2002)

# A Methodology for Evaluating Aggregated Search Results

Jaime Arguello<sup>1</sup>, Fernando Diaz<sup>2</sup>, Jamie Callan<sup>1</sup>, and Ben Carterette<sup>3</sup>

<sup>1</sup> Carnegie Mellon University

<sup>2</sup> Yahoo! Research

<sup>3</sup> University of Delaware

**Abstract.** Aggregated search is the task of incorporating results from different specialized search services, or *verticals*, into Web search results. While most prior work focuses on deciding *which* verticals to present, the task of deciding *where* in the Web results to embed the vertical results has received less attention. We propose a methodology for evaluating an aggregated set of results. Our method elicits a relatively small number of human judgements for a given query and then uses these to facilitate a metric-based evaluation of *any* possible presentation for the query. An extensive user study with 13 verticals confirms that, when users prefer one presentation of results over another, our metric agrees with the stated preference. By using Amazon’s Mechanical Turk, we show that reliable assessments can be obtained quickly and inexpensively.

## 1 Introduction

Commercial search engines provide access to multiple specialized search services or *verticals*, such as image search, news search, local business search, and items for sale. There are two ways that users typically access vertical content. In some cases, if a user wants results from a particular vertical, they can issue the query to the vertical directly. In other cases, however, a user may not know that a vertical is relevant or may want results from multiple verticals at once. For these reasons, commercial search engines sometimes incorporate vertical results into the Web results. This is referred to as *aggregated search*.

Aggregated search can be viewed as a two-part task. Most prior work focuses on *vertical selection*—the task of predicting which verticals (if any) are relevant to a query [5,10,16,2]. The second task of deciding *where* in the Web results to embed the vertical results has received less attention. One possible reason for this is that a well-defined methodology for evaluating an aggregated set of results does not currently exist.

To date, aggregated results are evaluated based on user feedback, collected either implicitly (e.g., by observing clicks and skips [5,13]) or explicitly (e.g., by directly asking users which results they prefer [15]). Existing approaches, however, focus on the integration of at most a *single* vertical into the Web results. Focusing on a single vertical simplifies evaluation by limiting the space of possible layouts or *presentations* to a manageable size. User feedback can be collected for

every possible presentation of results and, thereby, we can determine, not only whether one presentation is preferred over another, but whether one is preferred over all. This is not possible, however, if we consider many verticals (e.g., more than 10) simultaneously competing for space across the search results page. In this case, the space is too large to explore fully. The question, then, is: how can we measure the quality of *any* possible presentation for a given query? This question is central to aggregated search evaluation and is the question we address in this work.

We propose and validate a methodology for evaluating aggregated search. The goal is to elicit a relatively small number of human judgements for a given query and then to use these to evaluate *any* possible presentation of results. A central component of our methodology is the prediction of a *reference* presentation, which marks the best possible presentation that a system can produce for the given query. Given the prohibitively large space of presentations, we do not elicit human judgements on full presentations. Instead, we take a piece-wise, bottom-up approach. We collect pairwise preferences on blocks of results and use these to derive the *reference* presentation. Finally, we propose that any arbitrary presentation for the query can be evaluated based on its distance (using a rank-based metric) to the *reference*. To validate our methodology we present a user study in which we test the following hypothesis: given two alternative presentations for a query, if users prefer one over the other, then they prefer the one that is closest (in the metric space) to the *reference*.

Two resources were required to validate our methodology. First, we required a wide range of operational verticals, resembling those available to a commercial search engine. We used a set of 13 verticals developed using freely-available search APIs from various on-line services (e.g., eBay, Google, Twitter, Yahoo!, YouTube). Second, we required a pool of assessors. We used Amazon's Mechanical Turk (AMT)<sup>1</sup>. By doing so, we show that the proposed methodology can be applied using a large pool of inexpensive, non-expert assessors and does not require an operational system with users. Therefore, it is applicable to both commercial and non-commercial environments.

## 2 Modeling Assumptions and Problem Definition

At query time, an aggregated search system issues the query to the Web search engine and to every vertical. At this point, every vertical that retrieves results is a *candidate* vertical. The task, then, is to decide *which* candidate verticals to present and *where* in the Web results to present them. The decision of where to present vertical results is subject to a set of layout constraints.

We make the following layout assumptions. First, we assume that vertical results can only be embedded in 4 positions relative to the top 10 Web results: above the first Web result, between Web results 3-4, between Web results 6-7, and below the last Web result. A similar assumption is made in prior work [13,15,5]. Effectively, this divides the top 10 Web results into three blocks of results, denoted as  $w_1$ ,  $w_2$ , and  $w_3$ . Multiple verticals can be embedded between any two

<sup>1</sup> <http://www.mturk.com>



Web blocks, above  $w_1$ , or below  $w_3$ . Second, we assume that users prefer to not see results from non-relevant verticals, even below  $w_3$ . Non-relevant verticals should be suppressed entirely. Third, we assume that if a vertical is presented, then a fixed set of its top results must be presented and must appear adjacent in the ranking. Finally, we assume that Web results are always presented and never re-ranked. That is,  $w_{1-3}$  are always presented in their original order.

Given these assumptions, we can formulate the aggregation task as one of ranking blocks of Web and vertical results. A *block* is defined as a set of Web or vertical results which cannot be split in the aggregated results. If a block is presented, then all its results must be presented and must appear adjacent in the ranking. If a block is suppressed, then all its results must be suppressed. Let  $\mathcal{B}_q$  denote the set of blocks associated with query  $q$ , which always includes all three Web blocks ( $w_{1-3}$ ) and one block for each candidate vertical. The aggregation task is to produce a partial ranking of  $\mathcal{B}_q$ , denoted by  $\sigma_q$ . Suppressed verticals will be handled using an imaginary “end of search results” block, denoted by *eos*. Blocks that are ranked below *eos* are suppressed. We say that  $\sigma_q$  is a *partial* ranking because all blocks ranked below *eos* (i.e., those that are suppressed) are effectively tied.

Our objective is an evaluation measure  $\mu$  that can determine the quality of *any* possible presentation  $\sigma_q$  for query  $q$ . The raw input to  $\mu$  is a set of human judgements, denoted by  $\pi_q$ . Given the prohibitively large number of possible presentations, we do not elicit judgements directly on full presentations. Instead, we take a piece-wise, bottom-up approach and collect judgements on individual blocks. Prior work shows that assessor agreement is higher on document preference judgements than on absolute judgements [3]. Here, we assume this to also be true when assessing blocks of results. Therefore, we collect preference judgements on *all* pairs of blocks in  $\mathcal{B}_q$ . We use  $\pi_q(i, j)$  to denote the number of assessors who preferred block  $i$  over block  $j$ .

A validation of  $\mu$  should be grounded on user preferences. Suppose we have two alternative presentations for a given query. If users prefer one over the other, then the preferred presentation should be the one judged superior by  $\mu$ .

### 3 Related Work

As previously mentioned, most prior work on aggregated search focuses on vertical selection. Li *et al.* [10] classified queries into two classes of vertical intent (*product* and *job*) and evaluated based on precision and recall for each class independently. Diaz [5] focused on predicting when to display news results (always displayed above Web results) and evaluated in terms of correctly predicted clicks and skips. Arguello *et al.* [1] focused on *single*-vertical selection, where at most a single vertical is predicted for each query. Evaluation was in terms of the number of correctly predicted queries. Diaz and Arguello [6] investigated vertical selection in the presence of user feedback. Evaluation was based on a simulated stream of queries, where each query had at most one relevant vertical. Arguello *et al.* [2] focused on model adaptation for vertical selection and evaluated in terms of precision and recall for each vertical independently.

The above work assumes at most a *single* relevant vertical per query and, either implicitly or explicitly, assumes a fixed presentation template (e.g., *news* results are presented above Web results, if at all [5]). Users, however, may prefer the vertical results in different locations for different queries, and, in some cases, may prefer results from multiple verticals. Our proposed methodology may facilitate a more comprehensive evaluation of vertical selection. Suppose we have access to a high-quality *reference* presentation for a query. Then, for example, we might weight a false negative selection decision more if the vertical is ranked high in the *reference* and weight it less if it is ranked low.

In terms of *where* to embed the vertical results, several studies investigate user preference behavior. Sushmita *et al.* [13] investigated the effects of position and relevance on click-through behavior. They focused on presentations with one of three verticals (*images*, *news*, and *video*) slotted in one of three positions in the Web results. A positive correlation was found between both relevance and position and click-through rate. More surprisingly, perhaps, they found a bias in favor of *video* results. Users clicked more on *video* results irrespective of position and relevance. Zhu and Carterette [15] focused on user preferences with the *images* vertical and three slotting positions. They observed a strong preference towards *images* above Web results for queries likely to have image intent. From these studies, we can draw the conclusion that users care not only about *which* verticals are presented, but also *where* they are presented.

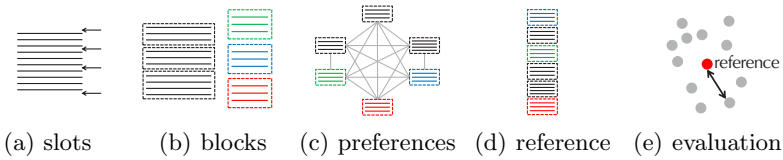
Several works elicit preference judgements on pairs of search results, as we do. Thomas and Hawking [14] validated the side-by-side comparison approach by presenting assessors with pairs of different quality (e.g., Google results 1-10/11-20, or overlapping sets 1-10/6-15). Users preferred results 1-10. Sanderson *et al.* [11] used a similar interface with Mechanical Turk to validate a set of test-collection-based metrics. NDCG agreed the most with user preferences (63% agreement overall and 82% for navigational queries).

## 4 Preference-Based Evaluation Methodology

Our objective is an evaluation measure that can determine the quality of any possible presentation  $\sigma_q$  for query  $q$ . Our method is depicted in Fig. 1. The general idea is to evaluate presentation  $\sigma_q$  based on its distance to a ground truth or *reference* presentation  $\sigma_q^*$ , which is generated from a set of preference judgements on pairs of Web and vertical blocks. Given query  $q$ , a set of blocks  $\mathcal{B}_q$  is composed from Web blocks  $w_{1-3}$  and from every candidate vertical. Each block-pair  $i, j \in \mathcal{B}_q$  is presented to multiple assessors who are asked to state a preference. Then, we use a voting method to derive  $\sigma_q^*$  from these block-wise preference judgements. Finally, we propose that any presentation  $\sigma_q$  can be evaluated by using a rank-based metric to measure its distance to  $\sigma_q^*$ .

### 4.1 Constructing the Reference Presentation

For every query  $q$ , we collected preference judgements on all pairs of blocks from  $\mathcal{B}_q$ . Each judgement consisted of a query  $q$  and block pair  $i, j \in \mathcal{B}_q$  presented



**Fig. 1.** Approach Overview

side-by-side in random order. Assessors were given three choices:  $i$  is better,  $j$  is better, and both are bad. We omitted the choice that both  $i$  and  $j$  are equally good to prevent assessors from abstaining from difficult decisions. We interpreted the assessor selecting “both are bad” as evidence that  $i$  and  $j$  should be suppressed for  $q$ . Each triplet of the form  $(q, i, j)$  was assessed by four different assessors. These preference judgements, denoted by  $\pi_q$ , are the raw input to the method that derives the *reference* presentation  $\sigma_q^*$ .

There exist many voting methods for aggregating item preference data into a single ranking. In this work, we used the Schulze voting method because of its widespread adoption and ease of implementation [12]. The general idea is the following. Let  $\pi_q(i, j)$  denote the number of assessors who preferred  $i$  over  $j$ . We say that  $i$  directly defeats  $j$  if  $\pi_q(i, j) > \pi_q(j, i)$ . That is, if more assessors preferred  $i$  over  $j$  than vice versa. A *beatpath* from  $i$  to  $j$  is a direct or indirect defeat from  $i$  to  $j$ . An *indirect* beatpath from  $i$  to  $j$  is a sequence of direct defeats from  $i$  to  $j$ . For example, if  $i$  directly defeats  $k$  and  $k$  directly defeats  $j$ , then this is an *indirect* beatpath from  $i$  to  $j$ . The strength of an indirect beatpath is the number of votes associated with its weakest direct defeat. Finally, we say that  $i$  defeats  $j$  if the strongest (direct or indirect) beatpath from  $i$  to  $j$  is stronger than the one from  $j$  to  $i$ . Blocks are then ranked by their number of defeats.

As previously mentioned, the aggregation task is not only ranking blocks, but also deciding which vertical blocks to suppress. Suppressed verticals were handled using the imaginary *eos* block. The *eos* block was treated by the Schulze method the same as any non-imaginary block. Every time an assessor selected that both  $i$  and  $j$  are bad, we incremented the value of  $\pi_q(eos, j)$  and  $\pi_q(eos, i)$ . Also, recall that we assume that Web blocks ( $w_{1-3}$ ) are always presented and never re-ranked. This constraint was imposed by setting  $\pi_q(eos, w_*) = \pi_q(w_x, w_y) = 0$ , where  $x > y$ , and by setting  $\pi_q(w_*, eos) = \pi_q(w_x, w_y) = N$ , where  $x < y$  and  $N$  is some large number (we used  $N = 1000$ ).

## 4.2 Measuring Distance from the Reference

Our proposed method is to evaluate *any* possible presentation  $\sigma_q$  by measuring its distance to the *reference*  $\sigma_q^*$ . We used a rank-based distance metric. Possibly the most widely used rank-based distance metric is Kendall’s tau ( $K$ ), which counts the number of discordant pairs between two rankings,

$$K(\sigma^*, \sigma) = \sum_{\sigma^*(i) < \sigma^*(j)} [\sigma(i) > \sigma(j)],$$

where  $\sigma(i)$  denotes the rank of element  $i$  in  $\sigma$ . Kendall’s tau treats all discordant pairs equally regardless of position. In our case, however, we assume that users scan results from top-to-bottom. Therefore, we care more about a discordant pair at the top of the ranking than one at the bottom. For this reason, we used a variation of Kendall’s tau proposed by Kumar and Vassilvitskii [8], referred to as *generalized* Kendall’s tau ( $K^*$ ), which can encode positional information using element weights. To account for positional information,  $K^*$  models the cost of an *adjacent* swap, denoted by  $\delta$ . In traditional Kendall’s tau,  $\delta = 1$ , irrespective of rank. Adjacent swaps are treated equally regardless of position. In our case, however, we would like discordant pairs at the top to be more influential. Let  $\delta_r$  denote the cost of an adjacent swap between elements at rank  $r - 1$  and  $r$ . We used the DCG-like cost function proposed in Kumar and Vassilvitskii [8],

$$\delta_r = \frac{1}{\log(r)} + \frac{1}{\log(r+1)},$$

which is defined for  $2 \leq r \leq n$ . Given rankings  $\sigma^*$  and  $\sigma$ , element  $i$ ’s displacement weight  $\bar{p}_i(\sigma^*, \sigma)$  is given by the average cost (in terms of adjacent swaps) it incurs in moving from rank  $\sigma_q^*(i)$  to rank  $\sigma_q(i)$ ,

$$\bar{p}_i(\sigma^*, \sigma) = \begin{cases} 1 & \text{if } \sigma^*(i) = \sigma(i) \\ \frac{p_{\sigma^*(i)} - p_{\sigma(i)}}{\sigma(i)^* - \sigma(i)} & \text{otherwise} \end{cases},$$

where  $p_r = \sum_2^r \delta_r$ . The  $K^*$  distance is then given by,

$$K^*(\sigma^*, \sigma) = \sum_{\sigma^*(i) < \sigma^*(j)} \bar{p}_i(\sigma^*, \sigma) \bar{p}_j(\sigma^*, \sigma) [\sigma(i) > \sigma(j)].$$

A discordant pair’s contribution to the metric is equal to the product of the two element weights.

## 5 Materials and Methods

### 5.1 Verticals and Queries

We focused on a set of 13 verticals constructed using freely-available search APIs provided by eBay (*shopping*), Google (*blogs*, *books*, *weather*), Recipe Puppy (*recipes*), Yahoo! (*answers*, *finance*, *images*, *local*, *maps*, *news*), Twitter (*micro-blogs*), and YouTube (*video*). A few example vertical blocks are presented in Fig. 2. Each vertical was associated with a unique presentation of results. For example, *news* results were associated with the article title and url, the news source title and url, and the article’s publication date, and included an optional thumbnail image associated with the top result. *Shopping* results were associated with the product name and thumbnail, its condition (e.g., new, used), and its price. *Local* results were associated with the business name and url, its address and telephone number, and the number of reviews associated with it, and



**Fig. 2.** Example vertical blocks

included a map. Each vertical was associated with a maximum number of top results (e.g., 4) from which to construct a block.

Our evaluation was conducted on a set of 72 queries from two different sources: the AOL query log and Google Trends. Google Trend queries cover recent events and topics currently discussed in news articles, blogs, and on Twitter (e.g., “us open fight”). AOL queries cover more persistent topics likely to be relevant to verticals such as *local* (e.g., “cheap hotels in anaheim ca”), *recipe* (e.g., “cooking ribs”), and *weather* (e.g., “marbella weather”). Queries were selected manually in order to ensure coverage for our set of 13 verticals.

## 5.2 Preference Judgements on Block-Pairs

While collecting block-pair judgements, in addition to the query, assessors were given a topic description to help disambiguate the user’s intent. In a preliminary experiment, we observed an improvement in inter-annotator agreement from giving assessors topic descriptions. We were careful, however, to not explicitly mention vertical intent. For example, for the query “pressure cooker”, we stated: “The user plans to buy a pressure cooker and is looking for product information.” We did not say: “The user is looking for shopping results.” The assessments were conducted using Amazon’s Mechanical Turk (AMT). Turkers were compensated 0.01 US\$ for each judgement.

Following Sanderson *et al.* [11], quality control was done by including 150 “trap” HITs (a Human Intelligence Task is a task associated with AMT). Each trap HIT consisted of a triplet  $(q, i, j)$  where either  $i$  or  $j$  was taken from a query other than  $q$ . We interpreted an assessor preferring the set of extraneous results as evidence of malicious or careless judgement. Assessors who failed more than a couple of trap HITs were removed from the judgement pool.

## 6 Assessor Agreement on Block-Pair Judgements

Of the 120 assessors who contributed HITs, 2 had their assessments removed from the assessment pool due to failing more than 2 trap HITs. For the remaining 118/120, participation followed a power law distribution—about 20% (24/118) of the assessors completed about 80% (9,856/12,293) of our HITs.

We report inter-annotator agreement in terms of Fleiss’ Kappa ( $\kappa_f$ ) [7] and Cohen’s Kappa ( $\kappa_c$ ) [4], both which correct for agreement due to chance. Fleiss’ Kappa measures the (chance-corrected) agreement between *any* pair of assessors over a set of triplets. Cohen’s Kappa measures the (chance-corrected) agreement between a *specific* pair of assessors over a *common* set of triplets. For our purpose, Fleiss’ Kappa is convenient because it ignores the identity of the assessor-pair. It is designed to measure agreement over instances labeled by different (even disjoint) sets of assessors. However, precisely because it ignores the identity of the assessor-pair, it is dominated by the agreement between the most active assessors, which we know to be a selected few. To compensate for this, in addition to Fleiss’ Kappa, we present the Cohen’s Kappa agreement for all pairs of assessors who labeled at least 100 triplets in common.

The Fleiss’ Kappa agreement over *all* triplets was  $\kappa_f = 0.656$ , which is considered *substantial* agreement based on Landis and Koch [9]. In terms of Cohen’s Kappa agreement, there were 25 pairs of assessors with at least 100 triplets in common. Of these, 5 (20%) had *moderate* agreement ( $0.40 < \kappa_c \leq 0.60$ ), 16 (64%) had *substantial* agreement ( $0.60 < \kappa_c \leq 0.80$ ), and the remaining 4 (16%) had *perfect* agreement ( $0.80 < \kappa_c \leq 1.00$ ). Overall, assessor agreement on block-pairs was high. We view this as evidence that assessors did not have difficulty providing preferences for pairs of Web and vertical blocks.

## 7 Empirical Analysis and Validation

A desirable property of any evaluation measure is that it should correlate with user preference. We conducted a user study to test whether our metric (the  $K^*$  distance between  $\sigma_q$  and  $\sigma_q^*$ ) satisfies this criterion. Users were shown pairs of presentations side-by-side (along with the query and its description) and were asked to state a preference (“left is better”, “right is better”). We assumed that assessors would have difficulty deciding between two bad presentations. Therefore, to reduce cognitive load, we also included a “both are bad” option. Our hypothesis is that our metric will agree with the stated preference. Significance was tested using a sign test, where the null hypothesis is that the metric selects the preferred presentation randomly with equal probability.

Conducting this analysis requires a method for selecting pairs of presentations to show assessors. One alternative is to sample pairs uniformly from the set of all presentations. However, we were particularly interested in pairs of presentations from *specific* regions of the metric space. For example, is the metric correlated with user preference when one presentation is presumably high-quality (close to the reference) and the other is low-quality (far from the reference). Is it correlated when *both* presentations are presumably high-quality or when *both*

are low-quality? To investigate these questions, we sampled presentation-pairs using a binning approach. For each query, presentations were divided into three bins: a high-quality bin ( $\mathcal{H}$ ), a medium-quality bin ( $\mathcal{M}$ ), and a low-quality bin ( $\mathcal{L}$ ). The binning was done based on the metric value. The metric distribution is such that this produces bins where  $|\mathcal{H}| < |\mathcal{M}| < |\mathcal{L}|$ . The  $\mathcal{H}$  bin is the smallest and contains those presentations that are nearest to  $\sigma_q^*$ . The  $\mathcal{L}$  bin is the largest and contains those presentations that are furthest from  $\sigma_q^*$ . For each query, we sampled 4 presentation-pairs from each bin-combination ( $\mathcal{H}\text{-}\mathcal{H}$ ,  $\mathcal{H}\text{-}\mathcal{M}$ ,  $\mathcal{H}\text{-}\mathcal{L}$ ,  $\mathcal{M}\text{-}\mathcal{M}$ ,  $\mathcal{M}\text{-}\mathcal{L}$ , and  $\mathcal{L}\text{-}\mathcal{L}$ ) and collected 4 judgements per presentation-pair. This resulted in 1,728 presentation-pairs and 6,912 judgements. For this analysis, we also used Amazon’s Mechanical Turk.

## 7.1 Results

Assessor agreement on presentation-pairs was  $\kappa_f = 0.216$ , which is considered *fair* agreement [9]. Of all 1,728 presentation-pairs, only 1,151 (67%) had a majority preference of *at least* 3/4 and only 462 (27%) had a perfect 4/4 majority preference. It is perhaps not surprising that assessor agreement was lower on presentation-pairs than on block-pairs. Agreement on presentation-pairs requires that assessors make similar assumptions about the cost of different types of errors: a false-positive (displaying a non-relevant vertical), a false-negative (suppressing a relevant vertical), and a ranking error (displaying a relevant vertical in the wrong position). Assessors may require more instruction in order to improve agreement on presentation-pairs. Alternatively, more than 4 assessors may be required to see greater convergence.

Given this low level of inter-assessor agreement, rather than focus on the metric’s agreement with each individual preference, we focus on its agreement with the *majority* preference. We present results for two levels of majority preference: a majority preference of 3/4 or greater and a perfect (4/4) majority preference. These results are presented in Table 1. The “pairs” column shows the number of presentation pairs for which the level of majority preference was observed. The “% agreement” column shows the percentage of these pairs for which the metric agreed with the majority preference.

The metric’s agreement with the majority preference was 67% on pairs where at least 3/4 assessors preferred the same presentation and 73% on pairs where all (4/4) assessors preferred the same presentation (both significant at the  $p < 0.005$  level). Agreement with each individual preference (not in Table 1) was 60% (also significant at the  $p < 0.005$  level).

One important trend worth noting is that the metric’s agreement with the majority preference was higher on pairs where there was greater consensus between assessors. Overall, the metric’s agreement with the majority preference was higher on presentation-pairs that had a perfect (4/4) majority preference than on pairs that had a (3/4) majority preference or greater. This is a positive result if we primarily care about pairs in which one presentation was strongly preferred over the other.

**Table 1.** Metric agreement with majority preference. Significance is denoted by † and ‡ at the  $p < 0.05$  and  $p < 0.005$  level, respectively.

bins	majority preference	pairs	% agreement
all	3/4 preference	1151	67.07%‡
$\mathcal{H}\text{-}\mathcal{H}$	3/4 or greater	164	60.37%‡
$\mathcal{H}\text{-}\mathcal{M}$	3/4 or greater	210	81.90%‡
$\mathcal{H}\text{-}\mathcal{L}$	3/4 or greater	204	84.31%‡
$\mathcal{M}\text{-}\mathcal{M}$	3/4 or greater	184	57.61%†
$\mathcal{M}\text{-}\mathcal{L}$	3/4 or greater	187	50.80%
$\mathcal{L}\text{-}\mathcal{L}$	3/4 or greater	202	63.37%‡
all	4/4	462	72.51%‡
$\mathcal{H}\text{-}\mathcal{H}$	4/4	47	65.96%†
$\mathcal{H}\text{-}\mathcal{M}$	4/4	95	87.37%‡
$\mathcal{H}\text{-}\mathcal{L}$	4/4	97	91.75%‡
$\mathcal{M}\text{-}\mathcal{M}$	4/4	75	58.67%
$\mathcal{M}\text{-}\mathcal{L}$	4/4	71	54.93%
$\mathcal{L}\text{-}\mathcal{L}$	4/4	77	63.64%†

A similar trend was also observed across bin-combinations. The metric’s agreement with the majority was the highest on  $\mathcal{H}\text{-}\mathcal{M}$  and  $\mathcal{H}\text{-}\mathcal{L}$  pairs (82-92%). These were also the bin-combinations with the highest inter-assessor agreement ( $\kappa_f = 0.290$  for  $\mathcal{H}\text{-}\mathcal{M}$  and  $\kappa_f = 0.303$  for  $\mathcal{H}\text{-}\mathcal{L}$ )<sup>2</sup>. This means that, on average,  $\sigma_q^*$  was good. Assessors strongly preferred presentations close to  $\sigma_q^*$  over presentations far from  $\sigma_q^*$  in terms of  $K^*$ .

The metric was less predictive for  $\mathcal{H}\text{-}\mathcal{H}$  pairs (60-66%). However, inter-assessor agreement on these pairs was also low ( $\kappa_f = 0.066$ , which is almost random). It turns out that most  $\mathcal{H}\text{-}\mathcal{H}$  pairs had identical top-ranked blocks. This is because the  $\mathcal{H}$  bin corresponds to those presentations closest to  $\sigma_q^*$  based on  $K^*$ , which focuses on discordant pairs in the top ranks. About half of all  $\mathcal{H}\text{-}\mathcal{H}$  pairs had the same top 3 blocks and *all* pairs had the same top 2 blocks. The low inter-assessor agreement may be explained by users primarily focusing on the top results, perhaps rarely scrolling down to see the results below the “fold”. Alternatively, it may be that assessors have a hard time distinguishing between good presentations. Further experiments are required to determine the exact cause of disagreement. The metric was also less predictive for  $\mathcal{M}\text{-}\mathcal{M}$ ,  $\mathcal{M}\text{-}\mathcal{L}$ , and  $\mathcal{L}\text{-}\mathcal{L}$  pairs. Again, inter-assessor agreement was also lower for pairs in these bin-combinations ( $\kappa_f = 0.216$ ,  $\kappa_f = 0.179$ , and  $\kappa_f = 0.237$ , respectively). Inter-assessor agreement (and the metric’s agreement with the majority preference) was lower when neither presentation was of high quality (close to  $\sigma_q^*$ ).

We examined the queries for which the metric’s agreement with the majority preference was the lowest. In some cases, assessors favored presentations with a particular vertical ranked high, but the vertical was not favored in the block-pair judgements (therefore, it was ranked low or suppressed in  $\sigma_q^*$ ). For example, for

<sup>2</sup> Inter-assessor agreement across bin-combinations is also reflected in column “pairs”.



the query “ihop nutritional facts”, assessors favored presentations with *images* ranked high. For the query “nikon coolpix”, assessors favored presentations with *shopping* ranked high. For the queries “san bruno fire”, “learn to play the banjo”, “miss universe 2010”, and “us open fight”, assessors favored presentations with *video* ranked high. All three verticals (*images*, *shopping*, and *video*) are visually appealing (i.e., their blocks include at least one image). Prior research found a click-through bias in favor of visually appealing verticals (e.g., *video*) [13]. It may be that this type of bias affected assessors more on presentation-pairs (i.e., where the vertical is embedded within less visually appealing results) than on block-pairs (where the vertical is shown in isolation). If accounting for such a bias is desired, then future work might consider incorporating more context into the block-pair assessment interface. One possibility could be to show each block embedded in the same position within the same set of results.

## 8 Conclusion

We described a new methodology for evaluating aggregated search results. The idea is to use preference judgements on block-pairs to derive a *reference* presentation for the query and then to evaluate alternative presentations based on their distance to the *reference*. The approach has several advantages. First, with only a relatively small number of assessments per query, we can evaluate *any* possible presentation of results. This is not only useful for evaluation, but may be useful for learning and optimization. Second, the approach is general. We used a particular interface for assessing block-pairs, a particular voting method for deriving the *reference*, and a particular rank-similarity metric for measuring distance from the *reference*. Future work may consider others. Third, we showed that reliable block-pair assessments can be collected from a pool of inexpensive assessors. Finally, we presented a user study to empirically validate our metric. Assessors were shown pairs of presentations and asked to state a preference. Overall, the metric’s agreement with the majority preference was in the 67-73% range. Agreement was in the 82-92% range on those pairs where there was greater consensus between assessors.

In terms of future work, some open questions remain. Assessor agreement on presentation-pairs was low. Further experiments are needed to understand why. It may be, for example, that users assign a different value to different types of errors (false positives, false negatives, ranking errors). Also, in some cases, assessors favored a particular vertical only when seen within the context of *other* results. There may be preferential biases that affect presentation-pair judgements more than block-pair judgements.

## Acknowledgments

This work was supported in part by the NSF grants IIS-0916553, IIS-0841275, and IIS-1017026 as well as a generous gift from Yahoo! through its Key Scientific

Challenges program. Any opinions, findings, conclusions, and recommendations expressed in this paper are the authors' and do not necessarily reflect those of the sponsors.

## References

1. Arguello, J., Diaz, F., Callan, J., Crespo, J.-F.: Sources of evidence for vertical selection. In: *SIGIR 2009*, pp. 315–322. ACM, New York (2009)
2. Arguello, J., Diaz, F., Paiement, J.-F.: Vertical selection in the presence of unlabeled verticals. In: *SIGIR 2010*, pp. 691–698. ACM, New York (2010)
3. Carterette, B., Bennett, P.N., Chickering, D.M., Dumais, S.T.: Here or there: preference judgments for relevance. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008*. LNCS, vol. 4956, pp. 16–27. Springer, Heidelberg (2008)
4. Cohen, J.: A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* 20(1), 37–46 (1960)
5. Diaz, F.: Integration of news content into web results. In: *WSDM 2009*, pp. 182–191. ACM, New York (2009)
6. Diaz, F., Arguello, J.: Adaptation of offline vertical selection predictions in the presence of user feedback. In: *SIGIR 2009*, pp. 323–330. ACM, New York (2009)
7. Fleiss, J.: Measuring nominal scale agreement among many raters. *Psychological Bulletin*. 76(5), 378–382 (1971)
8. Kumar, R., Vassilvitskii, S.: Generalized distances between rankings. In: *WWW 2010*, pp. 571–580. ACM, New York (2010)
9. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. *Biometrics* 33(1), 159–174 (1977)
10. Li, X., Wang, Y.-Y., Acero, A.: Learning query intent from regularized click graphs. In: *SIGIR 2008*, pp. 339–346. ACM, New York (2008)
11. Sanderson, M., Paramita, M.L., Clough, P., Kanoulas, E.: Do user preferences and evaluation measures line up? In: *SIGIR 2010*, pp. 555–562. ACM, New York (2010)
12. Schulze, M.: A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method. *Social Choice and Welfare* (July 2010)
13. Sushmita, S., Joho, H., Lalmas, M., Villa, R.: Factors affecting click-through behavior in aggregated search interfaces. In: *CIKM 2010*, pp. 519–528. ACM, New York (2010)
14. Thomas, P., Hawking, D.: Evaluation by comparing result sets in context. In: *CIKM 2006*, pp. 94–101. ACM, New York (2006)
15. Zhu, D., Carterette, B.: An analysis of assessor behavior in crowdsourced preference judgements. In: *SIGIR Workshop on Crowdsourcing for Search Evaluation*, pp. 21–26. ACM, New York (2010)

# Design and Implementation of Relevance Assessments Using Crowdsourcing

Omar Alonso<sup>1</sup> and Ricardo Baeza-Yates<sup>2</sup>

<sup>1</sup> Microsoft Corp., Mountain View, California, USA  
omar.alonso@microsoft.com

<sup>2</sup> Yahoo! Research, Barcelona, Spain  
rbaeza@acm.org

**Abstract.** In the last years crowdsourcing has emerged as a viable platform for conducting relevance assessments. The main reason behind this trend is that makes possible to conduct experiments extremely fast, with good results and at low cost. However, like in any experiment, there are several details that would make an experiment work or fail. To gather useful results, user interface guidelines, inter-agreement metrics, and justification analysis are important aspects of a successful crowdsourcing experiment. In this work we explore the design and execution of relevance judgments using Amazon Mechanical Turk as crowdsourcing platform, introducing a methodology for crowdsourcing relevance assessments and the results of a series of experiments using TREC 8 with a fixed budget. Our findings indicate that workers are as good as TREC experts, even providing detailed feedback for certain query-document pairs. We also explore the importance of document design and presentation when performing relevance assessment tasks. Finally, we show our methodology at work with several examples that are interesting in their own.

## 1 Introduction

In the world of the Web 2.0 and user generated content, one important sub-class of peer collaborative production is the phenomenon known as *crowdsourcing*. In crowdsourcing, potentially large jobs are broken into many small tasks that are then outsourced directly to individual workers via public solicitation. One of the best examples is Wikipedia, where each entry or part of an entry could be considered as a task being solicited. As in the later example, workers sometimes do it for free, motivated either because the work is fun or due to some form of social reward [1,12]. However, successful examples of volunteer crowdsourcing are difficult to replicate. As a result, crowdsourcing increasingly uses a financial compensation, usually as micro-payments of the order of a few cents per task. This is the model of Amazon Mechanical Turk (AMT<sup>1</sup>), where many tasks can be done quickly and cheaply.

AMT is currently used as a feasible alternative for conducting all kind of relevance experiments in information retrieval and related areas. The lower cost of

---

<sup>1</sup> [www.mturk.com](http://www.mturk.com)

running experiments in conjunction with the flexibility of the editorial approach at a larger scale, makes this approach very attractive for testing new ideas with a fast turnaround. In AMT workers choose from a list of jobs being offered, where the reward being offered per task and the number of tasks available for that request are indicated. Workers can click on a link to view a brief description or a preview of each task. The unit of work per task to be performed is called a HIT (Human Intelligence Task). Each HIT has an associated payment and an allotted completion time; workers can see sample HITs, along with the payment and time information, before choosing whether to work on them or not. After seeing the preview, workers can choose to accept the task, where optionally, a *qualification exam* must be passed to assign officially the task to them. Tasks are very diverse in size and nature, requiring from seconds to minutes to complete. On the other hand, the typical compensation ranges from one cent to less than a dollar per task and is usually correlated to the task complexity.

However, what is not clear is how exactly to implement an experiment. First, given a certain budget, how we spend it and how we design the tasks? That is, in our case, how many people evaluate how many queries looking at how many documents? Second, how the information for each relevance assessment task should be presented? What should be the right interaction? How can we collect the right user feedback, considering that relevance is a personal subjective decision? In this paper we explore these questions providing a methodology for crowdsourcing relevance assessments and its evaluation, giving guidelines to answer the questions above. In our analysis we consider *binary relevance assessments*. That is, after presenting a document to the user with some context, the possible outcome of the task is relevant or non-relevant. Ranked list relevance assessment is out of the scope of this work, but it is a matter of future research.

This paper is organized as follows. First, in Section 2 we present an overview of the related work in this area. Second, we describe our proposed methodology in Section 3. Then, we explain the experimental setup in Section 4 and discuss our experiments in Section 5. We end with some final remarks in Section 6.

## 2 Related Work

There is previous work on using crowdsourcing for IR and NLP. Alonso & Mizzaro [2] compared a single topic to TREC and found that workers were as good as the original assessors, and in some cases they were able to detect errors in the golden set. A similar work by Alonso *et al.* [3] in the context of INEX with a larger data set shows similar results. Kazai *et al.* [8] propose a method for gathering relevance assessments for collections of digital books and videos. Tang & Sanderson used crowdsourcing to evaluate user preferences on spatial diversity [14]. Grady and Lease focused their work in human factors for crowdsourcing assessments [7].

The research work by Snow *et al.* [13] shows the quality of workers in the context of four different NLP tasks, namely, affect recognition, word similarity, textual entailment, and event temporal ordering. Callison-Burch shows how AMT can be used for evaluating machine translation [6]. Mason & Watts [11]

recently found that increased financial incentives increase the quantity, but not the quality, of work performed by the workers.

### 3 Methodology

One of the most important aspects of performing evaluations using crowdsourcing is to design the experiment carefully. Our crowdsourcing based methodology for relevance assessments is based in three parameters that we later analyze:

- $P$  Number of people used for each evaluation task.
- $T$  Number of topics chosen for the relevance tasks in the target document collection.
- $D$  Number of documents per query that will be judged for relevance.

The other typical parameter is the compensation per HIT. However, as we already have three parameters, we decided to keep that constant. We do this for two reasons. First, as we already mentioned, quality does not really improve if we pay more [11]. Second, if quality would improve, we would not be able to compare relevance assessments done with different compensations.

#### 3.1 Data Preparation

The first step of the methodology is preparing the data, similar to any relevance assessment study:

- Select the document collection.
- Select the  $T$  topics (queries).
- For each topic, select  $D$  documents per topic. We will use an even number as it is better to have the same number of relevant and non-relevant documents for the case of binary relevance.
- Select the number of workers  $P$  that will judge each topic/document pair. We use always an odd number to have a majority vote in most cases. If we have ties, an additional relevance assessment is made (we use this) or we can use a non-binary relevance measure. Also, topics should be assigned to workers randomly, so any possible bias is eliminated.

#### 3.2 Interface Design

This is the most important part of the AMT experiment design: how to ask the right questions in the best possible way. At first, this looks pretty straightforward, but in practice the outcome can be disastrous if this is performed in an ad-hoc manner. The first step is to follow standard guidelines for survey and questionnaire design [5] in conjunction with usability techniques.

The second step is to provide clear instructions for the experiment. In TREC, relevance assessments are performed by hired editors and instructions are very technical. The original instructions have four pages [15] and it was not possible to use them as is. In AMT, one cannot make any assumptions about the population so it is better to use plain English to indicate the task and avoid jargon.

We created a template, based on the TREC guidelines, that presents the instructions along with the web form for performing the task. The form contains a document with a close question (binary relevance; yes/no) and an open-ended question for justifying the selection. A back-end process reads the *qrel* file and topic data, instantiates the template variables accordingly and produces a HIT for that particular query-document pair.

As part of the experiment setup, we automatically generate metadata so that users can identify each task on the AMT website. Like in any market place we compete with other requesters for workers to get our experiment done. A clear title, description, and keywords allow potential workers to find experiments and preview the content before accepting tasks. In terms of keywords, we use a common set for all experiments (*relevance*, *news articles*, *search*, *TREC*) and then add specific terms depending on a given run. For example, in experiment E2 we use: *behavioral genetics*, *osteoporosis*, *Ireland*, *peace talks*.

### 3.3 Filtering the Workers

A possible filter for selecting good workers is to use the *approval rate*. The approval rate is a metric provided by AMT that measures the overall rating of each worker. However, using very high approval rates decreases the worker population available and implies a longer time to do the evaluation.

An alternative is to use qualifications to control which workers can perform certain HITs. A *qualification test*, is a set of questions similar to a HIT that a worker must answer when requesting the qualification. A qualification test is a much better quality filter but also involves more development cycles. In the case of relevance evaluation is somewhat difficult to test “relevance”. What we propose is to generate questions about the topics so workers can get familiar with the content before performing the tasks, even if they search online for a particular answer. In our case, the qualification test has ten multiple choice questions with 10 points value each. The goal of the qualification test is to eliminate lazy workers and workers that have bad performance for our specific task.

Another approach instead of qualification tests, is to interleave assignments where we already know the correct answer, so it is easy to detect workers that are randomly selecting the answers. This technique, some times called *honey pots*, is useful for checking if the workers are paying attention to the task. For example, when testing a topic one could include a document or web page that is completely un-related to the question and expect the worker to answer correctly. If they failed to pass, then there is indication that they are not following instructions correctly or just doing spamming.

AMT provides features for blocking workers from experiments and rejecting individual assignments. That said, it is easier to blame workers for bad answers when probably the interface was confusing or instructions not clear. In the next section will show that an iterative approach (like when designing user interfaces) that incorporates feedback earlier in the process is a good approach.

### 3.4 Scheduling the Tasks

It is known that there is a delicate balance between the compensation and the filtering step, with regards to the time that the experiment will take. A low compensation and/or a strict filtering procedure would drastically reduce the number of interested workers and hence will significantly increase the time length of the experiment. In fact, the distribution of number of tasks versus completion time is a power law as we show later.

One solution is to split even more long tasks and submit them in parallel. This has several advantages. First, the waiting time will decrease even though the total time spent in the tasks may not. Second, the overall time spent may also decrease because shorter tasks usually attract more workers. So in summary, as common sense suggests, it is better to have many small tasks in the system than one very large task. In our case the smallest task is to judge the relevance for one single document.

Regarding the experiments schedule, it is better to submit shorter tasks first, such that any important implicit or explicit feedback coming from the experiment can be used to re-design larger experiments. This is also helpful for debugging the experiment in the long run.

## 4 Experimental Setup

As ground truth for relevance assessments we use TREC-8, using the LA Times and FBIS sub-collections. TREC-8 has  $T = 50$  different topics. To cover all possible topics, we can use for example  $D = 10$  documents per topic and  $P = 5$  workers per document. Paying \$0.04 cents for each assessment we would need to spend \$100. However we want to study a larger space of assessments that can be done with the same amount of money.

We measure the agreement level between the raters using Cohen's kappa ( $\kappa$ ). As pointed out by many researchers,  $\kappa$  is a very conservative measure of agreement and is not perfect. Because we have many raters and the number of them varies, as well as we do not know the number of TREC raters for each topic, we average the answers of the workers as the value of the crowd.

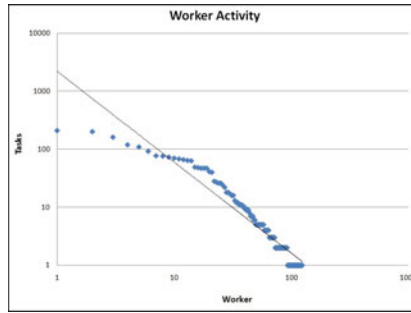
For the analysis, instead of trying all possible combinations, we covered all topics with different workers. In this way, each experiment is completely independent. Before running our seven experiments, we built an experimental test case and run it with a small number of documents and users. The goal was to make sure that workers understood the task and how long it took to complete.

We use an incremental approach for running our experiments (shown in table [II](#)). We increase the number of topics in every experiment until we covered most of the topics. We also keep some parameter constant, such that we can cover more variables combinations. In our case we do that with the number of documents and workers. In terms of cost per assignment (one query-document pair), we came up with \$0.04 cents. The rationale was \$0.02 cents per the answer (binary) plus \$0.02 cents per comment/feedback.

We tried to have one single experiment running on the system at all time. However, we noticed that as the experiments grew larger, the completion time

**Table 1.** Breakdown of the seven experiments

Exp	#T	# D	# P	Topics	Total cost
E1	1	4	1	401	\$0.16
E2	3	4	3	402, 403, 404	\$1.44
E3	5	6	3	405, 407, 408, 410, 411	\$3.60
E4	7	6	5	412, 413, 414, 415, 417, 418, 419	\$8.40
E5	10	8	5	420, 421, 422, 423, 424, 425, 426, 427, 428, 430	\$16.00
E6	11	8	5	406, 429, 433, 435, 445, 437, 438, 439, 442, 444	\$17.60
E7	11	10	7	437, 438, 439, 442, 444, 440, 441, 446, 448, 449, 450	\$33.60

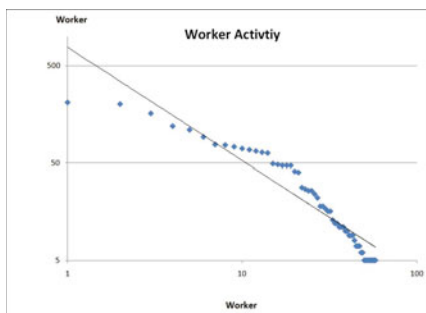
**Fig. 1.** Workers versus number of tasks

was taking longer. Starting with E5, we decided to split experiments in batches so they could finish in a reasonable amount of time. This shows that having smaller tasks is better. Another effect of this approach is to avoid *worker fatigue* in a single experiment and this allows some degree of parallelism.

## 5 Results and Discussion

For all experiments, there were 97 unique workers. The number of unique workers that completed 5 or less tasks was 60 and the number of unique workers that completed 1 task was 23. Figure 1 shows a graph of workers versus tasks. As we can see, this resembles a power law distribution, as expected. Figure 2 shows a similar graph but using workers that have 5 or more tasks. However, in both cases we can appreciate two power laws. One up to twenty workers with a much more flat exponent and one for more than twenty workers that is much more biased. This double power law behavior also appears in other cases such as word frequencies or number of links in web pages. This could imply that there are two types of workers, being the first group the ones that profit more from the AMT system. Figure 3 shows the number of unique workers in each experiment. Clearly this number decreases when a qualification exam is used and then increases but never to the level of the non-qualification exam case. This is due to the size of the work force in each case.





**Fig. 2.** Workers with 5 or more tasks versus number of tasks



**Fig. 3.** Unique workers vs. workers required

Table 2 shows the transaction details from AMT for all experiments. As we can see, for all experiments with no qualification test (E1-E4) we used a very high approval rate. For those with qualification test, we lowered a bit the approval rate but had our own test. Experiments with no qualification test tend to go faster compared with the first one with test in place (E5). This is expected as some workers may not feel taking the test or others fail. However, once a number of workers have passed the test, the other experiments tend to go faster (E6-E7). Another reason is trust. By then, workers know that we pay the work on time and in case we rejected the work, a clear explanation was given. This shows that the *word of mouth* effect works as in any other markets.

Table 3 shows more information about assignments and workers. Relevance evaluation is very subjective, so in principle we don't like to reject work because there is disagreement with a particular answer. We do reject when there is a clear indication of a robot doing tasks very fast or when the worker is choosing answers at random. After a number of rejections in E4, we decided to turn the qualification test a prerequisite for the rest of the experiments. Still, E5 had a higher number of rejections. This is possible due to the fact that even if a user passes a qualification test, she/he may chose answers at random.

To visualize the agreement between the workers and the original TREC assessments, in Figure 4 we show how the judgment converges to the correct value. Here we can see that in the relevant case, the majority always agrees with the

**Table 2.** Transaction details of every experiment

Exp	Approval rate	Qual. test	Completed time	Launched
E1	98%	No	8 min	Sunday AM
E2	98%	No	6 min	Sunday PM
E3	98%	No	5hrs, 31min	Sunday AM
E4	98%	No	4days, 2hrs, 45min	Friday AM
E5 batch1	98%	60%	8days, 4hrs, 40min	Thursday PM
E5 batch2	98%	60%	6days, 5hrs, 45min	Friday AM
E6 batch1	95%	60%	4days, 3hrs, 2min	Friday PM
E6 batch2	95%	60%	1day, 5hrs, 23min	Tuesday AM
E7 batch1	96%	70%	2days, 11hrs, 34min	Friday PM
E7 batch2	96%	70%	1day, 4hrs, 29min	Monday PM

**Table 3.** Details about assignments and workers per experiment

Exp	# workers	# approved	# rejected	# answers	# comments
E1	2	4	0	4	2
E2	11	53	1	53	44
E3	26	89	1	89	78
E4	28	181	28	181	141
E5 batch1	9	160	40	158	160
E5 batch2	8	200	0	199	200
E6 batch1	6	195	0	194	195
E6 batch2	9	235	0	235	235
E7 batch1	25	420	0	419	413
E7 batch2	19	420	0	419	420

**Table 4.** Inter agreement level between TREC and workers

Exp	Agreement level	Avg. topic difficulty
E1	0.00 (chance)	3.60
E2	0.66 (substantial)	3.13
E3	0.53 (moderate)	3.00
E4	0.39 (fair)	2.51
E5 batch1	0.25 (fair)	2.60
E5 batch2	0.25 (fair)	2.36
E6 batch1	0.21 (fair)	2.73
E6 batch2	0.32 (fair)	2.72
E7 batch1	0.71 (substantial)	2.08
E7 batch2	0.41 (moderate)	2.56

correct result. Nevertheless, the disagreement increases as is natural when the number of people involved in the decision grows. On the other hand, the majority does not agree in the case of non-relevant documents, showing that non-relevant cases are more difficult to judge. Hence, in practice, if the assessment seems to be non-relevant, additional workers should be included (say two more). In both cases, using a qualification exam clearly improves the judgments.



**Fig. 4.** Average R and NR judgments depending on the number of workers

After we finished running all batches we performed a short experiment that asked workers to rate on a scale 1 to 5 (1=easy, 5=very difficult) the topics. In Table 4 we show the values of  $\kappa$  for all experiments along with the average topic difficulty. Figure 5 shows that there is an inverse correlation between agreement and topic difficulty with the exception of two experiments. Notice that these values also depend on the money spent on each case. Normalizing by the cost we get the red dots in the same figure, where the inverse correlation is more clear. Choosing the number of workers to 5 in any experiment tends to be a good practice.

### 5.1 Presentation and Readability

One important factor is the effect of the user interface in the quality of the relevance assessments. To study that, we compared two different interfaces. One that helped the users by highlighting the terms of the query in the text and another one that just showed the plain text. The data preparation for this experiment consisted on taking the original document and producing two identical versions: plain and highlighting. The plain version has the visual effect of a continuous line. The highlighted version contains the topic title (up to 3 terms) highlighted in black with a yellow background. Figure 6 shows the results of this experiment.

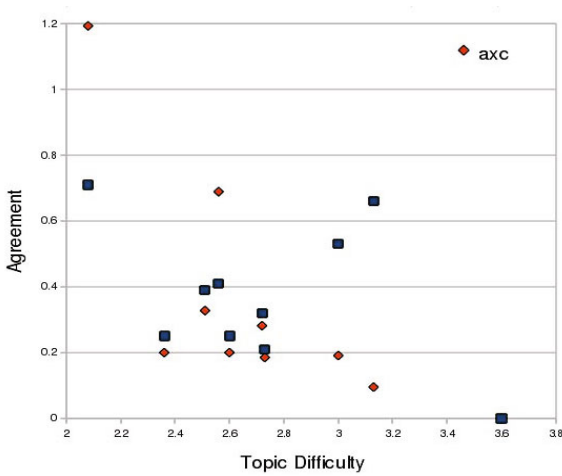


Fig. 5. Agreement vs. Topic Difficulty

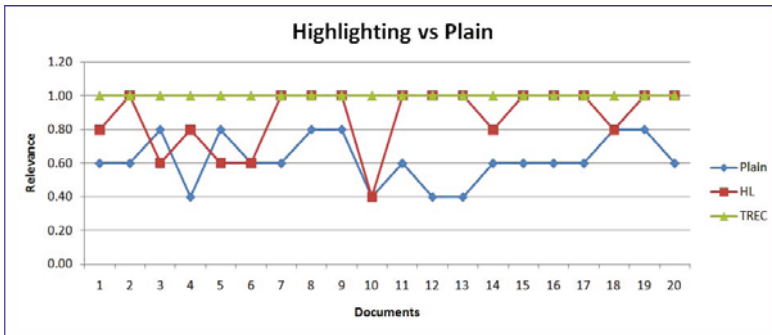
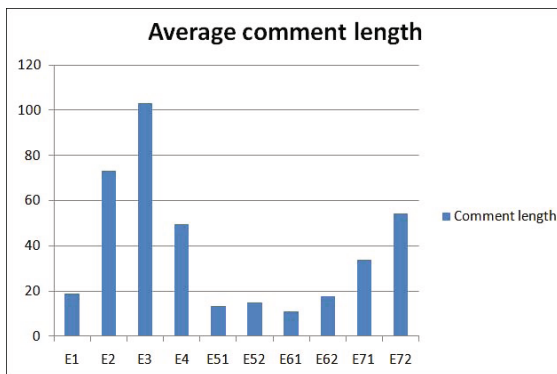


Fig. 6. Relevance votes on highlighting vs plain versions of the same documents

In the figure we show a set of relevant documents. For each document there is the TREC vote (1), and the number of votes for the highlighted and plain version. With the exception of two cases, highlighting does contribute to higher relevance judgments compared to plain. This tends to suggest that generalist workers may rely on a good document presentation when assessing relevance. In this particular experiment the number of documents is not that significant, however the results indicate that the presence of keywords (in our case highlighted) impact assessment [9].

### 5.2 Feedback Analysis

One way to get user feedback is to ask an optional open-ended question at the end of the task. Table 3 shows that the number of comments per experiment increases as more HITs are in the system. In experiments E1-E3, we made comments optional and found the feedback to be very useful. We also noticed that by asking



**Fig. 7.** Average comment length for all the experiments

workers to write a short explanation, we can not only gain more data but also detect spammers. By looking at some feedback we can observe that, in certain cases, a binary scale may not be suitable and a graded version should be applied.

Figure 7 shows the average comment length in characters for all experiments. As the number of documents (and experiments) goes up, the average length tends to go up as well. However, there is a dip starting in E4. The reason is that we wanted all workers to produce comments so we adjusted the guidelines and made it mandatory. Unfortunately, workers simply answered “relevant” or “not relevant” to make sure they get the money for the task. To prove that, we did re-launch E5 (the lowest with average 13.16) and changed the instructions so feedback was optional but with a bonus pay (\$0.01 cent per comment) if the content was good. The average comment length was 426.47, a clear indication that the bonus technique works for this kind of situation.

## 6 Concluding Remarks

We presented a methodology for crowdsourcing relevance assessments that consists on dividing a large document set into a series of smaller experiments and executing them separately according to a well-design template. We start with a smaller set and later tune documents, topics, and workers as parameters while keeping the same cost per assignments across all experiments. We have demonstrated the benefit of our approach by evaluating TREC-8 with a budget of just \$100.

Quality control is an important part of the experiment and it should be applied across different levels not just workers. As presented, quality of the instructions and document presentation have impact on the results. It is possible to detect bad workers but at the same time, the requester may have a bad reputation among workers. Our experience and findings show that the bulk of the experiment design should be on the the user interface and instructions. We showed in a small experiment that document presentation can have effects on relevance and readability, so it is not a matter of just uploading in some ad-hoc format and let the workers do the job. The user interface should make their tasks easier not

more difficult. As workers go through the experiments, diversity of topics in a single run can help avoid stalling the experiment due to lack of interest.

Our results together with previous results reinforce the advantages of crowdsourcing, in particular in the case of relevance assessments, which usually are not difficult, but are tedious and large in volume. This even works in non-English languages although experiments will take longer [4]. Overall, when it is possible to use social rewards, such as harnessing intrinsic motivation [10], the quality of the work will be good. If this is not possible, we should pay as little as possible, assuming that a large enough crowd exists to make up for the diminished quantity of individual output the low pay would imply. In other words, paying more may get the work done faster, but not better.

As future work we would like to study this dimension, compensating for whole topics or other possible aggregation of relevance assessments. Another avenue of work is to test the methodology with a ranked list to evaluate search engine search results. We also plan to continue working on document presentation and improving the overall user experience for getting better results.

## References

1. von Ahn, L.: Games with a purpose. *IEEE Computer* 39(6), 92–94 (2006)
2. Alonso, O., Mizzaro, S.: Can we get rid of TREC Assessors? Using Mechanical Turk for Relevance Assessment. In: *SIGIR Workshop Future of IR Evaluation (2009)*
3. Alonso, O., Schenkel, R., Theobald, M.: Crowdsourcing Assessments for XML Ranked Retrieval. In: *32 ECIR, Milton Keynes, UK (2010)*
4. Alonso, O., Baeza-Yates, R.: An Analysis of Crowdsourcing Relevance Assessments in Spanish. In: *CERI 2010, Madrid, Spain (2010)*
5. Bradburn, N., Sudman, S., Wansink, B.: *Asking Questions: The Definitive Guide to Questionnaire Design*. Josey-Bass (2004)
6. Callison-Burch, C.: Fast, Cheap, and Creative: Evaluating Translation Quality Using Amazon’s Mechanical Turk. In: *Proceedings of EMNLP (2009)*
7. Grady, C., Lease, M.: Crowdsourcing Document Relevance Assessment with Mechanical Turk. In: *NAACL HLT Workshop on Creating Speech and Language Data with Amazons Mechanical Turk (2010)*
8. Kazai, G., Milic-Frayling, N., Costello, J.: Towards Methods for the Collective Gathering and Quality Control of Relevance Assessments. In: *32 SIGIR (2009)*
9. Kinney, K., Huffman, S., Zhai, J.: How Evaluator Domain Expertise Affects Search Result Relevance Judgments. In: *17 CIKM (2008)*
10. Malone, T.W., Laubacher, R., Dellarocas, C.: *Harnessing Crowds: Mapping the Genome of Collective Intelligence*. MIT Press, Cambridge (2009)
11. Mason, W., Watts, D.: Financial Incentives and the ‘Performance of Crowds’. In: *HCOMP Workshop at KDD, Paris, France (2009)*
12. Nov, O., Naaman, M., Ye, C.: What Drives Content Tagging: The Case of Photos on Flickr. In: *CHI, Florence, Italy (2008)*
13. Snow, R., O’ Connor, B., Jurafsky, D., Ng, A.Y.: Cheap and Fast - But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks. In: *EMNLP (2008)*
14. Tang, J., Sanderson, M.: Evaluation and User Preference Study on Spatial Diversity. In: *32 ECIR, Milton Keynes, UK (2010)*
15. Voorhees, E.: Personal communication (2009)

# In Search of Quality in Crowdsourcing for Search Engine Evaluation

Gabriella Kazai

Microsoft Research Cambridge  
v-gabkaz@microsoft.com

**Abstract.** Crowdsourcing is increasingly looked upon as a feasible alternative to traditional methods of gathering relevance labels for the evaluation of search engines, offering a solution to the scalability problem that hinders traditional approaches. However, crowdsourcing raises a range of questions regarding the quality of the resulting data. What indeed can be said about the quality of the data that is contributed by anonymous workers who are only paid cents for their efforts? Can higher pay guarantee better quality? Do better qualified workers produce higher quality labels? In this paper, we investigate these and similar questions via a series of controlled crowdsourcing experiments where we vary pay, required effort and worker qualifications and observe their effects on the resulting label quality, measured based on agreement with a gold set.

**Keywords:** IR evaluation, relevance data gathering, crowdsourcing.

## 1 Introduction

The evaluation of a search engine's effectiveness typically requires sets of relevance labels, indicating the relevance of search results to a set of queries. However, the gathering of relevance labels is reportedly a time consuming and expensive process that is usually carried out by hired trained experts. For example, TREC<sup>1</sup> employs retired intelligence analysts as assessors.

Recently, crowdsourcing<sup>2</sup> has emerged as a feasible alternative to gather relevance data for the evaluation of search engines<sup>2,11,4</sup>. It promises to offer a solution to the scalability problem that hinders the traditional approaches, where either the size of the test collection or limitations on time and other resources are proving increasingly prohibitive. Crowdsourcing is an open call for contributions from members of the crowd to solve a problem or carry out human intelligence tasks (HITs), often in exchange for micro-payments, social recognition, or entertainment value. Crowdsourcing platforms, such as CrowdFlower<sup>3</sup> or Amazon's Mechanical Turk (AMT)<sup>3</sup> service, allow for anyone to create and publish HITs, and gather vast quantities of data from a large population within a short space of time and at a relatively low cost. However, crowdsourcing, and

---

<sup>1</sup> <http://trec.nist.gov/>

<sup>2</sup> <http://crowdflower.com/>

<sup>3</sup> <https://www.mturk.com/>

more specifically crowdsourcing when monetary incentives are involved, is a solution with its own set of challenges [8,10]. Indeed, crowdsourcing has been widely criticized for its mixed quality output. Marsden, for example, argues that 90% of crowdsourcing contributions are rubbish [9]. On the other hand, several studies in relevance data collection concluded that crowdsourcing leads to reliable labels [14]. At the same time, works such as [13,8] provide evidence of cheating and random behavior among members of the crowd.

Clearly, the gathering of useful data requires not only technical capabilities, but also sound experimental design. This is especially important in crowdsourcing where the interplay of the various motivations and incentives affects the quality of the collected data [11,10]. The usefulness or the quality of the data will of course depend on the goals and the context of a given crowdsourcing experiment. For example, in IR evaluation, relevance labels contributed by non-experts have been shown to lead to different conclusions when comparing system performance [3]. Thus, when crowdsourcing relevance labels for the evaluation of IR systems, quality may be defined in terms of agreement with expert judges.

With the aim to obtain insights that can guide the design of crowdsourcing experiments, in this paper, we explore the relationship between the quality of the crowdsourced relevance labels and properties of the task design: the monetary reward it offers, the effort it demands, and the qualifications it requires. Specifically, we aim to answer the following research questions:

- Does quality pay? Or rather, can higher pay guarantee better quality results?
- Do more qualified workers produce higher quality labels?
- Are increased levels of required effort detrimental to output quality?

To answer these questions, we designed several batches of HITs on AMT where we varied pay, required effort and worker qualifications, and observed their effects on the quality of the gathered labels. Our analysis reveals intricate relationships between the examined properties, highlighting the need to study crowdsourcing experiments as complex ecosystems with implications for the design of HITs as well as of spam filters.

Next, we motivate our work and describe the data used in the experiments. Section 3 details the HIT design. Results and findings are given in section 4.

## 2 Experiment Setup and Data

For our experiments we chose the problem of building a test collection for the evaluation of focused retrieval approaches. This challenge is faced by the INEX Book Track, which has thus far struggled to meet this need by relying on its participants' voluntary efforts alone due to the scale of the problem [7]: "The estimated effort required of a participant of the INEX 2008 Book Track to judge a single topic was to spend 95 minutes a day for 33.3 days". Motivated by this need to scale up the Cranfield method for constructing test collections, our work explores the possibility of employing crowdsourcing methods to replace or compliment traditional modes of gathering relevance data. In particular, our



```

<inex_topic track=book task=book-retrieval/book-ad-hoc topic_id=57>
<title> Titanic </title>
<description> I am interested in real life factual as well as artistic accounts of the
sinking of the Titanic.
</description>
<narrative>
<task> The story of the Titanic has been made popular with the success of the movie Titanic.
I would like to find out more about this tragic event and get a better feeling about
the effect it had on the people of the time.
</task>
<infneed> I am interested in historical information about the sinking of the Titanic, both
witness accounts and historians' take on the events. I am also interested in poems and
other artistic expressions that relate to this tragedy. I am however not interested
in the critiques of such arts.
</infneed>
</narrative>
</inex_topic>

```

**Fig. 1.** Topic from the INEX 2008 Book Track test set

goal is to study how reliable crowdsourcing is in terms of its output quality, what factors impact on the quality and how these influence each other.

Among the range of tasks investigated by the INEX Book Track, we chose the Focused Book Search (FBS) task, where users expect to be pointed directly to relevant book parts. In this task, systems return ranked lists of book parts (e.g., pages) estimated relevant to a topic. To evaluate the task, INEX collects relevance labels for a subset of the book pages retrieved by participating systems.

The data used in our experiments consists of the books, search topics, and relevance judgments of the INEX 2008 Book Track test collection. The corpus contains 50,239 out-of-copyright books (17 million pages), totaling 400GB. From the total of 86 available topics, we selected 8 topics (ID: 27, 31, 37, 39, 51, 57, 60 and 63) based on the number of available judgments in the INEX test collection. Figure 1 shows one of the topics. For our 8 topics, we have 470 judged books, of which 149 are relevant, and 4,490 judged pages, of which 1,109 are relevant. From these, we randomly picked 100 pages per topic ensuring a 40-60% ratio of relevant/irrelevant labels. These labels form the gold set for our AMT experiments, which are described next.

### 3 HIT Design

We designed several batches of HITs on AMT with the following objectives:

- To test the effects of various HIT properties (task parameters) on the quality of the resulting labels: pay, effort, and worker qualification.
- To gather relevance labels for the book pages in our gold set based on which we can measure the quality of a crowdsourcing experiment in terms of the agreement with the gold set (accuracy). For each book page, the following options were offered in the HIT: ‘Relevant’, ‘Not relevant’, ‘Broken link’, ‘Don’t know’. A free-text comment field was also provided. In addition, to reduce the attractiveness of the HITs to random clicking, we used a challenge-response test or ‘captcha’ asking workers to enter the last word printed on the scanned book page.
- To collect data about the workers’ perceptions of the task, e.g., if they thought it too difficult or if it paid too little, etc.

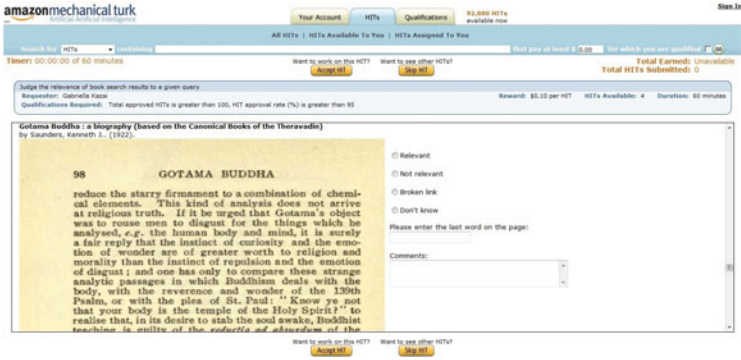


Fig. 2. Example HIT from the experiments on AMT

Figure 2 shows an example HIT. In order to display the book page images inside a HIT, we made use a web service call of the Book Search System<sup>4</sup> provided by the INEX Book Track.

### 3.1 Task Parameters

We experimented with the following task parameters:

- Pay: We experimented with two pay levels, paying \$0.10 or \$0.25 per HIT.
- Worker qualifications: This parameter was controlled through the use of AMT’s worker selection criteria. We used two settings: 1) no required qualifications (‘noQ’), and 2) restricting to workers with over 95% HIT approval rate with over 100 approved HITs (‘yesQ’).
- Effort: Effort was controlled via the number of book pages that workers were required to judge in a given HIT: 1) HITs containing 5 book pages and 2) HITs with 10 pages.

It is clear that some of these parameters are directly or indirectly related to each other. For example, one may expect higher pay for tasks that require more effort. We aim to uncover such relationships and explore their influences on each other and on the quality of the crowdsourced labels.

With this aim, we created 8 batches of HITs based on the different combinations of task parameter values, see Table 1. The naming of a batch combines the three task parameter values: pay (‘10’ or ‘25’ cents), qualifications (‘noQ’ or ‘yesQ’), and effort (‘5’ or ‘10’ book pages per HIT). Each batch contained 800 book pages, 100 per topic. This resulted in 160 HITs when 5 pages were included per HIT, and 80 HITs when effort was 10 pages per HIT. We requested 3 workers per HIT, yielding either 240 or 480 assignments per batch and giving us 2,400 relevance labels per batch. The total number of collected labels is thus 19,200. Note that these numbers exclude rejected work. The totals including rejected HITs is shown in brackets in Table 1.

<sup>4</sup> <http://www.booksearch.org.uk/>

**Table 1.** Batches of HITs with different task parameter settings

Batch	Pay	Qualif.	Effort	HITs	Assignments	Judged pages	Cost
10-noQ-5	\$0.10	no	5 pages	160	480 (608)	2,400 (3,040)	\$52.80
10-noQ-10	\$0.10	no	10 pages	80	240 (460)	2,400 (4,600)	\$26.40
10-yesQ-5	\$0.10	yes	5 pages	160	480 (722)	2,400 (3,610)	\$52.80
10-yesQ-10	\$0.10	yes	10 pages	80	240 (358)	2,400 (3,580)	\$26.40
25-noQ-5	\$0.25	no	5 pages	160	480 (592)	2,400 (2,960)	\$132.00
25-noQ-10	\$0.25	no	10 pages	80	240 (299)	2,400 (2,990)	\$66.00
25-yesQ-5	\$0.25	yes	5 pages	160	480 (480)	2,400 (2,400)	\$132.00
25-yesQ-10	\$0.25	yes	10 pages	80	240 (304)	2,400 (3,040)	\$66.00

### 3.2 Workers’ Perception of the Task

With the aim to examine the use of self-reported measures as possible indicators of quality, we also gathered various feedback from the workers. For example, we may expect that more knowledgeable workers would produce higher quality labels [3]. Workers could provide feedback on the following aspects:

- Familiarity: We asked workers to rate their familiarity with the subject of the topic for which relevance labels were sought in a HIT. We used a 4 point scale (0-3) with ‘Minimal’ and ‘Extensive’ as end points. To encourage truthful answers, we emphasized that their answer would not affect their pay.
- Task difficulty: We collected workers’ opinions on the difficulty of the task, with rating options of ‘difficult’, ‘ok’, or ‘easy’.
- Interest in the task: We asked workers to indicate if the task was in their opinion ‘boring’, ‘ok’, or ‘interesting’.
- Pay: We solicited workers’ opinions if they thought they were being paid ‘too little’, ‘ok’, or ‘too much’, or if pay did not matter to them.

## 4 Results and Findings

In this section, we present our findings with the aim of answering our original research questions. Since all book pages included in the HITs have known labels, we use accuracy as a measure of crowdsourced label quality, calculated as the ratio of the total number of correct labels and the total number of labels in a given batch or subset. A relevance label submitted by a worker is correct if it matches the label (relevant/irrelevant) in our gold set for the given book page. We note that 45 of the 800 book pages are actually ‘missing’ from the corpus in the sense that their page image, fetched from the Book Search System, fails to load in the HIT. In the case of these known missing pages, we accept ‘Broken link’ as the correct answer.

Table 2 (rows 1-8) shows the calculated accuracy levels for the 8 batches over three filter sets: 1) all the collected data, including rejected work and unusable labels: when workers clicked ‘Don’t know’ or ‘Broken link’ or when no label was given (all data), 2) the subset that excludes rejected work (no spam), and 3) the fully cleaned subset that excludes rejected work and pages with unusable

**Table 2.** Number of unique workers, average time spent per book page, and accuracy of the crowdsourced relevance labels across the different batches or subsets, corresponding to different task parameter combinations

	Batch/Subset	All Gathered Labels			No Spam			Cleaned		
		#Wkrs	Time	Acc.	#Wkrs	Time	Acc.	#Wkrs	Time	Acc.
1.	10-noQ-5	70	42	59.74%	66	51	61.79%	65	48	66.38%
2.	10-noQ-10	69	26	34.98%	63	42	59.29%	61	40	67.26%
3.	10-yesQ-5	66	42	60.78%	62	58	62.62%	62	58	66.97%
4.	10-yesQ-10	35	42	59.22%	33	59	59.75%	33	58	62.90%
5.	25-noQ-5	71	51	52.03%	68	61	54.79%	66	59	57.48%
6.	25-noQ-10	58	41	52.34%	54	49	63.50%	54	48	72.56%
7.	25-yesQ-5	43	61	71.50%	43	61	71.50%	43	61	73.58%
8.	25-yesQ-10	54	33	67.04%	48	38	69.83%	48	38	74.01%
9.	\$.010	199	37	52.18%	186	52	60.86%	183	51	65.85%
10.	\$.025	197	46	60.22%	184	52	64.91%	182	51	69.36%
11.	\$.001	99	33	45.59%	92	50	59.52%	90	49	65.01%
12.	\$.002	130	42	60.30%	122	54	62.21%	121	53	66.67%
13.	\$.0025	105	37	59.75%	95	43	66.67%	95	42	73.31%
14.	\$.005	108	56	60.75%	105	61	63.15%	103	60	65.63%
15.	\$.010 noQ	121	32	44.83%	113	46	60.54%	110	44	66.81%
16.	\$.010 yesQ	90	42	60.00%	84	58	61.19%	84	58	64.93%
17.	\$.025 noQ	121	46	52.18%	114	55	59.15%	112	53	64.70%
18.	\$.025 yesQ	92	45	69.01%	86	49	70.67%	86	49	73.79%
19.	noQ	225	38	48.05%	211	51	59.84%	207	49	65.75%
20.	yesQ	155	43	63.88%	148	54	65.93%	148	54	69.40%
21.	5 Pages	225	48	60.50%	215	58	62.68%	212	56	66.14%
22.	10 Pages	189	34	51.60%	175	47	63.09%	173	46	69.15%

labels (cleaned). HITs were rejected if the worker completing the HIT filled in, on average over all their HITs, less than 30% of the captcha fields and spent less than 20 seconds judging a page.

As it can be seen, we obtain accuracy levels mostly in the region of 52-74%, with the exception of 10-noQ-10 resulting in only 35% agreement with the gold set labels (over all collected data). This is perhaps not surprising as this batch presents the ‘hardest deal’ where workers needed to judge 10 pages per HIT for only \$.010 payment, while this batch was also open to all workers (noQ). The best accuracy is 74% obtained for the 25-yesQ-10 batch (cleaned data), similar to inter-assessor agreement levels at TREC. It is worthwhile to note how the perception of quality changes depending on how the collected data is filtered. For example, batch 25-noQ-10 ranks 6th when quality is calculated over all collected data, but it ranks 3rd when rejected work is removed. In the next sections, we will report findings based on all three filter sets and attempt to explain the differences between them.

#### 4.1 Does Quality Pay?

Our first research question regards whether pay affects the quality of the collected relevance labels. To answer this question, we grouped the collected labels into two

bins based on the amount of pay per HIT, see rows 9-10 in Table 2. We see that increasing pay from \$0.10 to \$0.25 per HIT leads to improved label quality in all our data sets (all data, no spam, cleaned). A two sample t-test confirms that pay leads to a significant difference in accuracy per HIT in the unfiltered and no spam sets ( $p < 0.01$ , two-tailed). Unlike previous reports, e.g., [10], this finding suggests that pay does impact quality: encouraging better work. At the same time, we observe a reduced benefit to the increase in pay as we filter out spam and unusable labels: the increase in accuracy gained by paying more drops from 115% (all data) to 107% (no spam) and then to 105% (cleaned). So, what does this mean?

Looking at the differences between the three filter sets, we see that more unusable and spam labels are contributed when pay is lower: in total 4,293 incorrect labels (29% of all collected labels) are removed during our filtering from the \$0.10 set, while this is only 17% (1,990 incorrect labels) in the \$0.25 set. This is also reflected in the relative increase in accuracy between the unfiltered and the cleaned data sets: 126% improvement for the \$0.10 HITs while only 115% for the \$0.25 HITs. This advocates that pay also impacts quality indirectly, where higher pay leads to more HITs with usable labels and less spam. This is also supported by the average time that workers spent on judging a page: 37 seconds in the lower pay condition vs. 46 seconds in the higher pay batches (all data). Interestingly, after excluding rejected work, the average time is 52 seconds per page for both pay levels. However, as we will see later, time spent per page is influenced by a range of aspects, e.g., expertise, captcha, which complicate its use for indicating label quality.

Comparing quality between different pay per label sets (instead of pay per HIT), see rows 11-14 in Table 2 confirms the same trend: quality increases with pay. However, we can also observe evidence of a diminishing return effect, whereby the rate of increase in pay is matched by a slowing (or dropping) rate of increase in quality. This is especially clear on the cleaned data set, where accuracy first increases from 65% (\$0.01 per page) to 67% (\$0.02 per page) and then to 73% (\$0.25 per page) but then it drops back down to 66% (\$0.05 per page). The difference in accuracy across the three filter sets indicates that the lowest paid condition is most inducing of spamming. Interestingly, the best accuracy is achieved by the subset that has the second highest level of spam and unusable labels (\$0.025 per page). As very little (detected) spam or unusable labels are contributed in the highest paid condition, we may reason that the drop in accuracy is indeed due to the phenomena of diminishing return.

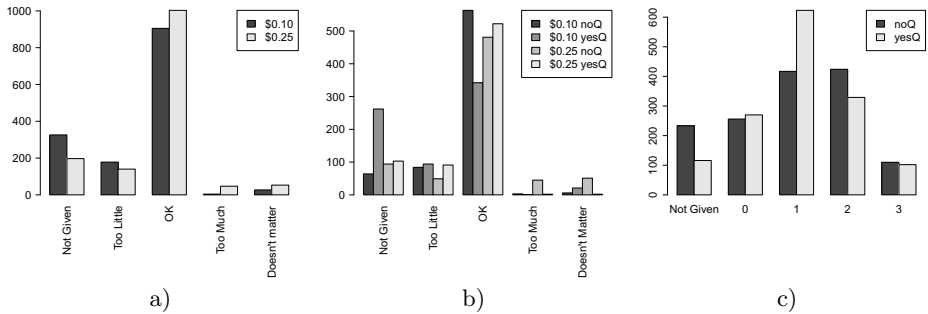
To investigate whether pay affects different groups of workers differently, we now focus on the subsets of labels contributed in the ‘noQ’ and ‘yesQ’ conditions within the two pay levels, see rows 15-18 in Table 2. We find differences in label quality for both qualified and non-qualified workers: accuracy of 60% (yesQ) and 45% (noQ) when paid \$0.10 per HIT vs. 69% (yesQ) and 52% (noQ) when receiving \$0.25 per HIT (all data). This suggests that pay effects both groups equally: higher pay encourages better work regardless of the AMT qualifications. However, after removing spam and unusable labels, we find that the level of pay shows no effect on the accuracy of non-qualified workers (around 66%). After

a more detailed look at the data, we found that this was in fact a result of labels contributed by possibly unethical workers who escaped our spam filter or by workers who mistook the task for OCR correction (instead of relevance assessment). We found that in the ‘\$0.25-noQ’ batches a higher number of HITs (115) were submitted by workers who filled in over 30% of captcha fields, but labeled most pages ( $> 80\%$ ) as relevant (a less likely label given our 40/60 ratio of relevant/irrelevant pages), compared with 5 HITs in the ‘\$0.10-noQ’ subset and 27 and 40 HITs in the ‘\$0.10-yesQ’ and ‘\$0.25-yesQ’ subsets, respectively. Of course, with an adapted spam filter, such HITs can again be removed from the cleaned data. Overall, thus, we conclude that higher pay induces two different behavior in workers: on the one hand it encourages better work, especially for qualified workers, but at the same time it also attracts more unethical workers, especially when workers are not pre-filtered.

Next, we analyze workers’ feedback about the amount of pay offered per HIT, see Figure 3a (no spam set). Unsurprisingly, we find that workers were more satisfied with higher pay, but we also see that the majority of the responses indicated that workers were content (‘pay is ok’) with the offered pay in both pay categories: 63% in the \$0.10 batches and 70% in the \$0.25 batches. This confirms that we estimated the minimum level of pay correctly at \$0.01 per page, but it also shows that workers accept a relatively wide spectrum of pay for the same work: from \$0.05 down to \$0.01 per judgment. Only 2% of the responses indicated that workers did not care about pay in the low paid batches, while this was 4% in the higher paid batches. Among the subset of HITs with ‘too much pay’ or ‘pay does not matter’ responses, workers indicated high (66% of responses) or moderate (31%) interest in the task. This ratio is very different for workers who selected ‘pay is ok’ as their feedback: only 20% of the responses expressed high interest and 71% indicated moderate interest. Oddly, when workers were unsatisfied with their pay (‘pay is too little’), 30% of the responses registered high interest, and 60% moderate interest. This highlights the duality of pay and interest, where pay becomes secondary when interest is high, e.g., as demonstrated by the ESP game [12]. A chi-square test also confirmed that pay and interest are significantly related ( $p < 0.01$ ).

When broken down by qualifications, we found that qualified workers were more disgruntled by lower pay than non-qualified workers: this is indicated by the drop in ‘pay is ok’ responses in the yesQ batches (48%) compared with 78% in the noQ batches, see Figure 3b. Thus it seems that qualified workers have higher expectations on pay, while non-qualified workers estimate the value of their work at a lower rate.

Looking at accuracy for the different feedback categories on pay, we find no difference between accuracy for the ‘pay is ok’ and ‘pay is too little’ sets: 70% for both. Accuracy drops slightly for ‘pay does not matter’ responses (63%) or when no feedback was given (60%). The lowest accuracy is obtained in the set where workers indicated that pay was ‘too much’ (40%). This highlights the possible use of these fields as weak signals to filter workers.



**Fig. 3.** Workers’ feedback on the amount of pay per HIT in the \$0.10 and \$0.25 pay-level subsets (a) and further broken down by worker qualifications (b). Workers’ familiarity level (0=‘Minimal’,3=‘Extensive’) in noQ and yesQ batches (c).

## 4.2 Does Worker Qualification Affect Quality?

To answer this question we looked at the quality of labels in two subsets of HITs: those that restricted participation by requiring workers to have over 95% acceptance rate and more than 100 HITs (yesQ), and those where no qualifications were required (noQ), see rows 19-20 in Table 2. We see that pre-selecting workers leads to better accuracy in all our filter sets, e.g., 48% for noQ vs. 64% for yesQ (all data). Moreover, a two sample t-test shows that qualification leads to a significant difference in accuracy per HIT for each of the three filter sets ( $p < 0.01$ , two-tailed). The differences between the reported accuracy levels between the unfiltered and filtered sets clearly show that non-qualified workers contribute more spam and unusable labels. However, in the cleaned data set we only observe a small benefit to pre-selecting workers: 66% for noQ vs. 69% for yesQ. This is of course a result of our filters which lead to 41% of the collected labels being thrown away from the noQ set (incl. 78% incorrect labels), compared with 33% in the yesQ set (incl. 47% incorrect labels). The difference between the percentages of incorrect labels in the discarded data suggests that different filtering methods may be more appropriate for different types of workers.

Looking at the reported levels of familiarity with the topic by qualified and non-qualified workers, we see that non-qualified workers tend to be more confident and report higher familiarity than qualified workers (not statistically significant), see Figure 3c.

When calculating accuracy for each familiarity rating, we find that the most self confident workers in fact have the lowest accuracy (45% on all data, 58% on no spam), while accuracy for lower levels of familiarity ranges between 52-61% (all data) or 63-67% (no spam). When broken down by worker qualifications, we see that it is those non-qualified workers who rated their knowledge on the topic highly who do worst (accuracy of 33% on all data and 55% on no spam), while qualified ‘expert’ workers achieve 60-61% accuracy (all data and no spam). In general, we see a negative correlation between self-reported expertise and accuracy for non-qualified workers (accuracy drops as familiarity increases: 56% (familiarity level 0), 55% (familiarity 1), 42% (familiarity 2) and 33% (familiarity 3) on all data),

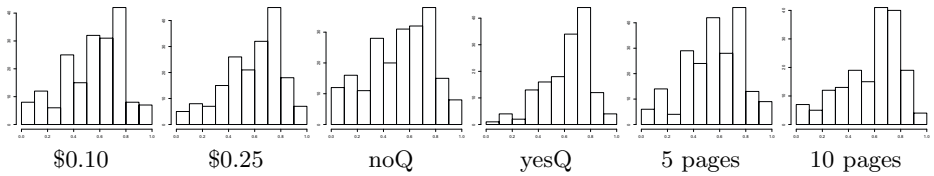


Fig. 4. Distribution of workers over accuracy (no spam set)

while for qualified workers accuracy varies less across familiarity levels (63-67% on all data and 65-70 on no spam (familiarity levels 0-2)). This indicates that more veteran workers on AMT are better at gaging their own level of expertise while new workers may be more prone to effects of satisficing [6]. It is interesting to note that best accuracy is obtained when workers reported only minimal knowledge of the topic (familiarity 0) for both qualified and non-qualified workers (on all our filter sets). This is somewhat counterintuitive as we would expect that more knowledge of a topic would lead to better accuracy [3]. A possible explanation is that these less confident workers may take more care in completing the task, while over confident workers may be prone to mistakes. This is especially true for non-qualified workers who may care less about the quality of their work as they do not yet have a track record to protect. This is also partly supported by the observation that more knowledgeable workers took on average less time to judge a page (no spam set): 41 seconds (familiarity of 2) vs. 53 seconds (familiarity of 0). However, it should be noted that the time alone cannot give a true picture of workers' commitment to the task, since workers claiming to have more extensive knowledge have in fact only filled in on average 55% of the captcha fields, compared with 72-88% in HITs where lower familiarity levels were reported.

### 4.3 Does Effort Affect Quality?

To answer this question, we grouped results from batches that asked workers to label 5 or 10 book pages, resp., see rows 21-22 in Table 2. Looking at the unfiltered data set, we may conclude that effort does matter and better results can be produced in conditions when workers are not overloaded (61% accuracy for HITs with 5 pages vs. 52% for HITs with 10 pages). Indeed, we find that effort leads to a significant difference in accuracy per HIT in the unfiltered set ( $p < 0.01$ , two-tailed, two sample t-test). However, accuracy in the cleaned set suggests the opposite (5 pages: 66% vs. 10 pages: 69%). That said, workers do seem to be less motivated to do well on tasks that require more effort and as a result the collected data contains more unusable and spam labels. This is corroborated by the finding that workers spent on average longer judging a page in the low effort HITs: 48 seconds vs. 34 seconds (all data).

The self reported difficulty revealed that non-qualified workers who found the task difficult performed the worst (35% all data, 50% no spam), compared with 48-50% (all data) and 60-61% (no spam) accuracy when the task was reported as not difficult. Qualified workers achieved a more consistent accuracy regardless whether they found the task easy or difficult (68-72%, no spam).



## 5 Conclusions

In this paper, in the context of crowdsourcing relevance labels, we investigated three basic parameters of crowdsourcing experiment design (pay, effort and worker qualifications) and their influence on the quality of the output, measured by accuracy. Unlike in [10], our findings show that pay does matter: higher pay encourages better work while lower pay results in increased levels of unusable and spam labels. Looking at pay per label (rather than per HIT), we found evidence of diminishing return where the rate of increase in quality flattens out or even drops as pay increases. This was partly due to the higher pay attracting more sophisticated unethical workers who escaped our simple filter. From this, it is clear that experiment designers need to find the right balance between too low pay that results in sloppy work and too high pay that attracts unethical workers. Estimating reward per unit of effort promises a suitable method for this. In addition, higher pay should also be balanced with better quality control mechanisms built into the design (e.g., pre-filtering workers, captcha, or training [8]) and more robust spam filters.

When comparing different groups of workers, we found that more qualified workers produce better quality work. This may be a consequence of the ‘reputation’ system in AMT, where these workers strive to maintain their qualification levels, e.g., HIT approval rate, as this allows them to have access to a wider range of tasks. However, quality cannot be guaranteed just by pre-filtering workers as more sophisticated unethical workers can easily build a false reputation. At the same time, there are plenty of ethical non-qualified workers who aim to build up their reputation and may thus be more diligent when completing a task. To take advantage of both groups requires different HIT designs to engage the right workers as well as adapted spam filters to process the output. Both pay and qualifications lead to significant differences in output quality.

With respect to effort, we found that while higher effort induces more spam, it also leads to slightly better quality after spam removal than low effort HITs, though this is not statistically significant.

Figure 4 shows the distribution of workers over agreement, for the various subsets of the gathered labels (no spam set), confirming the above observations.

In summary, our findings highlight a network of influences between the different task parameters and the output quality. We found that all investigated task parameters had influence on the output quality, both directly and indirectly, e.g., higher pay encouraging better work while also attracting more sophisticated spammers. We conclude that increasing pay, reducing effort, and introducing qualification requirements can all help in reducing undesirable behavior. However, due to the interplay of the parameters and their influence on each other, on the HIT design, and the output, each such decision needs to be balanced overall, e.g., increased pay may call for additional quality control elements. Our analysis of the collected data also revealed the need for a deeper understanding of the observed variables, e.g., time spent judging a page, to aid in the detection of sloppy or unethical workers. In addition, we found that self-reported data, e.g.,

familiarity, perceived difficulty of the task, and satisfaction with pay, can also help experimenters in filtering out sub-quality data.

Our future work will explore these relationships in more detail. We also plan to expand the set of task parameters to include factors such as clarity, emotion, aesthetics, pre-task qualification tests, seeding, etc. Our ultimate goal is to provide experimenters a framework to guide the design of their crowdsourcing tasks to maximize quality.

## Acknowledgments

Many thanks to Jaap Kamps for helpful comments on the camera ready version of this paper.

## References

1. Alonso, O., Mizzaro, S.: Can we get rid of TREC assessors? using Mechanical Turk for relevance assessment. In: Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation, pp. 557–566 (2009)
2. Alonso, O., Rose, D.E., Stewart, B.: Crowdsourcing for relevance evaluation. *SIGIR Forum* 42(2), 9–15 (2008)
3. Bailey, P., Craswell, N., Soboroff, I., Thomas, P., de Vries, A.P., Yilmaz, E.: Relevance assessment: are judges exchangeable and does it matter. In: *SIGIR 2008: Proceedings of the 31st Annual International ACM SIGIR Conference*, pp. 667–674 (2008)
4. Grady, C., Lease, M.: Crowdsourcing document relevance assessment with mechanical turk. In: Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, pp. 172–179 (2010)
5. Howe, J.: *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Crown Publishing Group, New York (2008)
6. Kapelner, A., Chandler, D.: Preventing satisficing in online surveys: A ‘kapcha’ to ensure higher quality data. In: *The World’s First Conference on the Future of Distributed Work (CrowdConf 2010)* (2010)
7. Kazai, G., Koolen, M., Doucet, A., Landoni, M.: Overview of the INEX 2009 book track. In: Geva, S., Kamps, J., Trotman, A. (eds.) *INEX 2009*. LNCS, vol. 6203, pp. 145–159. Springer, Heidelberg (2010)
8. Le, J., Edmonds, A., Hester, V., Biewald, L.: Ensuring quality in crowdsourced search relevance evaluation. In: *SIGIR Workshop on Crowdsourcing for Search Evaluation*, pp. 17–20 (2010)
9. Marsden, P.: Crowdsourcing. *Contagious Magazine* 18, 24–28 (2009)
10. Mason, W., Watts, D.J.: Financial incentives and the “performance of crowds”. In: *HCOMP 2009: Proceedings of the ACM SIGKDD Workshop on Human Computation*, pp. 77–85 (2009)
11. Quinn, A.J., Bederson, B.B.: A taxonomy of distributed human computation. Technical Report HCIL-2009-23, University of Maryland (2009)
12. von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2004, pp. 319–326 (2004)
13. Zhu, D., Carterette, B.: An analysis of assessor behavior in crowdsourced preference judgments. In: *SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation* (2010)

# Summarizing a Document Stream

Hiroya Takamura, Hikaru Yokono, and Manabu Okumura

Precision and Intelligence Laboratory,  
Tokyo Institute of Technology  
{takamura, oku}@pi.titech.ac.jp, yokono@lr.pi.titech.ac.jp

**Abstract.** We introduce the task of summarizing a stream of short documents on microblogs such as Twitter. On microblogs, thousands of short documents on a certain topic such as sports matches or TV dramas are posted by users. Noticeable characteristics of microblog data are that documents are often very highly redundant and aligned on timeline. There can be thousands of documents on one event in the topic. Two very similar documents will refer to two distinct events when the documents are temporally distant. We examine the microblog data to gain more understanding of those characteristics, and propose a summarization model for a stream of short documents on timeline, along with an approximate fast algorithm for generating summary. We empirically show that our model generates a good summary on the datasets of microblog documents on sports matches.

## 1 Introduction

Microblogs such as Twitter<sup>1</sup> have recently gained popularity. They are different from other conventional blogs in that entries of microblogs are very short, called tweets and 140-characters long in Twitter, and therefore are mainly used to describe what users are doing or how they are feeling at this very moment, while conventional blogs usually deal with more coarse units of time such as days or weeks. As a new source of information, microblogs have drawn a great deal of attention of the public.

One distinctive aspect of microblogs as a source of information is that numerous short documents on a single topic are posted by many users. A typical example of topics of our interest is sports matches on TV. While people are watching a sports match on TV, they often post short documents about the match on the microblogs. We call such a set of short documents on timeline, a *document stream*. The purpose of this paper is to propose a summarization model for document streams of microblogs. With such a technique, we would be able to learn what is going on with regard to the topic, or what people think about the topic. Their contents can be description of facts, applause, criticism, or sometimes serious opinions. Sports matches are not the only example that interests us. Another example will be microblogs on TV dramas.

---

<sup>1</sup> <http://twitter.com>

Video streaming service such as Ustream<sup>2</sup> would provide us with more potential application areas of the technique for document stream summarization.

In the document stream summarization, we need to take into consideration that short documents on microblogs are aligned on timeline. Let us take a soccer match as an example of topics. On microblogs, “good pass” at the 10th minute and “good pass” at the 85th minute should be two distinctive events in the match. However, there is often no apparent evidence of their distinctiveness, except posted time. We also need to be aware that documents on microblogs are not always posted at the instant that the event occurs; they are often posted with some delay. In this work, we take the extractive summarization approach [8], in which we choose several short documents from the data in order to generate a summary. Although the sentence extraction is often used in the standard text summarization task, the document extraction is sufficient in the current task since documents are very short in microblogs (not longer than 140 characters in Twitter) and each document usually conveys a simple piece of information. The remaining question is how to select documents. We will answer this question by proposing a new summarization method based on the  $p$ -median problem.

## 2 Numerous Short Documents on Timeline

In order to gain more understanding of characteristics of microblogs, we preliminarily collected and analyzed microblog data<sup>3</sup>. We choose a document stream on a soccer match, namely Japan vs. Hong Kong in East Asian Football Championship 2010, which Japan won with 3-0. We collected Japanese tweets (short documents) about this match from Twitter using Streaming API.

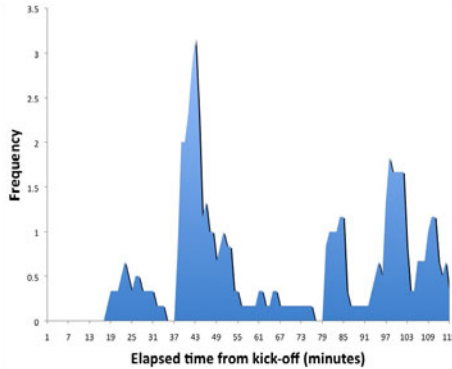
We first counted the frequency of the term “goal” at each time point in the match and obtained Figure 1. The curve in this figure was smoothed with the moving average method for the demonstration purpose.

The curve in Figure 1 has several peaks. We examined the tweets data and found out that they are caused by different events in the match. The peak at around the 25th minute was caused by tweets of users who were complaining about the lack of goals of Japan, which had shown poor performance in the previous scoreless match against China. The peaks at around the 40th, the 80th, the 100th minutes were caused respectively by the first goal of a Japanese striker Keiji Tamada, the goal of Marcus Tulio Tanaka, and the second goal of Keiji Tamada. Tweets on these goal events are often similar to each other. This observation suggests that when we measure the similarity of two documents on timeline during the summarization process, we have to take the difference of the posted times into account so that we can distinguish documents on one event from those on another similar event at another time point.

We also point out that users do not always post tweets right after the event (i.e., goal). For example, the tweets containing “goal” start to increase at the 37th minute, and are frequently posted until around the 55th minute. We need to be able to recognize that those tweets mention a single event.

<sup>2</sup> <http://www.ustream.tv>

<sup>3</sup> This dataset is different from the one used in the empirical evaluation in Section 6.



**Fig. 1.** Frequency of the messages containing the term “goal” in tweets on Japan vs. Hong Kong on Feb. 11, 2010. The curve was smoothed with the moving average method.

These two observations show two opposite characteristics of microblogs; (i) very similar documents can mention different events if they are temporally distant, (ii) documents on a single topic can be posted with some temporal delay. We conjecture that these characteristics are not unique to microblogs on soccer, but are shared by microblogs with other topics as well. We would like to construct a summarization model that can handle these two opposite characteristics.

### 3 Summarization Model Based on Facility Location Problem

We first introduce a text summarization model proposed by Takamura and Okumura [15], which we will use as the basis of our method. Their model is based on the facility location problems [3]. These problems are applicable to practical issues of determining, for example, hospital location or school location, where the facility is desired to be accessible to the customers in the area. The  $p$ -median problem is a facility location problem which has the cardinality constraints that the number of selected facility sites is upperbounded by constant  $p$ . We consider here that customer locations are also potential facility sites. In their model, a summary is generated such that every sentence (short document, in our case) in the given data is assigned to and represented by one selected sentence (short document) as much as possible. In the following, we will explain their method as a method for selecting documents, though their method is originally based on sentence extraction.

Let us denote by  $e_{ij}$  the extent to which document  $d_j$  is inferred by document  $d_i$ . We call this score  $e_{ij}$  the *inter-documental coefficient*. In the previous work using this model [15],  $e_{ij}$  was defined as  $e_{ij} = |d_i \cap d_j|/|d_j|$ , where  $d_i$  is regarded as the set of content words contained in the document, and therefore  $d_i \cap d_j$  represents the intersection of  $d_i$  and  $d_j$ .

If we denote by  $z_{ij}$  the variable which is 1 if  $d_j$  is assigned to  $d_i$ , otherwise 0, then the score of this whole summary is going to be  $\sum_{i,j} e_{ij} z_{ij}$ , which will be maximized. We next have to impose the cardinality constraint on this maximization so that we can obtain a summary of length  $p$  or shorter, measured by the number of documents. Let  $x_i$  denote a variable which is 1 if  $d_i$  is selected, 0 otherwise. The cardinality constraint is then represented as  $\sum_i x_i \leq p$ . The  $p$ -median summarization model is formalized as follows:

$$\begin{aligned} \max. \quad & \sum_{i,j} e_{ij} z_{ij} \\ \text{s.t.} \quad & z_{ij} \leq x_i; \quad \forall i, j, \end{aligned} \tag{1}$$

$$\sum_i x_i \leq p, \tag{2}$$

$$\sum_i z_{ij} = 1; \quad \forall j, \tag{3}$$

$$z_{ii} = x_i; \quad \forall i, \tag{4}$$

$$x_i \in \{0, 1\}; \quad \forall i, \tag{5}$$

$$z_{ij} \in \{0, 1\}; \quad \forall i, j. \tag{6}$$

**(1)** guarantees that any document to which another document is assigned is in a summary. **(2)** is the cardinality constraint. **(3)** guarantees that every document is assigned to a document, and **(4)** means that any selected document is assigned to itself. The integrality constraint on  $z_{ij}$  **(6)** is automatically satisfied in the problem above. Although this optimization problem is NP-hard and intractable in general, if the problem size is not so large, we can still find the optimal solution by means of the branch-and-bound method **(4)**.

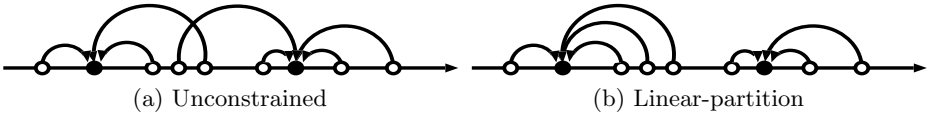
## 4 Summarizing a Document Stream

On the basis of the text summarization model in Section **(3)**, we will propose a summarization model for a document stream consisting of short documents on timeline.

We focus on the two characteristics of microblogs observed in Section **(2)**: (i) very similar documents can mention different events if they are temporally distant, (ii) documents can be posted with some temporal delay. We explain two distinct ideas for incorporating these two characteristics. The first idea is the modification of coefficients  $e_{ij}$ . The second one is the linear-partition constraint on document assignment.

### 4.1 Modification of Coefficients $e_{ij}$

If two documents are temporally distant, we would like them to have a small coefficient  $e_{ij}$  even if their contents are similar. Thus we multiply  $e_{ij}$  with a decreasing function of difference in time as a decay factor. Although there are many possible decreasing functions, we use the function  $: 0.5^{|t(d_i) - t(d_j)|/\beta}$ , where  $t(d)$  is the time  $d$  was created at (measured by seconds), and  $\beta$  is a positive constant. Exploration for good decreasing functions would be left as future work.



**Fig. 2.** Document assignments. The horizontal straight arrow represents the timeline. Black circles represent selected documents. White circles represent unselected documents. The curved arrows represent the assignment relations.

The coefficient of two temporally-distant documents is going to be very small due to the decay factor. The similar documents posted with a slight delay will still have a high coefficient. Note that too large  $\beta$  results in giving a large coefficient to temporally-distant similar documents, and that too small  $\beta$  results in giving a small coefficient to temporally-close similar documents. The final definition of the coefficient is as follows :

$$e_{ij}^{\text{time}} = e_{ij} 0.5^{|t(d_i) - t(d_j)|/\beta}. \quad (7)$$

The proposed model will tend to select longer documents, which cover more words. We also test a variant of the proposed model where we penalize longer documents by multiplying  $e_{ij}^{\text{time}}$  with length-penalty factor  $1/c_i$ .

## 4.2 Linear-Partition Constraint on Document Assignment

Another idea for handling the characteristics of microblogs is to impose, what we call, *the linear-partition constraint* on document assignment in the  $p$ -median summarization model in Section 3.

Let us begin with defining the unconstrained assignment and the linear-partition assignment. In our task, the documents are aligned on timeline. The  $p$ -median summarization model allows a document to be assigned to any document on timeline. Therefore the arrows indicating document assignment can cross each other as in Figure 2 (a). In the figure, the horizontal straight arrow represents the timeline. Black circles represent selected documents. White circles represent unselected documents, which are assigned to one of the black circles. We call it *the unconstrained assignment*. In contrast, when the arrows do not cross each other, we call it *the assignment with the linear-partition constraint*, or simply *the linear-partition assignment*, illustrated by Figure 2 (b).

We propose to impose the linear-partition constraint on the  $p$ -median text summarization model; we use the linear-partition assignment. In this new model, two temporally-distant documents are unlikely to be linked unless all the documents between the two documents are similar. Delay in posted time is disregarded as long as similar documents are present in-between. Temporally-close documents are likely to form a cluster. The linear-partition constraint is incorporated into the  $p$ -median text summarization model by adding the following two constraints to the previous optimization model:

$$\begin{aligned} z_{ij} &\leq z_{ik}; \forall i, j, k (j \leq k \leq i), \\ z_{ij} &\leq z_{ik}; \forall i, j, k (i \leq k \leq j). \end{aligned}$$

Although the search space has been reduced by adding these constraints, this problem is not easy to deal with, since the number of constraints is on the order of  $O(N^3)$ , where  $N$  is the number of documents. To our knowledge, there is no algorithm that finds the exact solution within a reasonable computational time.

We should be aware that this model with the linear-partition constraint can capture the characteristics of microblogs only when we can assume that there are no parallel multiple topics. The model should work in the current task of interest. In the situation where there are parallel multiple topics, we would need to use the unconstrained assignment of documents. We do not insist that the actual relations between documents are linear-partitioning. We are merely attempting to generate a good summary by assuming the linear-partition. Our purpose is not to predict the true (linear-partition or unconstrained) assignments of sentences.

Adding the linear-partition constraint has the advantage that we can think of an approximate fast algorithm described in Section 4.3.

### 4.3 Approximate Algorithm for Summarization with the Linear-Partition Constraint

We introduce an approximate algorithm for solving the  $p$ -median problem with the linear-partition constraint. The algorithm iterates the document assignment to medians and the median update, as well-known  $k$ -means clustering does. The pseudo-code of the algorithm can be described as follows. In the algorithm, the

#### Approximate algorithm for $p$ -median on a line

randomly pick initial  $p$  medians  $d_{m_1}, \dots, d_{m_p}$

such that  $\forall i \leq j, t(d_{m_i}) \leq t(d_{m_j})$

**while** there is any change in medians

**for**  $l = 1$  **to**  $p$

    reassign  $d_{m_{l+1}}, \dots, d_{m_{l+1}-1}$  to either  $d_{m_l}$  or  $d_{m_{l+1}}$

**end for**

**for**  $l = 1$  **to**  $p$

    find the best median out of documents assigned to  $d_{m_l}$

**end for**

**end while**

local reassignment of documents can be performed by first finding

$$h_{max} = \operatorname{argmax}_{h: m_l \leq h < m_{l+1}} \sum_{j=m_l}^h e_{m_l j} + \sum_{j=h+1}^{m_{l+1}} e_{m_{l+1} j}$$

and then assigning  $d_{h_{max}}$  and the documents on the left side of  $d_{h_{max}}$  to  $d_{m_l}$ , and those on the right side to  $d_{m_{l+1}}$ . This local maximization can be performed fast by means of a simple dynamic programming, since the value of the objective function at  $h+1$  can be obtained by adding  $e_{m_l h+1}$  to and subtracting  $e_{m_{l+1} h+1}$  from the value of the objective function at  $h$ .

Finding the best median out of documents assigned to  $d_{m_l}$  can be performed simply by calculating the objective function for each median candidate.



It is easy to show that each iteration increases the value of the objective function. Therefore, this algorithm at least finds the local maximum.

## 5 Related Work

There are a number of pieces of work in which integer linear programming is used for text summarization [9,6] or sentence compression [2].

Sharifi et al. [13] attempted to summarize microblogs. Their attempt was, given a query term, to find frequent linguistic patterns that contain the query term. The resulting summary is usually shorter than a single document of microblogs. Hence their objective is completely different from the objective in this paper, which is to generate a summary consisting of multiple documents.

Microblogs are currently being studied extensively [12,11]. O'Connor et al. [10] presented a search application for Twitter. Along with a list of tweets containing the query, this application returns frequent significant terms and tweets containing each term. Swan and Jensen [14] used temporal information to find significant terms, and applied their method on Topic Detection and Tracking (TDT) corpus [1]. Both of their objectives are different from ours.

The Topic Detection and Tracking (TDT) task [1] is related to the present task, in that documents are aligned on timeline in both tasks. Our technique in this paper will be applied to the documents that are found through TDT task. The current task can also be regarded as an instance of clustering data stream [7]. Methods developed in the field of clustering data stream would be promising in the current task and should be explored in the future work.

## 6 Experiments

### 6.1 Evaluation of Document Stream Summarization

We present an automatic evaluation measure for this task of short documents summarization on timeline. We assume that reference summaries consisting of time-stamped short documents are available. For the conventional text summarization, ROUGE [5] is very well-known and has been used by many researchers. It calculates the recall indicating how many word types in the reference summary are covered by the generated summary. However, ROUGE will not work as an evaluation measure for our task, because word types that appear at distant time points should be regarded as different word types. In our modified ROUGE, word types of a short document in the reference summary are regarded as covered only if the word type is contained in a selected short document and the difference of the time stamps of the two documents (i.e., one in the reference summary, the other in the generated summary) is within a constant. We set this constant to 10 minutes in our experiments.

We used only content words (i.e., nouns, verbs, adjectives) in the calculation of ROUGE scores. We also removed some Japanese stopwords such as *suru*, *iru*, *naru*, which roughly mean *do*, *be*, *become* respectively.

**Table 1.** Statistics on tweet data and gold standard data

opponent	fixture	tweet data		gold standard	
		# of documents (tweets) before filtering	after filtering	# of documents	# of words
Cameroon	June 14, 2010	61666	2814	26	439
Netherlands	June 19, 2010	56976	3219	29	580
Denmark	June 24, 2010	93336	5196	41	690

## 6.2 Data Preparation

We prepared datasets of document streams consisting of tweets on soccer matches. This dataset consists of Japanese tweets (short documents) mentioning the three matches of Japan in group stage of 2010 FIFA World Cup : Japan vs. Cameroon, Japan vs. Netherlands, Japan vs. Denmark. We used Streaming API provided by Twitter to collect tweets with the relevant hashtags : #soccer, #jfa, #wc2010, #jfa2010, #daihyo, #2010wc. Most users supposedly posted the tweets while they were watching TV broadcast of the matches.

After examining the data, we realized that many of the tweets are not appropriate for summary parts, since they are often simply shouts, just names of Japanese players, or some text fragments that do not make sense by themselves. We thus extracted the tweets that contain both names of players and football terms<sup>4</sup>. The numbers of tweets before and after this filtering for each match are shown in Table 1. Although the filtering greatly reduces the number of documents in Table 1, selecting a few documents from those is still a hard problem. For the matches against Cameroon, Netherlands and Denmark, there are several thousand documents left after filtering, in which case the exact solution of the optimization problem for our model is unobtainable. In order to make the exact solution obtainable for the purpose of comparative experiments, we also created smaller datasets by random sampling (details in Section 6.3). We also created the gold standard data consisting of reference summaries for evaluation purpose. Each document in the gold standard data is also very short, usually consisting of 1 or 2 sentences. The statistics of the gold standard data is also shown in Table 1. We used the text reports on the internet<sup>5</sup> as reference summaries.

Word segmentation and part-of-speech tagging were performed on all the documents including tweet data and gold standard data. We used MeCab<sup>6</sup>.

## 6.3 Experimental Setting

In the calculation of  $e_{ij}$  and  $e_{ij}^{\text{time}}$ , each sentence is represented as a set of content words. We use base forms of content words (nouns, verbs, and adjectives) that

<sup>4</sup> The lists of player names and football terms were manually created with the help of the webpage of Japan national football team (<http://samuraiblue.jp/>) operated by Japan Football Association.

<sup>5</sup> <http://mainichi.jp/enta/sports/soccer/10fwc/graph/2010061402,2010061901,2010062403>

<sup>6</sup> <http://mecab.sourceforge.net/>

are not stopwords. The maximum summary lengths were set to be the same as the lengths of the reference summaries : 26, 29 and 41 documents respectively for the matches against Cameroon, Netherlands and Denmark.

**The comparative experiment on the smaller datasets.** The first experiment is conducted on the smaller datasets each containing randomly sampled 500 documents<sup>7</sup>. This random sampling was performed 10 times for each setting. The result is the average of these 10 trials. Since this dataset is small, we were able to the branch-and-bound method implemented in ILOG CPLEX version 11.1 to obtain the exact solution to the  $p$ -median problem. We compared the exact solutions with the approximate solutions. The length-penalty factor was not employed in this experiment.

**The experiment on the larger datasets.** The second experiment is conducted on the larger datasets of 2010 FIFA World Cup, each of which consists of 2814, 3219 and 5196 documents, respectively in Table 1. Since it is practically impossible to obtain the exact solution of the  $p$ -median problem of this size, we use only the approximate algorithm (Section 4.3). The algorithm is applicable only to the summarization model with the linear-partition constraint. We test both the model with the length-penalty factor and the model without it. For this experiment, we compare the proposed method with the following baselines.

- *random*: this baseline method simply selects  $p$  documents randomly. Since the result depends on the generated random values, we executed this method 100 times to obtain 100 summaries and computed the average value of the ROUGE scores of those summaries. Although this method is governed by randomness, if there are similar documents in the data, this method is likely to select one out of those documents.
- *equal*: this baseline method sorts the documents in the order of their created times, and selects  $p$  documents so that the intervals of the selected documents are going to be equal. The intervals are measured by the number of documents, not by time.
- *interval*: this baseline method first splits the timeline into intervals of equal length (10 seconds, in our experiments) and selects  $p$  intervals that have the largest number of documents. For each of these intervals, the method selects the document with the largest cosine similarity (calculated by bag-of-words vectors) to the union of the documents in the interval.

## 6.4 Results

We first report the result of the experiment on the smaller datasets each consisting of 800 documents. We test both the branch-and-bound method that yields the exact solution with or without the linear-partition constraint, and the approximate algorithm. The result is summarized in Table 2. We can see that the

<sup>7</sup> For 600 or more documents, the branch-and-bound method did not converge after 4 hours for some settings.

**Table 2.** ROUGE scores of  $p$ -median summarization model on the smaller datasets

vs.	summary length	$p$ -median					
		unconstrained				linear-partition	
		<i>exact</i>				<i>exact</i>	<i>approx.</i>
		$e_{ij}$	$e_{ij}^{\text{time}}$			$e_{ij}$	$e_{ij}$
–	$\beta = 300$	$\beta = 600$	$\beta = 900$	–	–		
Cameroon	26	0.222	0.247	0.253	0.254	0.218	0.232
Netherlands	29	0.232	0.282	0.279	0.279	0.241	0.251
Denmark	41	0.265	0.300	0.305	0.299	0.296	0.274
average	32.0	0.240	0.276	0.279	0.277	0.252	0.252

**Table 3.** ROUGE scores of  $p$ -median summarization model with the linear-partition document assignment and the baseline methods on the larger datasets of Japan’s group stage of 2010 FIFA World Cup. The approximate algorithm was used for solving  $p$ -median problem. ‘w/o pnlty’ means the proposed model without the length-penalty factor, while ‘w/ pnlty’ means the one with the length-penalty factor.

vs.	summary length	baselines			$p$ -median	
		<i>random</i>	<i>equal</i>	<i>interval</i>	w/o pnlty	w/ pnlty
		<i>approx.</i>				
Cameroon	26	0.145	0.173	0.178	0.236	0.200
Netherlands	29	0.156	0.176	0.156	0.309	0.225
Denmark	41	0.188	0.214	0.245	0.314	0.270
average	32.0	0.163	0.188	0.193	0.286	0.232

modification of coefficient improves summarization performance for each match. The ROUGE score of  $e_{ij}^{\text{time}}$  is stable in the range of  $\beta = 300$  to 900. The linear-partition constraint also improves summarization performance without regard to the choice of the algorithms (exact or approximate). This result shows that both of the proposed models (i.e., the modification of coefficients, the linear-partition constraint) work well in the summarization of document stream on timeline.

The average computational times were 44.07 seconds for the exact algorithm with the unconstrained assignment, and 1341.79 seconds for the exact algorithm with the linear-partition constraint, while that of the approximate algorithm was less than a second.

We next report the result of the experiment on the larger datasets of 2010 FIFA World Cup, each of which consists of 2814, 3219 and 5196 documents, respectively in Table 1. The approximate algorithm (Section 4.3), imposing the linear-partition constraint, was used for solving  $p$ -median problem. The result is summarized in Table 3. As the table shows, the  $p$ -median summarization model with the linear-partition constraint outperformed the three baselines in terms of the ROUGE score for each of the three matches. Also when we impose the length-penalty, the proposed model is still superior to the baselines. We note that the summaries generated by the proposed model with the length-penalty was shorter than those by the baselines on average.

**Table 4.** An example of summary (Japan vs. Denmark). Tweets are originally in Japanese, and translated into English by the authors. We set the length limit to 10 tweets due to space limitation, and the length-penalty was imposed. Elapsed time from kick-off includes half time.

elapsed time	selected tweets
12m34s	Endo got the yellow card!!!
18m26s	Honda's beautiful freekick!
30m41s	Another freekick goal! This time, by Endo.
54m48s	The 1st half was over. We are 2 points ahead.
91m48s	Okazaki came on as sub of Matsui.
99m28s	Goalie Kawashima made a great save on penalty, but conceded a goal right after that.
116m43s	Okazaki, GOAL! Honda, good pass! Japan is ahead, 3-1.

As an example, we show the summary generated by the proposed method with the length-penalty. Due to the space limitation, we pick the match between Japan and Denmark and set the length limit to 10 tweets.

## 7 Conclusion

We introduced the task of summarizing document stream of microblogs such as Twitter. Through data analysis, we confirmed that temporally distant documents may refer to distinct events even if they are similar in terms of the words used in those documents, and also that users sometimes post documents with some delay in time. We proposed a summarization model that takes these characteristics of microblogs into account. We also proposed a fast approximate algorithm for generating a summary out of the proposed model. Through experiments on microblog data on soccer matches, we showed that the proposed model improves the quality of summaries.

As future work, we will have to conduct more detailed evaluation on the proposed method including experiments on diverse datasets and manual evaluation of the generated summaries. The method would become more sophisticated with the exploration of other inter-documental coefficients. We would also like to apply our method for other types of topics such as TV dramas or Ustream broadcasting. We filtered tweets using players' names and football terms. Those keywords have to be automatically acquired when this method is applied to many other domains. We are working on keyword extraction from web contents such as Wikipedia, from which Japanese players' names are also available<sup>8</sup>.

Another interesting direction is the evaluative summarization of document streams of microblogs. Currently, we focus on factual summarization. However, microblog users often express opinions or emotions on events. The modification to the inter-documental coefficients that gives larger weights on evaluative documents would make an evaluative summary of document streams.

<sup>8</sup> [http://en.wikipedia.org/wiki/Japan\\_national\\_football\\_team](http://en.wikipedia.org/wiki/Japan_national_football_team)

## References

1. Allan, J., Carbonell, J., Doddington, G., Yamron, J., Yang, Y., Amherst, U., Umass, J.A.: Topic detection and tracking pilot study. In: DARPA Broadcast News Transcription and Understanding Workshop, pp. 194–218 (1998)
2. Clarke, J., Lapata, M.: Global inference for sentence compression: An integer linear programming approach. *J. of Artificial Intelligence Research* 31, 399–429 (2008)
3. Drezner, Z., Hamacher, H.W. (eds.): *Facility Location: Applications and Theory*. Springer, Heidelberg (2004)
4. Hromkovič, J.: *Algorithmics for Hard Problems*. Springer, Heidelberg (2003)
5. Lin, C.: ROUGE: a package for automatic evaluation of summaries. In: *Proceedings of the Workshop on Text Summarization Branches Out*, pp. 74–81 (2004)
6. Lin, H., Bilmes, J.: Multi-document summarization via budgeted maximization of submodular functions. In: *HLT-NAACL*, pp. 912–920 (2010)
7. Mahdiraji, A.R.: Clustering data stream: A survey of algorithms. *International Journal of Knowledge-based and Intelligent Engineering Systems* 13, 39–44 (2009)
8. Mani, I.: *Automatic Summarization*. John Benjamins Publisher, Amsterdam (2001)
9. McDonald, R.: A study of global inference algorithms in multi-document summarization. In: Amati, G., Carpineto, C., Romano, G. (eds.) *ECIR 2007*. LNCS, vol. 4425, pp. 557–564. Springer, Heidelberg (2007)
10. O’Connory, B., Krieger, M., Ahn, D.: Tweetmotif: Exploratory search and topic summarization for twitter. In: *AAAI Conf. on Weblogs and Social Media*, pp. 384–385 (2010)
11. Petrovic, S., Osborne, M., Lavrenko, V.: Streaming first story detection with application to twitter. In: *NAACL*, pp. 181–189 (2010)
12. Ritter, A., Cherry, C., Dolan, B.: Unsupervised modeling of twitter conversations. In: *NAACL*, pp. 172–180 (2010)
13. Sharifi, B., Hutton, M.A., Kalita, J.: Summarizing microblogs automatically. In: *NAACL*, short paper, pp. 685–688 (2010)
14. Swan, R., Jensen, D.: Timemines: Constructing timelines with statistical models of word usage. In: *SIGKDD Workshop on Text Mining*, pp. 73–80 (2000)
15. Takamura, H., Okumura, M.: Text summarization model based on the budgeted median problem. In: *CIKM*, short paper, pp. 1589–1592 (2009)

# A Link Prediction Approach to Recommendations in Large-Scale User-Generated Content Systems

Nitin Chiluka, Nazareno Andrade, and Johan Pouwelse

Delft University of Technology,  
The Netherlands

{n.j.chiluka,n.ferreiradeandrade,j.a.pouwelse}@tudelft.nl

**Abstract.** Recommending interesting and relevant content from the vast repositories of User-Generated Content systems (UGCs) such as YouTube, Flickr and Digg is a significant challenge. Part of this challenge stems from the fact that classical collaborative filtering techniques – such as k-Nearest Neighbor – cannot be assumed to perform as well in UGCs as in other applications. Such technique has severe limitations regarding data sparsity and scalability that are unfitting for UGCs. In this paper, we employ adaptations of popular Link Prediction algorithms that were shown to be effective in massive online social networks for recommending items in UGCs. We evaluate these algorithms on a large dataset we collect from Flickr. Our results suggest that Link Prediction algorithms are a more scalable and accurate alternative to classical collaborative filtering in the context of UGCs. Moreover, our experiments show that the algorithms considering the immediate neighborhood of users in an user-item graph to recommend items outperform the algorithms that use the entire graph structure for the same. Finally, we find that, contrary to intuition, exploiting explicit social links among users in the recommendation algorithms improves only marginally their performance.

**Keywords:** User-Generated Content Systems, Recommendation, Collaborative Filtering, Link Prediction, Flickr.

## 1 Introduction

User-Generated Content systems (UGCs) such as YouTube, Flickr, and Digg have transformed the way media is shared and consumed. Prior to UGCs, content distribution to large audiences was costly enough to have its control lying in a few hands: typically, those of professional and commercial producers. However, with technological advances in the recent years, the cost of producing and distributing media has drastically reduced. UGCs have created venues where many of those who were previously passive content consumers now publish and share videos, photos, news articles and other content.

The democratic nature of publishing and sharing through UGCs has attracted millions of users to contribute to these systems. Such large user bases, combined

with the low cost and effort required to produce and publish content, typically create massive and fast-growing content repositories in UGCs [8]. On the one hand, such sheer volume of content may cater to varied tastes of users. On the other hand, this *scale* gives rise to the problem of information overload, also formulated as the Babel objection [7]: differentiating quality from noise in such large user-generated data is very difficult. In essence, finding interesting and relevant content in UGCs is a significant challenge.

Recommendation algorithms are a candidate solution to address this problem. UGCs can use these algorithms to suggest interesting and relevant content to users based on their past preferences, thereby partially automating the process of content discovery. In spite of this potential, however, few studies have explored recommending items in the context of UGCs (with the exception of YouTube [6]). Collaborative filtering (CF) techniques have been established as a good fit for this purpose in editorially-generated content systems (Non-UGCs), such as MovieLens and Netflix, but there is little or no evidence that these techniques would perform well if applied to UGCs.

**Motivation and Challenges.** There are primarily two reasons why classical CF techniques which were found to be effective for Non-UGCs – such as k-Nearest Neighbor – cannot be assumed to perform as well in UGCs. First, the *user-item matrix* is substantially larger and sparser in UGCs than in Non-UGCs. This difference exacerbates the already existing limitations in CF techniques regarding scalability and sparsity [12]. While Non-UGCs such as Internet Movie Database have about a million items [8], UGCs may contain hundreds of millions of items. For instance, YouTube has recently claimed [4] that hundreds of thousands of videos are uploaded everyday on its website, accounting for 24 hours of videos every minute. Also, Flickr has celebrated the five billionth photo [1] uploaded in September 2010, and 3000 photos are being uploaded every minute.

The second reason that challenges the application of classical CF techniques in UGCs relates to the lack of editorial control in these systems. While all items in Non-UGCs are introduced and organized in respective categories by a few trusted editors, UGCs contain content published by millions of users, and not all items are properly categorized or of good quality. This essentially magnifies the problem of users to sift through noise and find content of interest and relevance.

**Approach and Contributions.** In this paper, we employ adaptations of Link Prediction algorithms [11] for recommending items in large UGCs. These are a family of graph-based algorithms from the social network analysis literature which were found to be effective in predicting new links that might form in a given social network graph. A recent study [16] has shown that these algorithms are effective in predicting the formation of links between users in large and sparse social network graphs such as those of YouTube, Flickr, Digg, and LiveJournal. Inspired by their effectiveness to handle graphs with millions of nodes, we adopt these algorithms for CF in UGCs. To apply them, we modify these algorithms to suit *user-item graphs*, since they were originally designed for graphs with only one kind of nodes (e.g., users). We use six Link Prediction algorithms in this study based on node-neighborhood, popularity and path-ensemble.



We evaluate the recommendation performance of these algorithms on a large dataset<sup>1</sup> we collected from Flickr containing 120,812 users and 83,435 photos. We use the popular item-based collaborative filtering technique [12] as a baseline to analyze the effectiveness of Link Prediction-based recommendation.

Our results show that the adapted Link Prediction algorithms outperform a widely used item-based CF technique [12] we consider. Moreover, the relative performance of different Link Prediction algorithms unveils that most users are interested in photos within a short distance from them in the user-item graph. We also examine whether exploiting the explicit relationships among users which are often present in UGCs improves the recommendation performance of our algorithms. Our findings suggest that, contrary to intuition, there is only a slight improvement in recommendation performance. Finally, for all Link Prediction algorithms, the more friends a user has and the more photos she bookmarks, the better the recommendations she gets.

## 2 Background and Related Work

In this section, we first characterize what constitutes UGCs and then review the literature related to recommendations and social influence in UGCs.

**User-Generated Content Systems.** Two main characteristics define the current User-Generated Content systems (UGCs). First, any user can publish and share content items that other users can view and express opinions about, in the form of ratings, bookmarks and comments. For example, in YouTube, a user can upload a video which other users can view, give a rating (Like/Dislike), or add as a *favorite*. Similarly, in Flickr, a user can upload a photo which other users can view or add as a *favorite*. Digg has similar features as well: a user can submit a story which others can view or bookmark as a *digg*. Additionally, users can comment on any content item in each of these UGCs.

Second, in UGCs, users can form *social relationships* with other users. These are usually framed as *friendships* or *subscriptions*, and primarily indicate interest of a user in another user's activity. For instance, in YouTube, a user can subscribe to other users to keep updated of their uploaded videos as well as browse through the videos they have added as favorites. Flickr also allows a user to add others as *contacts*, which helps her to keep abreast of their activity. A Digg user can also add others as *friends* to follow their submissions and diggs. A relationship between any two users in such systems is typically asymmetric, i.e., a friendship link from a user  $A$  to user  $B$  means that the former is interested in the latter's activity, but not necessarily vice-versa.

**Item Recommendations in UGCs.** Few studies have explored *item* recommendations in UGCs. To the best of our knowledge, the work on YouTube by Baluja et al [6] is the only extensive study on recommending items in UGCs. The authors propose a novel graph-based technique called Adsorption, which recommends videos given a co-view graph representing which user viewed what video. Adsorption considers a video is relevant to a user if there are many short paths

---

<sup>1</sup> This dataset is available upon request.

in the co-view graph between the user and the video which avoid high-degree nodes. This method is similar to Katz Measure [10], one of the Link Prediction algorithms we use in this paper.

**Social Influence in UGCs.** Some recent studies analyzed the role of social (user-user) links for disseminating and promoting user-generated content in UGCs. For example, an in-depth analysis of dissemination of photos along user-user links in the Flickr social network revealed that over half of all favorite-markings are exchanged between friends, thereby indicating a significant social influence on this behavior [9]. Another extensive study on diffusion of user-generated content in YouTube [13] found that social influence plays a prominent role in both the success of video as well as the magnitude of the impact. Different from these studies which analyze content diffusion along social links, our work focuses on content adoption by users irrespective of social influence.

### 3 A Link Prediction Approach to Collaborative Filtering

The fundamental task of collaborative filtering (CF) is to predict the interest-iness and relevance of an item to a user. This is typically done based on *how closely* this item is related to the user’s tastes. Basically, *proximity* – the measure of *closeness* – lies at the heart of CF. The challenge of applying CF to UGCs translates into developing methods for calculating proximity that are both effective and scalable for large user-item spaces.

In this paper, we advance the hypothesis that the methods based on Link Prediction algorithms [11] provide an effective and scalable solution for CF in UGCs. Like CF, the underlying rationale of most Link Prediction algorithms is based on proximity. The Link Prediction problem is to predict the formation of links in a social network graph, and the corresponding solutions explore the principle that the closer two nodes are in such a graph, the higher the chance a link between them forms. Unlike classical CF techniques, however, some of the Link Prediction algorithms have been shown [16] to be highly scalable, performing well in massive and sparse social network graphs such as those of YouTube, Flickr, Digg, and LiveJournal.

To bridge Link Prediction algorithms and CF in UGCs, we develop variants of some of the Link Prediction algorithms in the literature [11,16] to suit *user-item graphs*. These variants are necessary because Link Prediction algorithms were mainly designed for graphs containing only one kind of nodes (e.g., users). In contrast, CF in UGCs concerns with predicting links between two types of nodes: users and items. Hence, CF can essentially be viewed as a *user-item link prediction problem*: given a graph with users and items as nodes, and users’ tastes in items represented by user-item edges, how accurately can we infer whether a link will form between an item and a particular user? Each Link Prediction algorithm we use in the paper solves this problem by calculating a *proximity score* that expresses how *relevant* any item is to a particular user in the graph.

### 3.1 Notation

We model a user-generated content system as a directed graph  $G = (V, E)$  where the set of nodes  $V$  consists of all users  $U$  and items  $I$  present in the system ( $V = U \cup I$ ), and  $E$  is the set of edges that represent various relationships among these nodes ( $E \subseteq U \times U \cup U \times I$ ). A node-pair of a user and an item is always connected by two edges, one in each direction (*user-item links*). A node-pair of users may be connected by either a single edge or two edges in opposite directions (*user-user links*). Two item nodes are never connected.

The adjacency matrix  $A$  of the *user-item graph*  $G$  is such that  $A[x, y] = 1$  if edge  $(x, y) \in E$ , otherwise  $A[x, y] = 0$ . In addition, we define  $N_u(x)$  and  $N_i(x)$  as the set of users and items that are neighbors of a node  $x$  in the user-item graph, respectively. That is,  $N_u(x) = \{y \mid (x, y) \in E \text{ and } y \in U\}$  and  $N_i(x) = \{y \mid (x, y) \in E \text{ and } y \in I\}$ . Finally, we denote  $N^{-1}(x)$  as the set of nodes having  $x$  as their neighbor ( $N^{-1}(x) = \{y \mid (y, x) \in E\}$ ).

### 3.2 Algorithms

We adapt and employ six Link Prediction algorithms in this paper. Link Prediction algorithms can be broadly classified according to the node characteristic they rely on when calculating proximity. Most of these algorithms are based on node neighborhood, popularity or path ensemble. The six algorithms we adapt include two algorithms based on each of these characteristics, namely:

- *Node-neighborhood-based*: Common Neighbors and Adamic/Adar;
- *Popularity-based*: Global Popularity and PageRank;
- *Path-ensemble-based*: Katz Measure and Rooted PageRank.

Among these, node-neighborhood-based algorithms have restricted scalability, and do not necessarily constitute a viable approach for UGCs. However, we adapt and include them in this study because such methods were shown to provide highly effective predictions from both theoretical [15] and practical perspectives [11], even for large datasets [16]. As such, they are used in our experiments as references for the performance that should ideally be achieved by the other two groups of algorithms, which have been shown to scale for graphs with millions of nodes [16].

In the rest of the section, we propose these algorithms for a bipartite user-item graph which does not include edges among users. Later, in Sec. 5.2, we revisit these algorithms taking into account user-user links as well. Each algorithm  $ALG$  calculates a proximity score  $ALG(u, z)$  for a given user  $u$  and item  $z$ .

**Common Neighbors.** The rationale behind this algorithm is that the greater the intersection of the neighbor sets of any two nodes, the greater the chance of future association between them [11]:  $CN(u, z) = \sum_{v \in N_i(u)} |N_u(v) \cap N_u(z)|$ .

**Adamic/Adar.** This method also measures the intersection of neighbor-sets of two nodes in the graph, but emphasizes the smaller overlap [5]:  $AA(u, z) = \sum_{j \in N_i(u)} \sum_{v \in N_u(z) \cap N_u(j)} (\log |N_i(v)|)^{-1}$ .

**Global Popularity.** This method is a variant of *preferential attachment*, which quantifies the popularity of a node as its degree, and uses this popularity as the proximity from any other node [11]:  $GP(u, z) = N_u(z)$ .

**PageRank.** This algorithm leverages the link structure of a graph to quantify the popularity of each node [14], by assigning to this node a global rank  $PR(u, z) = PR(z)$ , where  $PR(z) = (1 - d)(|V|)^{-1} + d \sum_{x \in N^{-1}(z)} PR(x)(|N(z)|)^{-1}$ , and  $d(=0.85$  in this paper) is the teleportation parameter.

**Katz Measure.** The proximity score for this method is calculated by considering all paths in the graph [10]. The logic is that the more the paths between any two nodes in the graph and the shorter these paths, the greater the “bond” between these nodes:  $KM = (I - \beta_{Katz}A)^{-1}$  where  $\beta_{Katz}$  is the damping factor.

**Rooted PageRank.** This method [11][16] is a variant of the personalized PageRank algorithm. It attempts to capture the probability of random walks starting from two nodes in the graph to come across each other, and uses this probability to quantify the proximity between these nodes. Let  $D$  be the diagonal matrix with  $D[i, i] = \sum_j A[i, j]$ , then  $RPR = (1 - \beta_{RPR})(I - \beta_{RPR}D^{-1}A)^{-1}$ , where  $\beta_{RPR}$  is the teleportation parameter.

In each of the matrices  $KM$  and  $RPR$ , the element in row- $u$  and column- $z$  denotes the proximity score for the respective algorithm, given a user  $u$  and an item  $z$ . For scalable computation of the matrix inversion used in Katz Measure and Rooted PageRank, we use a dimensionality reduction technique called Proximity Embedding [16]. We also use  $\beta_{Katz} = \beta_{RPR} = \beta = 0.005$ .

## 4 Experimental Methodology

We evaluate our algorithms on Flickr [2], which contains explicit user-item and user-user links, depicting tastes and sources of influence for any user respectively. With reference to the terminology introduced in Section 2, in Flickr, we consider *favorites* as user-item links<sup>2</sup> and *contacts* as (directed) user-user links. Nevertheless, in the rest of the paper, we use the terms ‘photo’ and ‘item’ interchangeably. Also, we intermingle the terms ‘contacts’ and ‘friends’.

**Data Collection.** Flickr currently has several millions of users and items. Since obtaining all relevant data for this study is infeasible, we sample the user-item graph. We use snowball sampling, a popular approach in online social networks. This method leads to nearly complete data for a particular neighborhood of the graph, which is of interest to recommendation algorithms. The time window considered for collecting data about users and favorites is three months.

**Dataset Description.** Table 1 presents the summary of the collected data. The data is divided into a training period of one month and a testing period of two months. Note that the median value for the number of favorites is five compared to 50 for contacts: the number of user-user links is nearly an order of magnitude larger than the number of favorite markings.

**Metrics.** The recommendation performance of the six Link Prediction algorithms is measured in our experiments in terms of precision, recall and mean average precision (MAP). To calculate these metrics, we adopt the approach of Baluja et al. [6]. Specifically, for each user  $u$ , we use the photos she bookmarked

<sup>2</sup> The data about who viewed what item is private to Flickr. We therefore limit our discussion to the data publicly available through the Flickr API [3].

**Table 1.** Summary of Flickr dataset

Training period	Jan 1,2010 to Jan 31, 2010
Testing period	Feb 1, 2010 to Mar 31, 2010
# Users active in both periods	120,812
# Photos active in both periods	83,435
# Favorite markings in training period	1,755,575
# Favorite markings in testing period	1,234,854
Median favorites per user	5
Median contacts per user	50

during the training period to generate a ranked list  $R_u$  of photos for each given algorithm. The top- $N$  (in Sec. 5,  $N = 200$ ) photos from this list are then used for evaluation. Let  $R_u^i$  be the set of the first  $i$  photos in  $R_u$ , and let  $W_u$  be the set of photos a user  $u$  bookmarked during testing period. Then, for  $i < |R|$ , for the user  $u$  at rank-position  $i$ , we define: (i) precision:  $p_u^i = |W_u \cap R_u^i|/|R_u^i|$  and (ii) recall:  $r_u^i = |W_u \cap R_u^i|/|W_u|$ . The computation of precision and recall at rank-position  $i$  for each algorithm averages  $p_u^i$  and  $r_u^i$  across all users, respectively. The calculation of MAP for user  $u$  averages precision values at rank-positions in the ranked list which match relevant items. MAP for each algorithm  $ALG$  is calculated by averaging across all users.

These metrics reflect the effectiveness of the algorithms, but not their efficiency. The efficiency, and hence the scalability of these methods, has been extensively studied before [16], with results that also hold for user-item graphs.

**Baseline Method.** We compare the recommendation performance of our algorithms against the widely-used item-based collaborative filtering technique [12] as a baseline. Although this method is infeasible for millions of users and items, it can still be applied to our dataset.

## 5 Evaluation

In this section, we evaluate the recommendation performance of our Link Prediction algorithms on the crawled Flickr dataset. We aim to answer the following questions: (i) How do these algorithms perform in comparison to an item-based collaborative filtering technique [12], which we use as a baseline? (ii) Does exploiting the knowledge of explicit user-user links improve recommendation performance? (iii) How does the behavior of a user influence the quality of her recommendations?

### 5.1 Link Prediction Performance

In this experiment, we compare the recommendation performance of our Link Prediction algorithms against the item-based collaborative filtering technique as a baseline, considering the user-item graph with only user-item links. Table 2 shows the recommendation performance in terms of MAP, precision and recall of each algorithm at various rank positions.

**Table 2.** Recommendation performance (%) of the algorithms

Algorithm	MAP	Precision@1	Precision@10	Precision@100	Recall@10	Recall@100	Recall@200
CN	6.434	7.130	<b>1.730</b>	0.577	<b>2.298</b>	5.732	7.468
AA	6.240	6.795	<b>1.783</b>	<b>0.613</b>	<b>2.425</b>	<b>6.448</b>	<b>8.385</b>
GP	6.566	<b>7.269</b>	0.817	0.198	1.120	2.185	2.810
PR	6.576	<b>7.269</b>	0.823	0.192	1.132	2.246	3.037
KM	5.785	6.898	1.365	0.547	1.713	5.583	7.754
RPR	4.561	5.061	1.308	<b>0.609</b>	1.571	<b>6.238</b>	<b>9.198</b>
Item-CF	5.183	5.702	1.529	0.525	2.045	5.762	7.914

Overall, Link Prediction algorithms outperform the baseline item-based CF technique. Among Link Prediction algorithms, the algorithms based on node-neighborhood perform better than those based on popularity and path-ensemble. On the other hand, popularity-based algorithms perform the worst.

Both neighborhood-based algorithms – Common Neighbors and Adamic/Adar – have similar recommendation performance. Although Adamic/Adar has a marginally higher overall performance than that of Common Neighbors, the latter performs better for the top-few ranked items. A closer look at our datasets reveals that these algorithms perform well because most favorites marked by a node are in its close neighborhood. In our data, 75% of the favorites marked during the testing period were within a distance of three hops from the users who bookmarked them in the graph.

The recommendation performances of both popularity-based algorithms – Global Popularity and PageRank – are also similar. To examine why this happens, we compare the sheer popularity (node degrees) and PageRank values for each item. This analysis shows that the *most popular* items also have high PageRank values. We also note that recall values for these algorithms do not improve to the extent of other algorithms, suggesting that only the top few ranked items were of interest and relevance to users.

Among the path-ensemble algorithms, Katz Measure performs better than Rooted PageRank in terms of MAP, as well as precision for top-few ranked items. On the other hand, Rooted PageRank surprisingly has the highest recall value at 200-th rank-position among all algorithms. This shows that, although Rooted PageRank is able to correctly recommend more items than any other algorithm, the order in which these recommended items are ranked is not as effective as that of neighborhood-based algorithms or Katz Measure. We also note that varying the parameter of path-ensemble algorithms from  $\beta = 0.005$  to an order of magnitude greater and smaller caused no performance changes.

We now focus on the performance of these path-ensemble algorithms while recommending items beyond three hops from users in the user-item graph. In this context, Rooted PageRank outperforms Katz Measure. The former has 1.5 times better precision at the top-ranked position and 5 times higher Precision@10 compared to Katz Measure. We also observe that Rooted PageRank has 6 times better recall for top-200 items than that of Katz Measure. These observations suggest that predicting more items which are beyond three hops correctly may be one of the reasons why the recall of Rooted PageRank improves significantly.

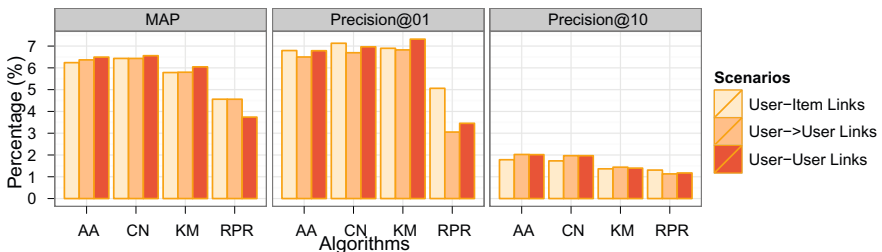
We observe that the overall values of MAP, precision and recall in our experiments are considerably smaller than those in most studies in the field of information retrieval. This can be due to two reasons. First, the large size and sparsity of the dataset challenge the algorithms. We note here that the study on recommending videos in YouTube also reported low precision and recall values [6], suggesting that collaborative filtering in UGCs is a especially difficult task. Second, some items that were recommended by these algorithms might have been bookmarked either before the training period or after the testing period, which the collected data does not capture.

## 5.2 Influence of Explicit Social Links

Our second experiment investigates whether exploiting explicit user-user links improves the recommendation performance of our algorithms. For this purpose, we first include both user-item and user-user links in the user-item graph and then employ adapted versions of our algorithms. Furthermore, we explore the direction of user-user links is affects recommendations, comparing results in the original graph with those from a symmetrized graph. In the symmetrization, for each node pair containing only one edge, we add a link in the reverse direction.

The algorithms are adapted for considering user-user links as follows. The scores for node-neighborhood methods are redefined as  $CN(u, z) = |N_u(u) \cap N_u(z)|$  and  $AA(u, z) = \sum_{v \in N_u(u) \cap N_u(z)} \frac{1}{\log |N_i(v) + N_u(v)|}$ . For Katz Measure and Rooted PageRank, we incorporate edges among users in the adjacency matrix of the graph before computing the scores using the same techniques. Since the popularity-based algorithms are unaffected by the user-user links in the graph, they are not considered in this experiment.

Figure 1 shows the recommendation performance of each algorithm across three scenarios: (i) the one with only user-item links (as a baseline), (ii) the one with the unmodified user-item graph considering user-user links and (iii) the one with reciprocated user-user links. We make the following observations:



**Fig. 1.** Recommendation performance in three scenarios based on either inclusion or the directedness of user-user links

**Effect of User-User Links.** The overall recommendation performance of Common Neighbors, Adamic/Adar and Katz Measure slightly improves when user-user links are considered. Rooted PageRank, however, has a reduced performance in terms Precision@10. Although Katz Measure has a better Precision@1 than any other algorithm, its overall performance is still below that of neighborhood algorithms without considering user-user links. The precision of Rooted PageRank is reduced by 40%, which may be attributed to the increment in the node-degree for each user in the graph due to the inclusion of user-user links. This increment may reduce the influence of each neighbor during a random walk.

**Effect of Link Symmetry.** The reciprocation of user-user links marginally improves recommendation performance. The low effect size indicates that the influence among users in Flickr is not usually mutual, i.e., if a user  $A$  has added a user  $B$  as a contact, and there is no reciprocation from user  $B$ , the influence of  $A$ 's tastes on  $B$  is small. Song et al. [16] had similar findings for the prediction of user-user links in Flickr: accuracy was nearly the same when graph was symmetrized.

### 5.3 Influence of User Behavior

We now examine how the behavior of a user influences the quality of her recommendations. In this experiment, user behavior is characterized by the number of items bookmarked and friends. The former indicates a user's activeness, while the latter suggests the amount of social influence on her.

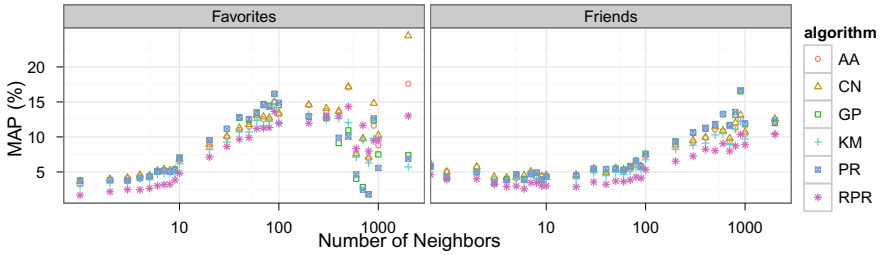
**Effect of User Activeness.** We characterize users with different activity levels by grouping users in bins according to their activity. This is done approximating the number of bookmarked items of each user to the nearest power of 10. After this, the MAP value for users in each bin is averaged.

The left half of Figure 2 presents the recommendation performance of each algorithm as a function of the number of bookmarked items (*favorites*). The recommendation performance of each algorithm is in general higher for users that were more active than for those who bookmarked fewer items during the training period. A positive correlation between available information and recommendation quality is in accordance with the intuition in recommender systems. Nevertheless, these results show that this intuition is valid also in the context of UGCs, and provide evidence on precisely how recommendation performance grows with user activity. Finally, we note that the relation between these two factors is particularly clear for users with less than 500 favorites. The higher variance among users with more than 500 bookmarked items is, however, by and large the result of a low activity of this users during the testing period.

**Effect of Social Influence.** Similar to the previous experiment, we bin users based on their number of friends, and then average the MAP values for each bin. The number of friends is approximated to the nearest power of 10.

The right half of Figure 2 presents the recommendation performance of each algorithm as a function of users' number of friends. The recommendation performance of each algorithm is in general higher for users with a large number of contacts compared to those with few. MAP for users with around 1000 friends is 2-3 times higher compared to those with fewer than 100. It is also interesting





**Fig. 2.** Recommendation performance with respect to user behavior

to note that the recommendation performance of each algorithm does not vary much for users with fewer than 100 friends, but it increases gradually until 1000. This pattern is similar to that of favorites', except that the latter's improvement starts at 10 favorites. This may be ascribed to the ratio of median values of friends and favorites per user, which is precisely 10 for this dataset. In spite of this difference, however, we may conclude that the more friends a user has and the more items she bookmarks, the better the recommendations to her.

## 6 Conclusions

In this paper, we advanced a Link Prediction-based approach for recommending items in large-scale UGCs. We believe that, besides the work on YouTube [6], this paper is the only extensive study on recommending items in UGCs.

We evaluated the recommendation performance of six Link Prediction algorithms on a large dataset we collected from Flickr, with the widely-used item-based collaborative filtering technique as a baseline. Three of the Link Prediction algorithms – Common Neighbors, Adamic/Adar and Katz Measure – performed consistently better than the item-based collaborative filtering technique. We found that neighborhood-based methods outperform all other algorithms, suggesting that users are mostly interested within a small proximity of their tastes in the user-item space. Rooted PageRank, on the other hand, was very effective in recommending items that are beyond three hops from users in the user-item graph. In addition, exploiting the explicit relationships among users improved recommendation performance only marginally, contrary to our expectations. With respect to user behavior, the larger the number of friends of a user or the number of photos bookmarked by her, the better the quality of recommendations to her, implying that social influence and activeness of user does affect the performance. Finally, the low values of precision and recall in our study, along with the only other extensive investigation in our knowledge, suggest that recommending items in UGCs is highly challenging, and hence requires significant attention from IR research community.

**Acknowledgments.** This work was partially supported by the European Commission's 7th Framework Program through the P2P-Next and QLectives projects (grant no. 216217, 231200).

## References

1. 5,000,000,000.. flickr blog, <http://blog.flickr.net/en/2010/09/19/5000000000/>
2. Flickr, <http://www.flickr.com>
3. Flickr api, <http://www.flickr.com/services/api/>
4. Youtube fact sheet, [http://www.youtube.com/t/fact\\_sheet/](http://www.youtube.com/t/fact_sheet/)
5. Adamic, L.A., Adar, E.: E-commerce: the role of familiarity and trust. In: Social Networks, pp. 211–230 (2003)
6. Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., Ravichandran, D., Aly, M.: Video suggestion and discovery for youtube: taking random walks through the view graph. In: WWW 2008: Proceeding of the 17th International Conference on World Wide Web (2008)
7. Benkler, Y.: The Wealth of Networks: How Social Production Transforms Markets and Freedom, ch. 5
8. Cha, M., Kwak, H., Rodriguez, P., Ahn, Y.-Y., Moon, S.: I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement
9. Cha, M., Mislove, A., Gummadi, K.P.: A Measurement-driven Analysis of Information Propagation in the Flickr Social Network. In: WWW 2009: Proceedings of the 18th International World Wide Web Conference (2009)
10. Katz, L.: A new status index derived from sociometric analysis. *Psychometrika* (1953)
11. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: CIKM 2003: Proceedings of the Twelfth International Conference on Information and Knowledge Management (2003)
12. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* (2003)
13. Oh, J., Susarla, A., Tan, Y.: Examining the diffusion of user-generated content in online social networks. *SSRN eLibrary* (2008)
14. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University (1999)
15. Sarkar, P., Chakrabarti, D., Moore, A.W.: Theoretical justification of popular link prediction heuristics. In: COLT 2010: Proceedings of the 23rd Annual Conference on Learning Theory (2010)
16. Song, H.H., Cho, T.W., Dave, V., Zhang, Y., Qiu, L.: Scalable proximity estimation and link prediction in online social networks. In: IMC 2009: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference (2009)

# Topic Classification in Social Media Using Metadata from Hyperlinked Objects\*

Sheila Kinsella<sup>1</sup>, Alexandre Passant<sup>1</sup>, and John G. Breslin<sup>1,2</sup>

<sup>1</sup> Digital Enterprise Research Institute, National University of Ireland, Galway  
{firstname.lastname}@deri.org

<sup>2</sup> School of Engineering and Informatics, National University of Ireland, Galway  
john.breslin@nuigalway.ie

**Abstract.** Social media presents unique challenges for topic classification, including the brevity of posts, the informal nature of conversations, and the frequent reliance on external hyperlinks to give context to a conversation. In this paper we investigate the usefulness of these external hyperlinks for determining the topic of an individual post. We focus specifically on hyperlinks to objects which have related metadata available on the Web, including Amazon products and YouTube videos. Our experiments show that the inclusion of metadata from hyperlinked objects in addition to the original post content improved classifier performance measured with the F-score from 84% to 90%. Further, even classification based on object metadata alone outperforms classification based on the original post content.

## 1 Introduction

Social media such as blogs, message boards, micro-blogging services and social-networking sites have grown significantly in popularity in recent years. By lowering the barriers to online communication, social media enables users to easily access and share content, news, opinions and information in general. However navigating this wealth of information can be problematic due to the fact that contributions are often much shorter than a typical Web documents, and the quality of content in social media is highly variable [1].

A potential source of context to conversations in social media is the hyperlinks which are frequently posted to refer to related information. These objects are often an integral part of the conversation. For example, a poster may recommend a book by posting a link to a webpage where you can buy the book, rather than employing the traditional method of providing the title and the name of the author. Yet there still remains the question of identifying which snippets of content from these links are most relevant to the conversation at hand.

Recently, there has been a trend towards publishing structured information on the Web, resulting in an increasing amount of rich metadata associated with Web resources. Facebook have launched their Open Graph Protocol [2], which allows

\* The work presented in this paper has been funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

<sup>1</sup> <http://opengraphprotocol.org/> visited January 2011

external site owners to markup their content using Facebook-defined schemas, such that these enriched content items can then be used for metadata import into news feeds, profiles, etc. The Linking Open Data [5] project is a community effort to make structured datasets from diverse domains available on the Web, some of which we use as data sources in this work. Such rich representations of objects can be a useful resource for information retrieval and machine learning. In social media in particular, structured data from hyperlinked websites can provide important context in an otherwise chaotic domain.

In this paper, we investigate the potential of improving topic classification in social media by using metadata retrieved from external hyperlinks in user-generated posts[4]. We compare the results of topic classification based on post content, based on metadata from external websites, and based on a combination of the two. The usage of structured metadata allows us to include only specific, relevant external data. Our experiments show that incorporating metadata from hyperlinks can significantly improve the task of topic classification in social media posts. Our approach can be applied to recommend an appropriate forum in which to post a new message, or to aid the navigation of existing, unclassified posts.

Related work has been carried out in the field of Web document classification, proving that the classification of webpages can be boosted by taking into account the text of neighbouring webpages ([2], [9]). This work differs in that we incorporate not entire webpages but only metadata which is directly related to objects discussed in a post. There has also been previous work using tags and other textual features to classify social media. Our work is closely related to that of Figueiredo et al. [6], who assess the quality of various textual features in Web 2.0 sites such as YouTube for classifying objects within that site. Berendt and Hanser [4] investigated automatic domain classification of blog posts with different combinations of body, tags and title. Sun et al. [10] showed that blog topic classification can be improved by including tags and descriptions. Our paper differs from these because we use metadata from objects on the Web to describe a post which links to those objects. Thus our approach combines the usage of neighbouring pages in Web search, and of metadata in social media search.

## 2 Data Corpus

Our analysis uses the message board corpus from the [boards.ie](http://boards.ie) SIOC Data Competition[3] which was held in 2008 and covers ten years of discussions. Each post belongs to a thread, or conversation, and each thread belongs to a forum. A forum typically covers one particular area of interest, and may contain sub-forums which are more specialised (for example, Music and Hip-Hop). Our analysis uses a subset of the forums and is limited to the posts contained in the final year of the dataset, because these are most likely to have structured data. We examined the posts in the dataset which contained hyperlinks and identified potential sources of structured metadata. These are websites which publish

<sup>2</sup> A post is a message which a user submits as part of a conversation.

<sup>3</sup> <http://data.sioc-project.org/> visited January 2011.

**Table 1.** External websites and the metadata types used in our experiments

Website	Object type	Title	Description	Category	Tags
Amazon	Product	X		X	X
Flickr	Photo	X	X		X
IMDB	Movie	X		X	
MySpace	Music Artist	X		X	
Wikipedia	Article	X	X	X	
YouTube	Video	X	X	X	X

metadata about objects, such as videos (YouTube) or products (Amazon), and make it available via an API or as Linked Data. In some cases the metadata is published by external sources, e.g. DBpedia [3] provides a structured representation of Wikipedia. The sources on which our study focuses are listed in Table 1, along with the available metadata that we extracted. We consider the first paragraph of a Wikipedia article as a description. For this study, we focus on the most commonly available metadata, but in the future we plan to make use of additional information such as movie directors, book authors and music albums.

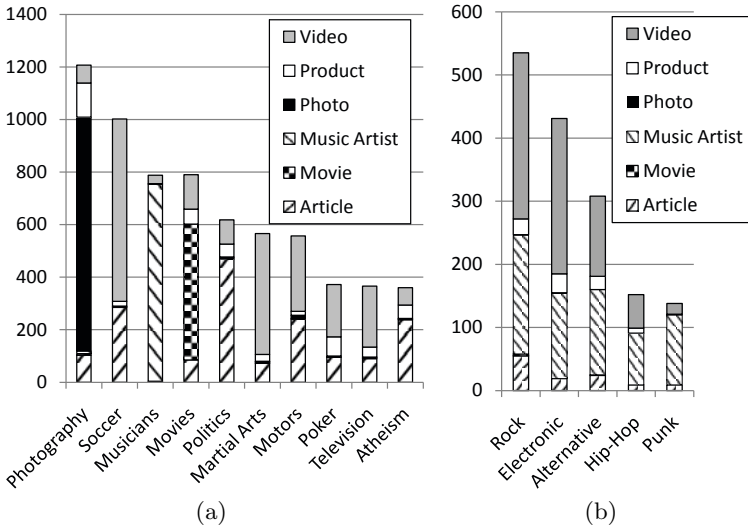
Amazon, Flickr and YouTube metadata was retrieved from the respective APIs. We obtained MySpace music artist information from DBTune<sup>4</sup> (an RDF wrapper of various musical sources including MySpace), IMDB movie information from LinkedMDB<sup>5</sup> (an export of IMDB data) and Wikipedia article information from DBpedia<sup>6</sup>. The latter three services are part of the Linking Open Data project [5]. The data collection process is described in more detail in [8].

Two groups of forums were chosen for classification experiments - the ten rather general forums shown in Figure 1(a), which we refer to as **General**, and the five more specific and closely related music forums shown in Figure 1(b), **Music**. These forums were chosen since they have good coverage in the dataset and they each have a clear topic (as opposed to general “chat” forums). The percentage of posts in **General** that have hyperlinks varies between forums, from 4% in Poker to 14% in Musicians, with an average of 8% across forums. Of the posts with hyperlinks, 23% link to an object with available structured data. The number of posts containing links to each type of object is shown in Figure 2. For **General**, hyperlinks to Music Artists occur mainly in the Musicians forum, Movies in the Films forum, and Photos in the Photography forum. The other object types are spread more evenly between the remaining seven forums. Note that in rare cases, a post contains links to multiple object types, in which case that post is included twice in a column. Therefore the total counts in Figure 2 are inflated by approximately 1%. In total, **General** contains 6,626 posts and **Music** contains 1,564 posts.

<sup>4</sup> <http://dbtune.org/> visited January 2011.

<sup>5</sup> <http://linkedmdb.org/> visited January 2011.

<sup>6</sup> <http://dbpedia.org/> visited January 2011.



**Fig. 1.** Number of posts containing each type of hyperlinked object for (a) 10 General forums and (b) 5 Music forums

### 3 Experiments

We evaluated the usefulness of various textual representations of message board posts for classification, where the class of a post is derived from the title of the forum in which it was posted. For each post, we derive four different bag-of-words representations, in order to compare their usefulness as sources of features for topic classification. **C-L** denotes the original content of the post, with all hyperlinks removed, while **C** denotes the full original content with hyperlinks intact. **M** denotes the external metadata retrieved from the hyperlinks of the post. **C+M** denotes the combination of **C** and **M** for a given post, i.e. the full original content plus the external metadata retrieved from its hyperlinks. For the 23% of posts which have a title, this is included as part of the content. The average number of unique terms was 38 for post content, and 20 for associated metadata. At present we simply concatenate the text of the metadata values rather than considering the metadata key-value pairs, however it would be interesting to weight the metadata based on which keys provide the most useful descriptors for classification.

Classification of documents was performed using the Multinomial Naïve Bayes classifier implemented in Weka [7]. For each textual representation of each post the following transformations are performed. All text is lower-cased and non-alphabetic characters are replaced with spaces. Stopwords are removed and TF-IDF and document length normalisation are applied.

Ten-fold cross validation was used to assess classifier performance on each type of document representation. To avoid duplication of post content due to one post quoting another, the data was split by thread so that posts from one

**Table 2.** Micro-averaged F-scores achieved by classifier on each dataset

Dataset	C-L	C	M	C+M
General	0.783	0.838	0.858	0.902
Music	0.663	0.694	0.780	0.803

**Table 3.** F-score achieved by classifier on each **General** forum

Forum	C-L	C	M	C+M
Musicians	0.948	0.976	0.901	0.980
Photography	0.777	0.918	0.895	0.948
Soccer	0.812	0.831	0.909	0.949
Martial Arts	0.775	0.810	0.877	0.914
Motors	0.751	0.785	0.865	0.907
Films	0.744	0.831	0.844	0.880
Politics	0.786	0.801	0.809	0.844
Poker	0.662	0.739	0.794	0.838
Atheism	0.800	0.824	0.771	0.829
Television	0.595	0.634	0.704	0.736
Macro-Averaged	0.765	0.815	0.837	0.883

thread do not occur in separate folds. Duplication of hyperlinks across splits was also disallowed, so the metadata of an object cannot occur in multiple folds. These restrictions resulted in the omission of approximately 11% of the posts in each forum group. The same ten folds are used to test **C-L**, **C**, **M** and **C+M**.

The results of the classification experiments are shown in Table 2. We performed a paired t-test at the 0.05 level over the results of the cross-validation in order to assess statistical significance. For **General**, all differences in results are significant. Simply using the content without hyperlinks (**C-L**) gives quite good results, but incorporating URL information from hyperlinks (**C**) improves the results. Using only metadata from hyperlinked objects (**M**) gives improved performance, and combining post content with metadata from hyperlinks (**C+M**) gives the best performance. For **Music**, the scores in general are lower, likely due to the higher similarity of the topics. Here, the difference between **M** and **C+M** is not significant, but all other differences are significant. When a post has a hyperlink, object metadata, with or without post content, provides a better description of the topic of a post than the original post content.

Detailed results for **General** are shown in Table 3, arranged in order of descending F-score for **C+M**. It can be seen that there is a large variation in classification performance across different forums. For example, classification of a post in the Musicians forum is trivial, since almost all posts feature a link to MySpace. However classification of a Television forum post is more challenging, because this forum can also cover any topic which is televised. Despite the variation between topics, **C+M** always results in the best performance, although not all of these results are statistically significant.

## 4 Conclusion

We investigated the potential of using metadata from external objects for classifying the topic of posts in online forums. To perform this study, we augmented a message board corpus with metadata associated with hyperlinks from posts. Our experiments reveal that this external metadata has better descriptive power for topic classification than the original posts, and that a combination of both gives best results. We conclude that for those posts that contain hyperlinks for which structured data is available, the external metadata can provide valuable features for topic classification. We plan to continue this work by comparing the usefulness of object titles, descriptions, categories and tags as features for improving topic classification. Thus not only textual metadata content will be considered, but also the relationships linking them to the original object. The growing amount of structured data on the Web means that this type of semantically-rich information can be retrieved from a significant and increasing proportion of hyperlinks. Potential applications of the approach include suggesting appropriate forums for new posts, and categorising existing posts for improved browsing and navigation. The enhanced representation of a post as *content plus hyperlink metadata* also has potential for improving search in social media.

## References

1. Agichtein, E., Castillo, C., Donato, D., Gionis, A., Mishne, G.: Finding high-quality content in social media. In: Proceedings of WSDM (2008)
2. Angelova, R., Weikum, G.: Graph-based text classification: Learn from your neighbors. In: Proceedings of SIGIR (2006)
3. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: Dbpedia: A nucleus for a web of open data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
4. Berendt, B., Hanser, C.: Tags are not metadata, but “just more content”—to some people. In: Proceedings of ICWSM (2007)
5. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The story so far. International Journal on Semantic Web and Information Systems 5(3) (2009)
6. Figueiredo, F., Belém, F., Pinto, H., Almeida, J., Gonçalves, M., Fernandes, D., Moura, E., Cristo, M.: Evidence of quality of textual features on the Web 2.0. In: Proceedings of CIKM (2009)
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.: The WEKA data mining software: An update. ACM SIGKDD Exp. 11(1) (2009)
8. Kinsella, S., Passant, A., Breslin, J.G.: Using hyperlinks to enrich message board content with Linked Data. In: Proceedings of I-SEMANTICS (2010)
9. Qi, X., Davison, B.: Classifiers without borders: Incorporating fielded text from neighboring web pages. In: Proceedings of SIGIR (2008)
10. Sun, A., Suryanto, M., Liu, Y.: Blog classification using tags: An empirical study. In: Goh, D.H.-L., Cao, T.H., Sølvsberg, I.T., Rasmussen, E. (eds.) ICADL 2007. LNCS, vol. 4822, pp. 307–316. Springer, Heidelberg (2007)



# Peddling or Creating? Investigating the Role of Twitter in News Reporting

Ilija Subašić and Bettina Berendt

Department of Computer Science, K.U. Leuven, Belgium

**Abstract.** The widespread use of social media is regarded by many as the emergence of a new highway for information and news sharing promising a new information-driven “social revolution”. In this paper, we analyze how this idea transfers to the news reporting domain. To analyze the role of social media in news reporting, we ask whether citizen journalists tend to *create* news or *peddle* (re-report) existing content. We introduce a framework for exploring divergence between news sources by providing multiple views on corpora in comparison. The results of our case study comparing Twitter and other news sources suggest that a major role of Twitter authors consists of neither creating nor peddling, but extending them by *commenting* on news.

## 1 Introduction

On January 16, 2009, a US Airways airplane made an emergency landing on the Hudson river. First reports on this events were spread via social media web sites. Although the idea of “citizen journalism” was present much before this incident took place, it has provided a major boost to citizen journalism platforms, placing them shoulder to shoulder with “traditional” news outlets. By some [16], this new way of discovering news is hailed as a beginning of a “social revolution” driven by information sharing, promising stronger social action and making *vox populi* a more important factor in all spheres of life. However, some researchers [9,7] have expressed doubts about such a social-media-led revolution. Transferring the same principles and contrasting standpoints to the news reporting domain, one could expect that social media either have a great potential for introducing and spreading new information, or alternatively serve solely as a channel for spreading the content produced by traditional media. Thus, is the (main) role of citizen journalists to *create* news or rather to *peddle* (re-report) existing content? In this paper, we aim to provide some insight into this question by defining a set of corpora-similarity measures on corpora created from Twitter and other news sources. The main idea of using corpora similarity is that higher similarity would suggest “peddling”, while lower would suggest originality and “creation”.

There has been substantial research into discovering the point of origin of a news story [8] and into the dynamics of content between news and blogs, e.g. [14]. We take a different approach: we start with news story already “discovered” and investigate whether social media provides a *different* reporting to traditional

media. We start by collecting corpora containing documents on the same news story originating from different sources. Our goal is to analyze differences between corpora by providing a multi-aspect view on similarity. Some news stories describe breaking events or spotlighted topics. These stories are often referred to as “breaking news”. We broaden our analysis and investigate whether during a breaking event reporting converges across sources.

The main contribution of this paper is a framework for comparing social media with traditional media that provides: (a) multiple-aspect corpora difference measures, (b) analysis of social vs. traditional media content, and (c) aggregation and visualization of news sources relations. We complement the framework (Section 3) by a case study (4).

## 2 Related Work

**Twitter research.** Out of many areas of Twitter research, we focus on the ones related to news mining. [10,18,9] investigate user motivation behind twittering. All of these studies report on news sharing as one of the main motivations for Twitter use. Studies of the role of news medium in influence spread [3,2] found that traditional news sources and celebrity-owned Twitter accounts were among the most influential posts. In contrast to these works, our objective is to detect the differences between news reports covering the same story on Twitter and other media.

**Corpora and text similarity.** Similarity between texts has been a long-standing topic in different fields producing a wide range of text similarity measures. [1] provides a valuable overview of different text similarity measures. Work in corpus linguistics [11] compares text similarity metrics on a corpus scale. This work introduces a  $\chi^2$ -test based model of corpus similarity and compares it with the probabilistic similarity measures perplexity [5] and mutual information [4]. Another family of probabilistic similarity measures, based on Kullback-Leibler ( $KL$ ) divergence [12], has been widely used in different domains as a measure of text similarity. It is used for measuring similarity between queries and documents in information retrieval [13], for detecting plagiarism in Wikipedia articles [1], and for comparing traditional and Open Access medical journals content [17]. We adopt a KL-based approach to corpora similarity, but provide multiple perspectives on similarity by combining several aspects of the corpora.

## 3 Measures of Corpora Divergence

**Notation.** A corpus  $C_{source}^{story}$  is a set of documents covering the same news *story* collected from a single *source* (e.g. Twitter, AP) or a family of sources (e.g. blogs). A representation of a corpus  $C_{source}^{story}$  is a language model  $\Theta_{source}^{story}$ , where the probability of a token  $t$  is denoted as  $\Theta_{source}^{story}(t)$ . We define token categories as: (1) plain words ( $pw$ ) - words in a document; (2) headline words ( $hw$ ) - words in headlines; (3) entity words ( $ew$ ): words referring to semantic entities (names, locations, companies, ...); and (4) sentiment words ( $sw$ ): words expressing

sentiment.  $\Theta_{source|category}^{story}$  denotes the category language model (e.g. for a unigram model,  $\Theta_{source|hw}^{story}$  are the probabilities of headline words).

**Divergence measures.** Among many different language models we choose the unigram model as it fits the writing style of tweets. Given two corpora from sources  $a$  and  $b$  covering a story  $x$ , we use a symmetrical variant of  $KL$  divergence, the Jensen-Shannon divergence ( $JS$ ), between their language models  $\Theta_a^x$  and  $\Theta_b^x$  to measure their distance as:  $JS(\Theta_a^x, \Theta_b^x) = \frac{1}{2}KL(\Theta_a^x \Theta_m^x) + \frac{1}{2}KL(\Theta_b^x, \Theta_m^x)$  where the probability of every  $t$  in  $\Theta_m^x$  is the average probability of  $t$  in  $\Theta_a^x$  and  $\Theta_b^x$ . We define a set of measures differing by token categories.

*Language divergence (LD).* The first measure we define covers the entire content of the corpora. In other words, we build a language model using  $pw$ . The reason for this is to capture stylistic, terminological, and content differences of sources. We define language divergence ( $LD$ ) of two sources  $a$  and  $b$  reports on a story  $x$  as:  $LD_x^{a,b} = JS(\Theta_{a|pw}^x, \Theta_{b|pw}^x)$ . Due to many differences in the format and length between documents between corpora, using this measure mostly captures differences in writing styles and vocabulary between sources.

*Headline divergence (HD).* Headlines in traditional news summarize their articles; they are a standard unit of analysis in media studies. Tweets have no substructure and are at most 140 characters long, making them their own headlines ( $hw = pw$ ). We define the headline divergence as:  $HD_x^{a,b} = JS(\Theta_{a|hw}^x, \Theta_{b|hw}^x)$ . Using headlines to measure difference between reports in social and traditional media tackles problems of style and length used among sources. However, it still does not take into account the semantic difference between reports.

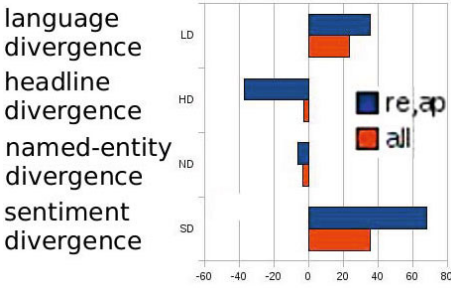
*Named-entity divergence (ND).* News stories revolve around different subjects, places, and organizations they describe or “feature”. We introduce a semantics divergence measure as:  $ND_x^{a,b} = JS(\Theta_{a|ew}^x, \Theta_{b|ew}^x)$ . Named entities carry semantically rich information conveyed by the reports, but fail to capture the position of the reporters towards the story. News texts are often more than reporting, and express opinions and sentiments towards the story.

*Sentiment divergence (SD).* We therefore define a last measure based on the differences in used sentiment words. Since many words used to express sentiment are rarely used and the probability of observing them in a corpus is low, we follow the approach described in [6] and bin  $sw$  tokens into 7 categories of strength and type of the sentiment they express (ranging from strong negative to strong positive). Therefore,  $SD$  measures differences in probability distributions over the categories of sentiment, and not sentiment-bearing words:  $SD_x^{a,b} = JS(\Theta_{a|bin(sw)}^x, \Theta_{b|bin(sw)}^x)$ .

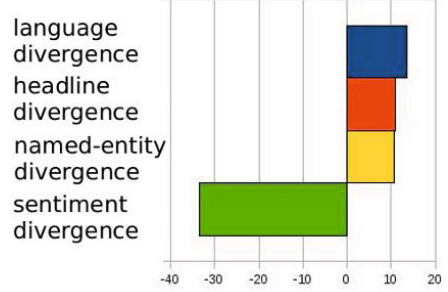
To be able to relate more than two sources to one another and to abstract from the (non-interpretable) absolute values of  $JS$ , we apply multidimensional scaling, projecting the obtained distance matrices into two dimensions.

## 4 Case Study

We present a case study comparing news reports from Twitter ( $tw$ ), blogosphere ( $bl$ ), professional news outlets ( $nw$ ), Reuters ( $rt$ ) and Associated Press ( $ap$ ).



**Fig. 1.** Average  $RD$  for all stories comparing divergences between **Twitter** and other sources with *all* and *re, ap* baselines

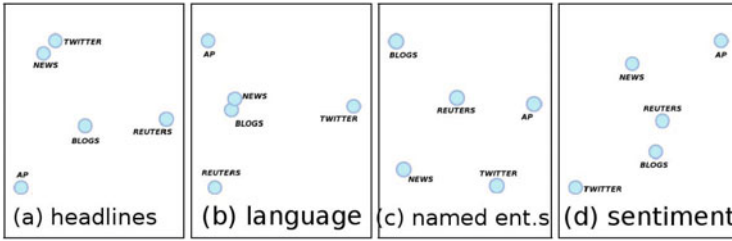


**Fig. 2.** Average  $RD$  for **breaking stories** with the *non-breaking stories* baseline

**Corpora and procedure.** We obtained the story reports using the same query across different sources’ search engines. For news and blogs, we used Google News and Blog search to harvest web pages, and extracted content as described in [15]. We collected 3 breaking stories covering the BP oil spill, the Pakistan floods in 2010, and the Chilean miners’ accident, and 3 non-breaking stories about Belgian politics, Iraq, and the European Union. As an indicator of breaking stories we used Twitter’s trending topic list. The upper bound of corpus size was the number of tweets for the respective story, and the lower bound was set to 50. For collected documents we extracted named entities with Open Calais (www.openalais.com), removed stop words, and lemmatized the rest.

**Relative divergence aggregation.** The absolute values of  $KL$  based measures have no clear interpretation; we therefore concentrated on values relative to a baseline. We defined 3 baseline divergences. The first one (*all*) averages over all pair-wise divergences across all sources. The second base divergence value (*re, ap*) is the divergence between Reuters and Associated Press corpora. Due to the same type of media, format, and reporting style, we consider this as a reasonable baseline. To compare breaking and non-breaking stories, we used the average divergence for non-breaking stories over all sources as a baseline. We denote these 3 value as  $baseline_{(all;re,ap;breaking)}$ . The relative divergence ( $RD$ ) of a source  $a$  for a story  $x$  is then:  $RD_a^x = ((avg_{b \in sources} D_{a,b}^x - baseline_x) / baseline_x) \times 100$ .

**Results.** Figure 1 shows the results of applying  $RD$  to Twitter. We start with the interpretation of the *re, ap* baseline. The largest relative divergence is for sentiment divergence. The  $RD$  value of 67.87 shows higher sentiment divergence between Twitter and other sources. This result can lead to two conclusions: (a) Twitter contains more contrasting sentiment than news-wire reports, and (b) Twitter expresses more sentiment than news-wire reports. To decide between these two, we calculated the share of sentiment words across sources. In *ap* corpora, there are 1.7% sentiment words, in *re* corpora 2.8%, and in *tw* corpora 4.2%. We find that both the share and the type of sentiment words influence the differences between corpora. For example, strongly negative words make up



**Fig. 3.** MDS maps of divergence measures: (a)  $HD$ , (b)  $LD$ , (c)  $ND$ , (d)  $SD$

0.17% of the *ap* and 0.9% of the *tw* corpora. For other categories, it is expected that language divergence ( $LD$ ) has a positive value, because more authors in Twitter use different writing styles, wording, and non-standard grammar. This is partly shown by the average number of unique language-words tokens (10221 in *ap*, 13122 in *re*, and 89619 in *tw*). The high positive value of  $HD$  can be explained by the differences in the sizes of the different corpora, where a small number of documents in news-wire agencies do not converge to the same headline words. On average for a story, we collected 69 documents from news-wire agencies and 3314 from Twitter. The lowest  $RD$  value ( $-6.13$ ) is found for named-entity divergences. This points to the conclusion that all sources are reporting on the same entities, but using different language and sentiment.

The difference between average  $RD$  values for the *all* baseline is always lower than for the *re*, *ap* baseline, on average  $\approx 19\%$  lower. The extreme case is the difference of headline divergence values, which is 12.9 times lower when using the *all* baseline. We see this as an effect of having more documents to compare tweets to, because the average number of documents across all sources is 451, and due to many similar titles the divergence measure converges across sources. The lower  $RD$  values for *all* compared to *re*, *ap* suggest that Twitter is more similar to other sources than to news agencies.

Figure 2 shows  $RD$  values that describe the difference between breaking and non-breaking news. As a baseline divergence, we used the *non-breaking* value, comparing the average of the *breaking* stories to it. Positive values of  $RD$  reflect a higher divergence of reporting for breaking news. The figure 2 shows that breaking news is consistently more different across sources, except for sentiment divergence ( $SD$ ). This suggests that for breaking news, informing the readers about the story is the main objective of the authors, while for non-breaking stories authors express their standpoints and analysis of the story.

To further investigate these differences and see which divergence contributes most, consider the MDS plots in Fig. 3. Figure 3(a) shows that the headline distance between reports in News and Twitter is the lowest. Many news-related tweets come from Twitter accounts operated by professional news outlets [2]. In our dataset, we found an average of 2.9% identical entries in the *tw* and *nw* corpora. Figure 3(b) shows that Twitter uses language closer to news and blogs than news-wire agencies, while news and blogs use similar language when compared to other sources. In terms of named entities (Figure 3(c)), *re* corpora

are far from other sources. This probably arises from the much number of named entities used by Reuters: an average per-document of 21.9 (compared to 0.22 in *tw*, 8.6 in *bl*, 12.91 in *nw*, and 13.2 in *ap*) Figure 3(d) visualizes sentiment divergence, showing that *bl*, *re*, and *nw* corpora contain similar amounts of sentiment, which are more different when compared to *tw* and *ap* corpora.

## 5 Conclusions and Outlook

This work is our starting effort in defining an easily interpretable, multi-aspect similarity measures for comparing news sources. Of course, this work cannot cover all the possible or interesting aspects of divergence in news reports, and absolute values of divergence measures are hard to interpret. Nonetheless, as the paper has shown, the inspection of relative differences can give interesting insights, opening many interesting research directions.

We started this investigation by focusing on two roles of social media platforms: to create new and different news, or to peddle or spread existing news. In contrast to both, our results suggest that the biggest role of citizen journalists in news is the role of a *commentator*, not only reporting but expressing opinions and taking positions on the news. Investigating this role will yield further measures of relations between corpora and a deeper understanding of the dynamics of social media in today's news environment.

## References

1. Barrón-Cedeño, A., Eiselt, A., Rosso, P.: Monolingual Text Similarity Measures: A Comparison of Models over Wikipedia Articles Revisions. In: Proc. of ICON 2009, pp. 29–38. Macmillan, Basingstoke (2009)
2. Cha, M., Haddadi, H., Benevenuto, F., Gummadi, K.P.: Measuring User Influence in Twitter: The Million Follower Fallacy. In: Proc. of ICSWM 2009. AAAI, Menlo Park (2009)
3. Cha, M., Haddadi, H., Benevenuto, F., Gummadi, K.P.: Measuring user influence in twitter: The million follower fallacy. In: Proc. of ICWSM 2010. AAAI, Menlo Park (2010)
4. Church, K.W., Hanks, P.: Word association norms, mutual information, and lexicography. *Computational Linguistics* 16(1), 22–29 (1990)
5. English, E.O., Brown, P.E., Pietra, V.J.D., Mercer, R.L., Pietra, S.A.D., Lai, J.C.: An estimate of an upper bound for the entropy of English. *Computational Linguistics* 18, 31–40 (1992)
6. Esuli, A., Sebastiani, F.: Sentiwordnet: A publicly available lexical resource for opinion mining. In: Proc. of LREC 2006. LREC (2006)
7. Gladwell, M.: Small change: Why the revolution will not be tweeted (2010). *New Yorker Magazine* (October 2010)
8. Grossman, L.: Iran protests: Twitter, the medium of the movement. *Time Magazine* (June 2009), <http://www.time.com/time/world/article/0,8599,1905125,00.html>
9. Huberman, B.A., Romero, D.M., Wu, F.: Social networks that matter: Twitter under the microscope. ArXiv e-prints (December 2008), <http://arxiv.org/abs/0812.1045>

10. Java, A., Song, X., Finin, T., Tseng, B.: Why we twitter: understanding microblogging usage and communities. In: Proc. of WebKDD/SNA-KDD 2007, pp. 56–65. ACM, USA (2007)
11. Kilgarriff, A.: Comparing corpora. *International Journal of Corpus Linguistics* 6, 1–37 (2001)
12. Kullback, S., Leibler, R.A.: On information and sufficiency. *The Annals of Mathematical Statistics* 22(1), 79–86 (1951)
13. Lafferty, J., Zhai, C.: Probabilistic relevance models based on document and query generation. In: *Language Modeling and Information Retrieval*, pp. 1–10. Kluwer, Dordrecht (2002)
14. Leskovec, J., Backstrom, L., Kleinberg, J.M.: Meme-tracking and the dynamics of the news cycle. In: Proc. of KDD 2009, pp. 497–506. ACM, New York (2009)
15. Prasad, J., Paepcke, A.: Coreex: content extraction from online news articles. In: Proc. of CIKM 2008, pp. 1391–1392. ACM, New York (2008)
16. Shirky, C.: How social media can make history. TED Talk (June 2009), [http://www.ted.com/talks/clay\\_shirky\\_how\\_cellphones\\_twitter\\_facebook\\_can\\_make\\_history.html](http://www.ted.com/talks/clay_shirky_how_cellphones_twitter_facebook_can_make_history.html)
17. Verspoor, K., Cohen, K.B., Hunter, L.: The textual characteristics of traditional and open access scientific journals are similar. *BMC bioinformatics* 10(1), 183+ (2009)
18. Zhao, D., Rosson, M.B.: How and why people twitter: the role that micro-blogging plays in informal communication at work. In: Proc. of GROUP 2009, pp. 243–252. ACM, New York (2009)

# Latent Sentiment Model for Weakly-Supervised Cross-Lingual Sentiment Classification

Yulan He

Knowledge Media Institute, The Open University  
Walton Hall, Milton Keynes MK7 6AA, UK  
y.he@open.ac.uk

**Abstract.** In this paper, we present a novel weakly-supervised method for cross-lingual sentiment analysis. In specific, we propose a latent sentiment model (LSM) based on latent Dirichlet allocation where sentiment labels are considered as topics. Prior information extracted from English sentiment lexicons through machine translation are incorporated into LSM model learning, where preferences on expectations of sentiment labels of those lexicon words are expressed using generalized expectation criteria. An efficient parameter estimation procedure using variational Bayes is presented. Experimental results on the Chinese product reviews show that the weakly-supervised LSM model performs comparably to supervised classifiers such as Support vector Machines with an average of 81% accuracy achieved over a total of 5484 review documents. Moreover, starting with a generic sentiment lexicon, the LSM model is able to extract highly domain-specific polarity words from text.

**Keywords:** Latent sentiment model (LSM), cross-lingual sentiment analysis, Generalized expectation, latent Dirichlet allocation.

## 1 Introduction

The objective of sentiment analysis is to automatically extract opinions, emotions and sentiments in text. It allows us to track attitudes and feelings on the web. Research in sentiment analysis has mainly focused on the English language. Little work has been carried out in other languages partly due to the lack of resources, such as subjectivity lexicons consisting of a list of words marked with their respective polarity (positive, negative or neutral) and manually labeled subjectivity corpora with documents labeled with their polarity.

Pilot studies on cross-lingual sentiment analysis utilize machine translation to perform sentiment analysis on the English translation of foreign language text [10,12,17]. These supervised learning algorithms suffer from the generalization problem since annotated data (either annotated English corpora or the translated corpora generated by machine translation) are required for classifier training. As such, they often fails when there is a domain mismatch between the source and target languages. Recent effort has been made to exploit bootstrapping-style approaches for weakly-supervised sentiment classification in languages other than English [19,18,12]. Other approaches use ensemble techniques by either combining lexicon-based and corpus-based algorithms [15] or



combining sentiment classification outputs from different experimental settings [16]. Nevertheless, all these approaches are either complex or require careful tuning of domain and data specific parameters.

This paper proposes a weakly-supervised approach for cross-lingual sentiment classification by incorporating lexical knowledge obtained from available English sentiment lexicons through machine translation. Preferences on expectations of sentiment labels of those lexicon words are expressed using generalized expectation criteria [9] and are used to modify the latent Dirichlet allocation (LDA) model objective function for model learning, which we named as latent sentiment model (LSM). The proposed approach performs sentiment analysis without the use of labeled documents. In addition, it is simple and computationally efficient; rendering more suitable for online and real-time sentiment classification from the Web.

Incorporating sentiment prior knowledge into LDA model for sentiment analysis has been previously studied in [8,7] where the LDA model has been modified to jointly model sentiment and topic. However their approach uses the sentiment prior information in the Gibbs sampling inference step that a sentiment label will only be sampled if the current word token has no prior sentiment as defined in a sentiment lexicon. This in fact implies a different generative process where many of the 1's are observed. The model is no longer "latent". Our proposed approach incorporate sentiment prior knowledge in a more principled way that we express preferences on expectations of sentiment labels of the lexicon words from a sentiment lexicon using generalized expectation criteria and essentially create an informed prior distribution for the sentiment labels. This would allow the model to actually be latent and would be consistent with the generative story.

We have explored several commonly used English sentiment lexicons and conducted extensive experiments on the Chinese reviews of four different product types. The empirical results show that the LSM model, despite using no labeled documents, performs comparably to the supervised classifiers such as Support Vector Machines (SVMs) trained from labeled corpora. Although this paper primarily studies sentiment analysis in Chinese, the proposed approach is applicable to any other language so long as a machine translation engine is available between the selected language and English.

The remainder of the paper is structured as follows. Related work on cross-lingual and weakly-supervised sentiment classification in languages other than English are discussed in Section 2. Existing algorithms on incorporating supervised information into LDA model learning are also reviewed in this section. The proposed LSM model and its inference and training procedures are presented in Section 3. The experimental setup and results of sentiment classification on the Chinese reviews of four different products are presented in Section 4 and 5 respectively. Finally, Section 6 concludes the paper.

## 2 Related Work

Early work on cross-lingual sentiment analysis rely on English corpora for subjectivity classification in other languages. For example, Mihalcea et al. [10] make use of a bilingual lexicon and a manually translated parallel text to generate the resources

to build subjectivity classifiers based on SVMs and Naïve Bayes (NB) in a new language; Banea et al. [11] use machine translation to produce a corpus in a new language and train SVMs and NB for subjectivity classification in the new language. Bautin et al. [12] also utilize machine translation to perform sentiment analysis on the English translation of a foreign language text. More recently, Wan [17] proposed a co-training approach to tackle the problem of cross-lingual sentiment classification by leveraging an available English corpus for Chinese sentiment classification. The major problem of these cross-lingual sentiment analysis algorithms is that they all utilize supervised learning to train sentiment classifiers from annotated English corpora (or the translated target language corpora generated by machine translation). As such, they cannot be generalized well when there is a domain mismatch between the source and target language.

Recent efforts have also been made for weakly-supervised sentiment classification in languages other than English. Zagibalov and Carroll [19][18] starts with a one-word sentiment seed vocabulary and use iterative retraining to gradually enlarge the seed vocabulary by adding more sentiment-bearing lexical items based on their relative frequency in both the positive and negative parts of the current training data. Sentiment direction of a document is then determined by the sum of sentiment scores of all the sentiment-bearing lexical items found in the document. Qiu et al. [12] also uses a lexicon-based iterative process as the first phase to iteratively enlarge an initial sentiment dictionary. Documents classified by the first phase are taken as the training set to train the SVMs which are subsequently used to revise the results produced by the first phase. Wan [16] combined sentiment scores calculated from Chinese product reviews using the Chinese HowNet sentiment dictionary<sup>1</sup> and from the English translation of Chinese reviews using the English MPQA subjectivity lexicon<sup>2</sup>. Various weighting strategies were explored to combine sentiment classification outputs from different experimental settings in order to improve classification accuracy. Nevertheless, all these weakly-supervised sentiment classification approaches are rather complex and require either iterative training or careful tuning of domain and data specific parameters, and hence unsuitable for online and real-time sentiment analysis in practical applications.

In recent years, there have been increasing interests in incorporating supervised information into LDA model learning. Blei and McAuliffe [3] proposed supervised LDA (sLDA) which uses the empirical topic frequencies as a covariant for a regression on document labels such as movie ratings. Mimno and McCallum [11] proposed a Dirichlet-multinomial regression which uses a log-linear prior on document-topic distributions that is a function of observed features of the document, such as author, publication venue, references, and dates. DiscLDA [6] and Labeled LDA [13] assume the availability of document class labels and utilize a transformation matrix to modify Dirichlet priors. DiscLDA introduces a class-dependent linear transformation to project a  $K$ -dimensional ( $K$  latent topics) document-topic distribution into a  $L$ -dimensional space ( $L$  document labels), while Labeled LDA simply defines a one-to-one correspondence between LDA's latent topics and document labels. Our work differs from theirs in that we use word prior sentiment as supervised information and modify the LDA objective function by adding the generalized expectation criteria terms.

---

<sup>1</sup> <http://www.keenage.com/download/sentiment.rar>

<sup>2</sup> <http://www.cs.pitt.edu/mpqa/>

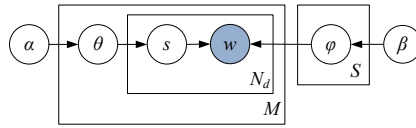


Fig. 1. Latent sentiment model

### 3 Latent Sentiment Model

Unlike existing approaches, we view sentiment classification as a generative problem that when an author writes a review document, he/she first decides on the overall sentiment or polarity (positive, negative, or neutral) of a document, then for each sentiment, decides on the words to be used. The LSM model, as shown in Figure 1, can be treated as a special case of LDA where a mixture of only three sentiment labels are modeled, i.e. positive, negative and neutral.

Assuming that we have a total number of  $S$  sentiment labels  $\mathcal{S} = \{neutral, positive, negative\}$ ; a corpus with a collection of  $M$  documents is denoted by  $\mathcal{D} = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_M\}$ , where the bold-font variables denote the vectors; each document in the corpus is a sequence of  $N_d$  words denoted by  $\mathbf{w} = (w_1, w_2, \dots, w_{N_d})$ , and each word in the document is an item from a vocabulary index with  $V$  distinct terms denoted by  $\{1, 2, \dots, V\}$ . The generative process is to first draw  $\varphi_s \sim \text{Dir}(\beta)$ , then for each document  $d \in [1, M]$ , choose a distribution  $\theta_d \sim \text{Dir}(\alpha)$ , and for each of the  $N_d$  word position  $w_t$ , sample a sentiment label  $s_t \sim \text{Multinomial}(\theta_d)$  and choose a word  $w_t \sim \text{Multinomial}(\varphi_{s_t})$ .

Letting  $\Lambda = \{\alpha, \beta\}$ , we obtain the marginal distribution of a document  $\mathbf{w}$  by integrating over  $\theta$  and  $\varphi$  and summing over  $s$ :

$$P(\mathbf{w}|\Lambda) = \int \int P(\theta; \alpha) \prod_{s=1}^S P(\varphi_s; \beta) \prod_{t=1}^{N_d} \sum_{s_t} p(s_t|\theta) P(w_t|s_t, \varphi_{s_t}) d\theta d\varphi \quad (1)$$

Taking the product of marginal probabilities of documents in a corpus gives us the probability of the corpus.

$$P(\mathcal{D}|\Lambda) = \prod_{d=1}^M P(\mathbf{w}_d|\Lambda) \quad (2)$$

Assume we have some labeled features where words are given with their prior sentiment orientation, we could construct a set of real-valued features of the observation to express some characteristic of the empirical distribution of the training data that should also hold of the model distribution.

$$f_{jk}(w, s) = \sum_{d=1}^M \sum_{t=1}^{N_d} \delta(s_{d,t} = j) \delta(w_{d,t} = k) \quad (3)$$

where  $\delta(x)$  is an indicator function which takes a value of 1 if  $x$  is true, 0 otherwise. Equation 3 calculates how often feature  $k$  and sentiment label  $j$  co-occur in the corpus.

We define the expectation of the features as

$$E_\Lambda[\mathbf{f}(w, s)] = E_{\tilde{P}(w)}[E_{P(w|s;\Lambda)}[\mathbf{f}(w, s)]] \tag{4}$$

where  $\tilde{P}(w)$  is the empirical distribution of  $w$  in document corpus  $\mathcal{D}$ , and  $P(w|s; \Lambda)$  is a conditional model distribution parameterized at  $\Lambda$ .

$E_\Lambda[\mathbf{f}(w, s)]$  is a matrix of size  $S \times K$  where  $S$  is the total number of sentiment labels and  $K$  is the total number of features or constraints used in model learning. The  $jk$ th entry denotes the expected number of times that feature  $k$  is assigned with label  $j$ .

We define a criterion that minimizes the KL divergence of the expected label distribution and a target expectation  $\hat{\mathbf{f}}$ , which is essentially an instance of generalized expectation criteria that penalizes the divergence of a specific model expectation from a target value.

$$G(E_\Lambda[\mathbf{f}(w, s)]) = KL(\hat{\mathbf{f}} || E_\Lambda[\mathbf{f}(w, s)]) \tag{5}$$

We can use the target expectation  $\hat{\mathbf{f}}$  to encode human or task prior knowledge. For example, the word “*excellent*” typically represent a positive orientation. We would expect that this word more likely appears in positive documents. In our implementation, we adopted a simple heuristic approach [144] that a majority of the probability mass for a feature is distributed uniformly among its associated labels, and the remaining probability mass is distributed uniformly among the other non-associated label(s). As we only have three sentiment labels here, the target expectation of a feature having its prior polarity (or associated sentiment label) is 0.9 and 0.05 for its non-associated sentiment labels.

The above encodes word sentiment prior knowledge in the form of  $\hat{P}(s|w)$ . However, the actual target expectation used in our approach is  $\hat{P}(w|s)$ . We could perform the following simple transformation:

$$\hat{P}(w|s) = \frac{\hat{P}(s|w)P(w)}{P(s)} \propto \hat{P}(s|w)\tilde{P}(w) \tag{6}$$

by assuming that the prior probability of  $w$  can be obtained from the empirical distribution of  $w$  in document corpus  $\mathcal{D}$ , and the prior probability of the three sentiment labels are uniformly distributed in the corpus.

We augment the likelihood maximization by adding the generalized expectation criteria objective function terms.

$$\mathcal{O}(\mathcal{D}|\Lambda) = \log P(\mathcal{D}|\Lambda) - \lambda G(E_\Lambda[\mathbf{f}(w, s)]) \tag{7}$$

where  $\lambda$  is a penalized parameter which controls the relative influence of the prior knowledge. This parameter is empirically set to 100 for all the datasets. For brevity, we omit  $\lambda$  in the subsequent derivations. The learning of the LSM model is to maximize the objective function in Equation 7. Exact inference on the LSM is intractable. We use the variational methods to approximate the posterior distribution over the latent variables. The variational distribution which is assumed to be fully factorized is:

$$q(\mathbf{s}, \boldsymbol{\theta}, \boldsymbol{\varphi}|\Omega) = \prod_{s=1}^S q(\varphi_s|\tilde{\beta}_s) \prod_{d=1}^M q(\theta_d|\tilde{\alpha}_d) \prod_{t=1}^N q(s_{dt}|\tilde{\gamma}_{dt})$$

where  $\Omega = \{\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}\}$  are free variational parameters,  $\theta \sim \text{Dirichlet}(\tilde{\alpha})$ ,  $\varphi \sim \text{Dirichlet}(\tilde{\beta})$ , and  $s_{dt} \sim \text{Multinomial}(\tilde{\gamma})$ .

We can bound the objective function in Equation 7 in the following way.

$$\mathcal{O}(\mathcal{D}|A) \geq E_q[\log P(\mathbf{w}, \mathbf{s}, \boldsymbol{\theta}, \boldsymbol{\varphi}|A) - G(E_\Lambda[\mathbf{f}(\mathbf{w}, \mathbf{s})])] - E_q[\log q(\mathbf{s}, \boldsymbol{\theta}, \boldsymbol{\varphi})] \quad (8)$$

By letting  $L(\Omega; A)$  denote the RHS of the above equation, we have:

$$\mathcal{O}(\mathcal{D}|A) = L(\Omega; A) + KL(q(\mathbf{s}, \boldsymbol{\theta}, \boldsymbol{\varphi}|\Omega)||P(\mathbf{s}, \boldsymbol{\theta}, \boldsymbol{\varphi}|A))$$

By maximizing the lower bound  $L(\Omega; A)$  with respect to  $\Omega$  is the same as minimizing the KL distance between the variational posterior probability and the true posterior probability.

Expanding the lower bound by using the factorizations of  $P$  and  $q$ , we have:

$$\begin{aligned} L(\Omega; A) = & E_q[\log P(\boldsymbol{\theta}|\alpha)] + E_q[\log P(\boldsymbol{\varphi}|\beta)] + E_q[\log P(\mathbf{s}|\boldsymbol{\theta})] + E_q[\log P(\mathbf{w}|\mathbf{s}, \boldsymbol{\varphi})] \\ & - E_q[\log q(\boldsymbol{\varphi})] - E_q[\log q(\boldsymbol{\theta})] - E_q[\log q(\mathbf{s})] - E_q[G(E_\Lambda[\mathbf{f}(w, \mathbf{s})])] \quad (9) \end{aligned}$$

The first seven terms are the same as in the LDA model. We show how to compute the last term in the above equation. For a sentiment label  $j$

$$\begin{aligned} E_q[G(E_\Lambda[\mathbf{f}(w, j)])] &= E_q\left[\sum_w \hat{f}_{jw} \log \frac{\hat{f}_{jw}}{E_\Lambda[\hat{f}_{jw}(w, j)]}\right] \\ &\leq \sum_w \hat{f}_{jw} (\log \hat{f}_{jw} - E_q\left[\sum_{d=1}^M \sum_{t=1}^{N_d} \log(P(w_{d,t}|s_{d,t}; \Lambda)\delta(s_{d,t} = j))\right]) \\ &= \sum_w \hat{f}_{jw} (\log \hat{f}_{jw} - \sum_{d=1}^M \sum_{t=1}^{N_d} \tilde{\gamma}_{d,t,s} \delta(s_{d,t} = j) (\Psi(\tilde{\beta}_{j,w}) - \Psi(\sum_{r=1}^V \tilde{\beta}_{j,r}))) \end{aligned}$$

We then employ a variational expectation-maximization (EM) algorithm to estimate the variational parameters  $\Omega$  and the model parameters  $\Lambda$ .

- (E-step): For each word, optimize values for the variational parameters  $\Omega = \{\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}\}$ . The update rules are

$$\tilde{\alpha}_{d,s} = \alpha + \sum_{t=1}^{N_d} \tilde{\gamma}_{d,t,s} \quad (10)$$

$$\tilde{\beta}_{s,v} = \beta + \sum_{d=1}^M \sum_{t=1}^{N_d} \delta(w_{d,t} = v) \tilde{\gamma}_{d,t,s} \quad (11)$$

$$\tilde{\gamma}_{d,t,s} = \begin{cases} \exp(\Psi(\tilde{\alpha}_{d,s}) + (1 + \hat{f}_{s,w_{d,t}})(\Psi(\tilde{\beta}_{s,w_{d,t}}) - \Psi(\sum_v \tilde{\beta}_{s,v}))) & \text{for labeled features} \\ \exp(\Psi(\tilde{\alpha}_{d,s}) + \Psi(\tilde{\beta}_{s,w_{d,t}}) - \Psi(\sum_v \tilde{\beta}_{s,v})) & \text{otherwise} \end{cases} \quad (12)$$

- (M-step): To estimate the model parameters, we maximize the lower bound on the log likelihood with respect to the parameters  $\Lambda = \{\alpha, \beta\}$ . There are no closed form solution for  $\alpha$  and  $\beta$  and an iterative searching algorithm is used to find the maximal values.

## 4 Experimental Setup

We conducted experiments on the four corpora<sup>3</sup> which were derived from product reviews harvested from the website IT1681<sup>4</sup> with each corresponding to different types of product reviews including mobile phones, digital cameras, MP3 players, and monitors. All the reviews were tagged by their authors as either positive or negative overall. The total number of review documents is 5484. Chinese word segmentation was performed on the four corpora using the conditional random fields based Chinese Word Segmenter<sup>5</sup>.

## 5 Experimental Results

This section presents the experimental results obtained using the LSM model with translated English lexicons tested on the Chinese product review corpora. The results are averaged over five runs using different random initialization.

### 5.1 Results with Different Sentiment Lexicons

We explored three widely used English sentiment lexicons in our experiments, namely the MPQA subjectivity lexicon, the appraisal lexicon<sup>6</sup>, and the SentiWordNet<sup>7</sup> [5]. For all these lexicons, we only extracted words bearing positive or negative polarities and discarded words bearing neutral polarity. For SentiWordNet, as it consists of words marked with positive and negative orientation scores ranging from 0 to 1, we extracted a subset of 8,780 opinionated words, by selecting those whose orientation strength is above a threshold of 0.6.

We used Google translator toolkit<sup>8</sup> to translate these three English lexicons into Chinese. After translation, duplicate entries, words that failed to translate, and words with contradictory polarities were removed. For comparison, we also tested a Chinese sentiment lexicon, NTU Sentiment Dictionary (NTUSD)<sup>9</sup> which was automatically generated by enlarging an initial manually created seed vocabulary by consulting two thesauri, the Chinese Synonym Thesaurus (tong2yi4ci2ci2lin2) and the Academia Sinica Bilingual Ontological WordNet 3.

Table 1 gives the classification accuracy results using the LSM model with prior sentiment label information provided by different sentiment lexicons. As for the individual lexicon, using the MPQA subjectivity lexicon outperforms the others on all the four corpora. In fact, it performs even better than the Chinese sentiment lexicon NTUSD. The

<sup>3</sup> <http://www.informatics.sussex.ac.uk/users/tz21/dataZH.tar.gz>

<sup>4</sup> <http://product.it168.com>

<sup>5</sup> <http://nlp.stanford.edu/software/stanford-chinese-segmenter-2008-05-21.tar.gz>

<sup>6</sup> [http://lingcog.iit.edu/arc/appraisal\\_lexicon\\_2007b.tar.gz](http://lingcog.iit.edu/arc/appraisal_lexicon_2007b.tar.gz)

<sup>7</sup> <http://sentiwordnet.isti.cnr.it/>

<sup>8</sup> <http://translate.google.com>

<sup>9</sup> <http://nlg18.csie.ntu.edu.tw:8080/opinion/pub1.html>

**Table 1.** Sentiment classification accuracy (%) by LSM with different sentiment lexicon

<i>Lexicon</i>	<i>Mobile</i>	<i>DigiCam</i>	<i>MP3</i>	<i>Monitors</i>	<i>Average</i>
(a) MPQA	80.95	78.65	81.85	79.91	80.34
(b) Appraisal	79.76	70.54	75.84	72.89	74.76
(c) SentiWN	76.06	66.90	75.23	70.28	72.12
(d) NTUSD	80.10	74.17	78.41	79.71	78.10
(a)+(b)	77.20	74.13	77.56	76.93	76.46
(a)+(d)	82.03	80.18	81.03	82.40	81.41
(a)+(b)+(d)	79.21	78.91	77.25	80.85	79.06

above results suggest that in the absence of any Chinese sentiment lexicon, the translated MPQA subjectivity lexicon can be used to provide sentiment prior information to the LSM model for cross-lingual sentiment classification.

We also conducted experiments by enlarging the MPQA subjectivity lexicon through adding unseen lexical terms from the Appraisal lexicon and NTUSD. We found that the enlargement of a sentiment lexicon does not necessarily lead to the improvement of classification accuracy. In particular, adding new lexical terms from the Appraisal lexicon hurts the classification performance. However, adding extra lexical terms from NTUSD gives the best overall classification accuracy with 81.41% being achieved. Thus, in all the subsequent experiments, the sentiment prior knowledge was extracted from the combination of MPQA subjectivity lexicon and NTUSD.

## 5.2 Comparison with Other Models

We compare our proposed approach with several other methods as described below:

- *Lexicon labeling.* We implemented a baseline model which simply assigns a score +1 and -1 to any matched positive and negative word respectively based on a sentiment lexicon. A review document is then classified as either positive or negative according to the aggregated sentiment score. Thus, in this baseline model, a document is classified as positive if there are more positive words than negative words in the document and vice versa.
- *LDA.* We evaluated sentiment classification performance with the LDA model where the number of topics were set to 3 corresponding to the 3 sentiment labels.
- *LDA init with prior.* The word prior polarity information obtained from a sentiment lexicon is incorporated during the initialization stage of LDA model learning. Each word token in the corpus is compared against the words in a sentiment lexicon. The matched word token get assigned its prior sentiment label. Otherwise, it is assigned with a randomly selected sentiment label.
- *LSM with random init.* The LSM model is trained with random initialization. That is, the word prior sentiment information is only incorporated by modifying the LDA objective function.
- *LSM init with prior.* Similar to *LDA init with prior*, the word prior polarity information is also used to initialize the LSM model.

Table 2 shows the classification accuracy results on the four corpora using different models. It can be observed that *Lexicon labeling* achieves the accuracy in the range

**Table 2.** Sentiment classification accuracy (%) using different models

<i>Corpus</i>	<i>Lexicon Labeling</i>	<i>LDA</i>	<i>LDA init with prior</i>	<i>LSM with random init</i>	<i>LSM init with prior</i>	<i>Naïve Bayes</i>	<i>SVM</i>
Mobile	68.48	54.88	62.88	74.17	82.03	86.52	84.49
DigiCam	71.70	58.86	62.36	65.91	80.18	82.27	82.04
MP3	70.44	63.11	70.40	74.48	81.03	82.64	79.43
Monitor	71.11	68.82	69.08	81.11	82.40	84.21	83.87
Average	70.43	61.42	66.18	73.92	81.41	84.41	82.46

of 68-72% with an average of 70.43% obtained over all the four corpora. LDA model without incorporating any sentiment prior information performs quite poorly with its accuracy being only better than random classification. An improvement is observed if the prior sentiment knowledge is incorporated during the initialization stage of LDA model learning. Still, the results are worse than the simple *Lexicon labeling*. For the LSM model, if the prior sentiment knowledge is only used to modify the LDA objective function, it improves upon *LDA init with prior* 4-13%. By additionally incorporating the prior sentiment information into the initialization stage of model learning, LSM outperforms all the other models with the best accuracy of 81.41% being achieved. Compared to *Lexicon labeling*, the improvement obtained by *LSM init with prior* ranges between 11% and 14% and this roughly corresponds to how much the model learned from the data. We can thus speculate that LSM is indeed able to learn the sentiment-word distributions from data.

For comparison purposes, we list the 10-fold cross validation results obtained using the supervised classifiers, Naïve Bayes and SVMs, trained on the labeled corpora [18]. It can be observed that the weakly-supervised LSM model performs comparably to SVMs and is only slightly worse than Naïve Bayes on the Chinese product review corpora despite using no labeled documents.

### 5.3 Impact of Prior Information

To further investigate the impact of the prior information on model learning, we plot the classification accuracy versus the EM iterations. Figure 2 shows the results on the four corpora. We notice that accuracies of all the other three models except *LDA init with prior* improve with the increasing number of EM iterations. Both *LSM with random init* and *LSM init with prior* converge quite fast, with the best classification accuracy results being achieved after six iterations. *LDA with random init* takes longer time to converge that it gives the best result after 20-26 iterations. The accuracy of *LDA init with prior* reaches the peak after 4 to 8 iterations and then gradually drops. This is more noticeable on Mobile and DigiCam where the accuracy drops about 17% and 11% respectively from the peak. This shows that incorporating prior information only at the initialization stage is not effective since the model is likely to migrate from the initialization state with the increasing number of iterations and thus results in degraded performance. Using the word prior sentiment knowledge to modify the LDA objective function, *LSM with random init* improves over the best results of *LDA init with prior*



by 3-14% and it gives more stable results. By incorporating the prior information in both the initialization stage and objective function modification, *LSM init with prior* gives the best results among all the four models.

#### 5.4 Domain-Specific Polarity-Bearing Words

While a generic sentiment lexicon provides useful prior knowledge for sentiment analysis, the contextual polarity of a word may be quite different from its prior polarity. Also, the same word might have different polarity in different domain. For example, the word “compact” might express positive polarity when used to describe a digital camera, but it could have negative orientation if it is used to describe a hotel room. Thus, it is worth to automatically distinguish between prior and contextual polarity. Our proposed LSM model is able to extract domain-specific polarity-bearing words. Table 3 lists some of the polarity words identified by the LSM model which are not found in the original sentiment lexicons. We can see that LSM is indeed able to recognize domain-specific positive or negative words, for example, 人性化 (user-friendly) for mobile phones, 小巧 (compact) for digital cameras, 金属 (metallic) and 杂音 (noise) for MP3, 清晰 (in focus) and 漏光 (leakage of light) for monitors.

The iterative approach proposed in [18] can also automatically acquire polarity words from data. However, it appears that only positive words were identified by their approach. Our proposed LSM model can extract both positive and negative words and most of them are highly domain-salient as can be seen from Table 3.

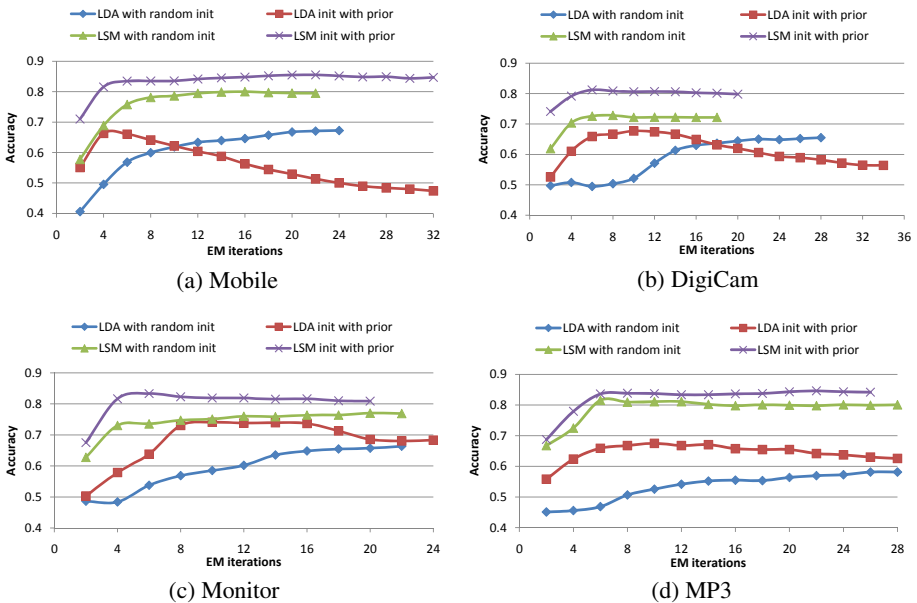


Fig. 2. Classification accuracy vs EM iterations

**Table 3.** Extracted example polarity words by LSM

<i>Corpus</i>	<i>Extracted Polarity Words</i>
Mobile	<p>Pos 不错 (not bad;pretty good), 人性化 (user-friendly), 时尚 (fashionable), 好用 (easy to use), 小巧 (compact), 舒服 (comfortable), 薄 (thin;light) 蓝牙 (blue-tooth), 强 (strong;strength), 容易 (easy)</p> <p>Neg 坏 (bad), 差 (poor), 死机 (crash), 慢 (slow), 没 (no;not), 难 (difficult;hard), 少 (less), 修 (repair)</p>
DigiCam	<p>Pos 简单 (simple), 防抖 (shake reduction), 优点 (advantage), 小巧 (compact), 时尚 (fashionable), 强 (strong;strength), 长焦 (telephoto), 动态 (dynamic), 全 (comprehensive), 专业 (professional)</p> <p>Neg 后悔 (regret), 坏 (bad), 差 (poor), 退货 (return;refund), 慢 (slow), 暗 (dark), 贵 (expensive), 难 (difficult;hard), 耗电 (consume much electricity), 塑料 (plastic), 修 (repair)</p>
MP3	<p>Pos 出色 (outstanding), 小巧 (compact), 齐全 (comprehensive) 简单 (simple), 强 (strong;strength), 美观 (beautiful), 质感 (textual), 金属 (metallic), 不错 (not bad;pretty good)</p> <p>Neg 杂音 (noise), 费电 (consume much electricity), 差 (poor), 坏 (bad), 短 (short), 贵 (expensive), 次 (substandard), 死机 (crash), 没 (no), 但是 (but)</p>
Monitors	<p>Pos 专业 (professional), 清晰 (in focus), 时尚 (fashionable), 简洁 (concise), 节能 (energy efficient), 纯平 (flat screen), 不错 (not bad;pretty good), 舒服 (comfortable), 显亮 (looks bright), 锐利 (sharp)</p> <p>Neg 变形 (deformation), 模糊 (blurred), 严重 (serious;severe), 失真 (distortion), 偏色 (color cast bad), 坏 (bad), 差 (poor), 漏光 (leakage of light), 黑屏 (black screen), 暗 (dark), 抖动 (jitter)</p>

## 6 Conclusions

This paper has proposed the latent sentiment model (LSM) for weakly-supervised cross-lingual sentiment classification. A mechanism has been introduced to incorporate prior information about polarity words from sentiment lexicons where preferences on expectations of sentiment labels of those lexicon words are expressed using generalized expectation criteria. Experimental results of sentiment classification on Chinese product reviews show that in the absence of a language-specific sentiment lexicon, the translated English lexicons can still produce satisfactory results with the sentiment classification accuracy of 80.34% being achieved averaging over four different types of product reviews. Compared to the existing approaches to cross-lingual sentiment classification which either rely on labeled corpora for classifier learning or iterative training for performance gains, the proposed approach is simple and readily to be used for online and real-time sentiment classification from the Web.

One issue relating to the proposed approach is that it still depends on the quality of machine translation and the performance of sentiment classification is thus affected by the language gap between the source and target language. A possible way to alleviate this problem is to construct a language-specific sentiment lexicon automatically from data and use it as the prior information source to be incorporated into LSM model learning.

## References

1. Banea, C., Mihalcea, R., Wiebe, J., Hassan, S.: Multilingual subjectivity analysis using machine translation. In: Proceedings of the EMNLP, pp. 127–135 (2008)
2. Bautin, M., Vijayarenu, L., Skiena, S.: International sentiment analysis for news and blogs. In: Proceedings of the ICWSM (2008)
3. Blei, D., McAuliffe, J.: Supervised topic models. *Advances in Neural Information Processing Systems* 20, 121–128 (2008)
4. Druck, G., Mann, G., McCallum, A.: Learning from labeled features using generalized expectation criteria. In: SIGIR, pp. 595–602 (2008)
5. Esuli, A., Sebastiani, F.: SentiWordNet: A publicly available lexical resource for opinion mining. In: Proceedings of LREC, vol. 6 (2006)
6. Lacoste-Julien, S., Sha, F., Jordan, M.: DiscLDA: Discriminative learning for dimensionality reduction and classification. In: NIPS (2008)
7. Lin, C., He, Y., Everson, R.: A Comparative Study of Bayesian Models for Unsupervised Sentiment Detection. In: CoNLL (2010)
8. Lin, C., He, Y.: Joint sentiment/topic model for sentiment analysis. In: CIKM (2009)
9. McCallum, A., Mann, G., Druck, G.: Generalized expectation criteria. Tech. Rep. 2007-60, University of Massachusetts Amherst (2007)
10. Mihalcea, R., Banea, C., Wiebe, J.: Learning multilingual subjective language via cross-lingual projections. In: Proceedings of the ACL, pp. 976–983 (2007)
11. Mimno, D., McCallum, A.: Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In: Proceedings of the UAI (2008)
12. Qiu, L., Zhang, W., Hu, C., Zhao, K.: Selc: a self-supervised model for sentiment classification. In: Proceeding of the CIKM, pp. 929–936 (2009)
13. Ramage, D., Hall, D., Nallapati, R., Manning, C.: Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In: EMNLP, pp. 248–256 (2009)
14. Schapire, R., Rochery, M., Rahim, M., Gupta, N.: Incorporating prior knowledge into boosting. In: ICML, pp. 538–545 (2002)
15. Tan, S., Wang, Y., Cheng, X.: Combining learn-based and lexicon-based techniques for sentiment detection without using labeled examples. In: Proceedings of the SIGIR, pp. 743–744 (2008)
16. Wan, X.: Using bilingual knowledge and ensemble techniques for unsupervised Chinese sentiment analysis. In: Proceedings of the EMNLP, pp. 553–561 (2008)
17. Wan, X.: Co-training for cross-lingual sentiment classification. In: Proceedings of the Joint Conference of the ACL and the AFNLP, pp. 235–243 (2009)
18. Zagibalov, T., Carroll, J.: Automatic seed word selection for unsupervised sentiment classification of Chinese text. In: Proceedings of the COLING, pp. 1073–1080 (2008)
19. Zagibalov, T., Carroll, J.: Unsupervised classification of sentiment and objectivity in chinese text. In: Proceedings of the IJCNLP, pp. 304–311 (2008)

# Fractional Similarity: Cross-Lingual Feature Selection for Search

Jagadeesh Jagarlamudi<sup>1</sup> and Paul N. Bennett<sup>2</sup>

<sup>1</sup> University of Maryland, Computer Science, College Park MD 20742, USA  
jags@umiacs.umd.edu

<sup>2</sup> Microsoft Research, One Microsoft Way, Redmond WA 98052, USA  
paul.n.bennett@microsoft.com

**Abstract.** Training data as well as supplementary data such as usage-based click behavior may abound in one search market (*i.e.*, a particular region, domain, or language) and be much scarcer in another market. Transfer methods attempt to improve performance in these resource-scarce markets by leveraging data across markets. However, differences in feature distributions across markets can change the optimal model. We introduce a method called Fractional Similarity, which uses query-based variance *within* a market to obtain more reliable estimates of feature deviations *across* markets. An empirical analysis demonstrates that using this scoring method as a feature selection criterion in cross-lingual transfer improves relevance ranking in the foreign language and compares favorably to a baseline based on KL divergence.

## 1 Introduction

Recent approaches [1,2,3,4] pose ranking search results as a machine learning problem by representing each query-document pair as a vector of features with an associated graded relevance label. The advantage of this approach is that a variety of heterogeneous features – such as traditional IR content-based features like BM25F [5], web graph based features like PageRank, and user behavioral data like aggregated query-click information [6] – can be fed into the system along with relevance judgments, and the ranker will learn an appropriate model. While ideally this learned model could be applied in any market (*i.e.*, a different region, domain, or language) that implements the input features, in practice there can be many challenges to *transferring* learned models.

For example, user behavior-based features, which have been shown to be very helpful in search accuracy [7,6,8,9], may vary in the amount of signal they carry across markets. This may occur when more behavioral data is available in one market than another – either because the number of users varies across markets or simply because of a difference in the amount of time a search engine has been available in those markets. Furthermore, the distributions of content-based features as well as their significance for relevance may vary across languages because of differences in parsing (*e.g.*, dealing with compound words, inflection, tokenization). In addition, the amount of labeled data available to use in model

transfer – a key ingredient in model performance – can vary across markets since acquiring the graded relevance labeled data can be costly. Thus, in order to facilitate the use of labeled data from different markets, we would like an automatic method to identify the commonalities and differences across the markets.

In this paper, we address the problem of using training data from one market (*e.g.*, English) to improve the search performance in a foreign language market (*e.g.*, German) by automatically identifying the ranker’s input features that deviate significantly across markets. Different flavors of this problem have been attempted by other researchers [10,11,12,13,14]. At a broader level, there are two possible approaches to transfer the useful information across languages depending on the availability of original queries in both the languages. When the queries are available we can identify a pair of queries (*e.g.*, “dresses” in English and “Kleider” in German) with the same intent and transfer the query level knowledge across markets [10]. Such an approach uses only aligned queries and discards many English queries which doesn’t have translation in German. In this paper, we devise a general approach that doesn’t rely on the availability of aligned queries and instead uses *only* the information available in the feature vector in all query-document pairs.

We take a machine learning approach and pose the knowledge transfer across languages as a feature selection problem. Specifically, we aim to identify features which have similar distribution across languages and hence their data from English can be used in training the foreign language ranker. We propose a technique called Fractional Similarity (Sec. 4.2) to identify a feature’s similarity across English and foreign language training data. This method addresses two shortcomings of statistical tests in the IR setting (Sec. 3). First, as has been noted elsewhere [?], variance in an observed statistic over an IR collection is primarily dependent on differences in the query set. This mismatch, which we refer to as query-set variance, is especially problematic in the transfer setting since the query sets from the two markets are generally different – often markedly so. Second, the document feature vectors for a specific query are often correlated (at least in some features), this query-block correlation can exacerbate the first problem by causing large shifts in a statistic for a feature based on whether the query is considered or not. Fractional Similarity addresses these shortcomings (Sec. 4.2) by adapting statistical tests to a query-centric sampling setting and thus enables a fair comparison between different features. In Sec. 5 we describe how to use Fractional Similarity in IR scenario to rank all the features based on their similarity value providing a way for feature selection. In our experiments (Sec. 6), we found that a simple approach of dropping the mismatched features from English data and using the rest of the training data helps in improving the search accuracy in the foreign language.

## 2 Related Work

Here we briefly describe the most relevant work before delving further into the problem. Gao *et al.* [10] uses English web search results to improve the ranking of non-ambiguous Chinese queries (referred to as Linguistically Non-local

queries). Other research [11,12] uses English as an assisting language to provide pseudo-relevant terms for queries in different languages. The generality of these approaches is limited either by the type of queries or in the setting (traditional TREC style) they are explored. In [13], English training data from a general domain has been used to improve the accuracy of the English queries from a Korean market. But in this particular case both in-domain and out-of-domain data are from English, hence the set of features used for the learning algorithm remain same. Here we study the problem in a general scenario by mapping it as a transfer learning problem. The main aim is to use the large amounts of training data available in English along with the foreign language data to improve the search performance in the foreign language.

In terms of tests to measure deviation of feature distributions, there have been some measures proposed in domain adaptation literature [16,17] to compute distance between the source and target distributions of a feature. But they are primarily intended to give theoretical bounds on the performance of the adapted classifier and are also not suitable to IR – both because they do not adequately handle query-set variance and query-block correlation and for the reasons mentioned in Sec. 3.

In addition to addressing both the IR specific issues, our method is more efficient because it only involves off-line computations, computing feature deviations across search markets and training the ranker with the additional English language data, and doesn't involve any computation during the actual querying.

### 3 Challenges in Computing Feature Similarity

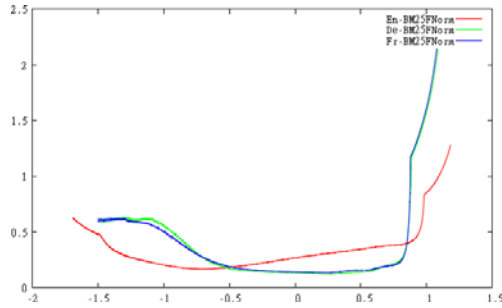
In this section we describe some of the challenges in computing the similarity of feature distributions across languages. While some of these challenges are specific to Information Retrieval the rest of them exist in other scenarios as well.

Note that in general, determining whether a feature distribution is similar across markets can be a challenging problem. For example, Figure 1 depicts the distributions of BM25F [5] in English, French, and German language training data (normalization was done to account for trivial differences).<sup>1</sup> From the figure, it is clear that the distribution of BM25F scores in French and German languages bear a closer resemblance than to that of English, but is this difference significant enough to negatively impact transfer learning or is it a minor difference that additional normalization might correct? Even if manual inspection was straightforward, a formal method of characterizing the deviation would be advantageous in many settings.

While statistical tests to measure divergence in distributions exist, they are ill-suited to this problem for two reasons stemming from query-based variance. The first of these effects on variance results from the query-block correlation that occurs among related documents. That is, in the IR learning to rank setting, we typically have a training/test instance for each query-document pair, and

---

<sup>1</sup> We refer to this type of probability density function (pdf) as the feature distribution of a feature in the rest of the paper.



**Fig. 1.** Feature distribution of BM25F in English, German, and French language data. The x-axis denotes the normalized feature value and the y-axis denotes the probability of the feature estimated using Kernel probability estimation technique.

for each query, a block of instances exist corresponding to differing documents relationship to the query. Because these documents are often related to the query in some way (*e.g.* top 100) that made them a candidate for final ranking, their features will often be correlated. For example, consider a slightly exaggerated case for a feature, “Number\_of\_query\_words\_found\_in\_document”. By the very nature of the documents being initial matches, all of them are likely to have similar values (probably near the query length). While for a different query, the same feature takes a different value but also highly correlated within the query. While the reader may think that a proper normalization (*e.g.*, normalization by query length) can alleviate this problem, we argue that this problem is more general both in this case and also may occur for other features such as those based on link analysis or behavioral data where there is no obvious normalization. The net effect is that even with in a language, statistics such as the mean or variance of a feature can often shift considerably based on the inclusion/exclusion of a small set of queries and their associated blocks.

A related challenge is that of query-set variance. That is, when comparing two different query sets, a feature can appear superficially different ultimately because the queries are different and not because how the feature characterizes the query-document relationship is different. Within a single language this can be a formidable challenge of its own and arises in large part because of query-block correlation, but across languages, this problem can be even worse since the query sets can differ even more (*e.g.* the most common queries, sites, *etc.* may differ across languages, cultures, regions).

As a result, a reasonable method should consider the impact of query-block correlation and query-set variance within the market whose data is to be transferred and use that as a reference to determine what deviations in the foreign language are actually significant. This is exactly what our method does. We now move on to the formal derivation of our approach.

## 4 Computing Feature Similarity

Our proposed method to compute the similarity of a feature’s distribution across languages builds on the well known T-test [18]. Thus, we start by briefly reviewing the T-test and then move on to the details of Fractional Similarity. Though we discuss it mainly in the context of T-test and continuous features, it can be extended to other relevant significance tests (*e.g.*,  $\chi^2$  test) and to discrete features as well.

### 4.1 T-test

Given two samples ( $X_1$  and  $X_2$ ) from an unknown underlying distributions, the T-test [18] can be used to verify if the means of both the samples are equal or not. Since the variance of a feature may differ across languages, we use the more general form that assumes a possibly different variance for the two random variables to be compared. Let  $\mu_1$  and  $\mu_2$  denote the means of both the samples and  $\sigma_1^2$  and  $\sigma_2^2$  denote the variances of both the samples, then the t-statistic and the degrees of freedom are given by:

$$t = \frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}} \quad \text{and} \quad d.f. = \frac{(\sigma_1^2/n_1 + \sigma_2^2/n_2)^2}{(\sigma_1^2/n_1)^2/(n_1 - 1) + (\sigma_2^2/n_2)^2/(n_2 - 1)}$$

where  $n_1$  and  $n_2$  are the respective sample sizes. Both these statistics along with the Students t-distribution [19] can be used to get the probability (referred to as the “p-value” function in our pseudo code) of observing the result under the null hypothesis that the means of the samples are equal. We would like to remind the reader that in statistical significance testing we are typically interested in showing that a new result is different from the baseline so we want the p-value to lower than the critical value. But in the case of transfer, we are interested in finding the similarity of two samples, so we want the means of both the samples to be same which means that we want p-value to be higher.

We will also need the result below, that the mean and variance of a convex combination of the above samples ( $X^\alpha = (1 - \alpha)X_1 + \alpha X_2$ ) with  $\alpha \in [0, 1]$  is given by:

$$\mu^\alpha = (1 - \alpha)\mu_1 + \alpha\mu_2 \quad \text{and} \quad \sigma_\alpha^2 = (1 - \alpha)(\sigma_1^2 + \mu_1^2) + \alpha(\sigma_2^2 + \mu_2^2) - (\mu^\alpha)^2 \quad (1)$$

We will use these expressions in Sec. 4.2 to compute the mean and variance of an arbitrary convex combination of two random samples.

### 4.2 Fractional Similarity

A direct application of the T-test to our problem would, for each feature, select random samples from both English and foreign language data sets and verify if the means of both the samples are equal or not. However, the impact of query-set variance across markets, typically yields a t-test value close to zero probability in almost all cases rendering it practically useless to rank the features. In reality,



the simple t-test indicates that the sets are composed of different queries (an obvious fact we know) and not that the relationship the feature characterizes between the documents and the queries is significantly different (what we are interested in knowing).

We use the following key idea to compute the similarity between two distributions ( $P$  and  $Q$ ). If both the given distributions are the same (*i.e.*  $P \equiv Q$ ), then, with high probability, any two random samples  $P_s$  (drawn from  $P$ ) and  $Q_s$  (drawn from  $Q$ ) are statistically indistinguishable among themselves and also from a convex combination  $((1 - \alpha)P_s + \alpha Q_s)$  of the samples. When the underlying distributions are indeed different ( $P \neq Q$ ), then for some value of  $\alpha$  the convex combination starts looking different from both the samples and we leverage on this intuition to compute the similarity. If we treat the convex combination as if we are replacing  $\alpha$  fraction<sup>2</sup> of examples from  $P_s$  with those of  $Q_s$ , then as we replace more and more instances the resulting sample starts looking different from the original sample  $P_s$ . We use this *fraction of examples* to be replaced to make it look different from  $P_s$  as indicative of the similarity between both the distributions (hence the name Fractional Similarity). The value of Fractional Similarity lies in the range 0 and 1, with a higher value indicating better similarity of the underlying distributions.

Let  $P_s^\alpha$  be the convex combination of the samples  $P_s$  and  $Q_s$ , *i.e.*,  $P_s^\alpha \leftarrow (1 - \alpha)P_s + \alpha Q_s$  and  $P_s^r$  be another random sample (superscript  $r$  stands for reference) drawn from  $P$ . Now the objective is to compare both  $P_s$  and the convex combination with the reference sample. If  $p_{pp}$  and  $p_{pq}^\alpha$  denote the p-values obtained by the T-test on pairs of samples  $(P_s, P_s^r)$  and  $(P_s^\alpha, P_s^r)$  respectively, then we propose to use the following statistic:

$$\text{frac}^\alpha = \frac{p_{pq}^\alpha}{p_{pp}} \tag{2}$$

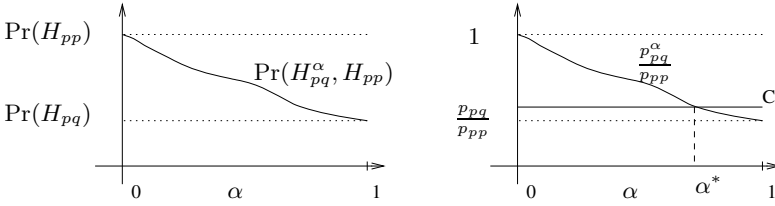
and we define Fractional Similarity as the maximum value of  $\alpha$  such that  $\text{frac}^\alpha$  is greater than a critical value ( $C$ ), *i.e.* Fractional Similarity =  $\arg \max_\alpha \text{frac}^\alpha > C$ . Our statistic explicitly takes within language variability into account, via the denominator of Eqn. <sup>2</sup> and as result it can be understood as a normalized version of across language variability.

Formally, the proposed statistic is inspired by the following observation. Let  $H_{pp}$  denote the event that the means of both the samples  $P_s$  and  $P_s^r$  are equal and also  $H_{pq}^\alpha$  denote the event that the means of  $P_s^\alpha$  and  $P_s^r$  are equal. Since we want to account for the variance of a feature in its original distribution, we assume the truth of  $H_{pp}$  event and find the probability of the event  $H_{pq}^\alpha$ . That is, we are interested in the following quantity:

$$\Pr(H_{pq}^\alpha | H_{pp}) = \frac{\Pr(H_{pp}, H_{pq}^\alpha)}{\Pr(H_{pp})} = \frac{\Pr(H_{pp} | H_{pq}^\alpha) \Pr(H_{pq}^\alpha)}{\Pr(H_{pp})} \tag{3}$$

---

<sup>2</sup> In our further notation, we use  $\alpha$  as a superscript when indicating an  $\alpha$  combined sample.



**Fig. 2.** When  $\alpha = 0$ ,  $P_s^\alpha$  becomes  $P_s$  as a result the joint probability (left image) becomes  $\Pr(H_{pp})$  and hence the  $\text{frac}^\alpha$  (right image) becomes 1. And as  $\alpha$  approaches 1, we introduce more and more noisy instances and eventually the joint probability reduces to  $\Pr(H_{pq})$  and  $\text{frac}^\alpha$  reaches its minimum value of  $\frac{p_{pq}}{p_{pp}}$ .

---

**Algorithm 1.** FractionalSimilarity( $P_s^r, P_s, Q_s, C$ )

---

```

1:  $p_{pp} \leftarrow \text{p-value}(P_s^r, P_s)$ . //With in language variability
2:  $\text{frac} \leftarrow \frac{1.0}{p_{pp}}$ 
3: if  $\text{frac} \leq C$  then
4:   return 0
5: end if
6:  $p_{pq} \leftarrow \text{p-value}(P_s^r, Q_s)$ . //Across language variance
7:  $\text{frac} \leftarrow \frac{p_{pq}}{p_{pp}}$ 
8: if  $\text{frac} > C$  then
9:   return 1
10: end if
11: Set  $\alpha \leftarrow 1$  //Prepare for binary search over  $\alpha$ 
12: Let  $P_s^\alpha \leftarrow (1 - \alpha)P_s + \alpha Q_s$  and  $\text{frac}^\alpha \leftarrow \frac{\text{p-value}(P_s^r, P_s^\alpha)}{p_{pp}}$ .
13: return  $\alpha^* = \arg \max_\alpha \text{ s.t. } \text{frac}^\alpha > C$  // Do a binary search over  $\alpha$ 

```

---

Now consider both the events in the numerator,  $H_{pq}^\alpha$  indicates the truth that  $P_s^r$  has the same mean as that of a  $\alpha$  noisy sample of  $P_s$  which automatically implies that it also has the same mean as the original, noiseless, sample  $P_s$  resulting in the truth of the event  $H_{pp}$ . That is to say  $\Pr(H_{pp} | H_{pq}^\alpha) = 1$ . Thus the conditional probability reduces to Eqn. 2.

The hypothetical behavior of the joint probability (numerator in Eqn. 3) and our statistic are shown in Fig. 2. As the value of  $\alpha$  increases, the conditional probability in Eqn. 3 reduces and Fractional Similarity is the value ( $\alpha^*$ ) at which the fraction falls below the critical value ( $C$ ). The pseudo code to compute this value is shown in Algorithm 1. The code between lines 1-10 checks if the samples are either too dissimilar or very similar to each other. While lines 11-13 suggest a binary search procedure to find the value of required  $\alpha^*$ . During the binary search procedure, for any given arbitrary  $\alpha$  we don't need to explicitly combine instances from  $P_s$  and  $Q_s$  to obtain  $P_s^\alpha$ . Instead, since the T-test requires only mean and variance, we use the analytically derived values (Eqn. 1) of the combined sample. This relaxation makes the search process both efficient and also more accurate as the corresponding curve becomes a non-increasing function.

---

**Algorithm 2.** FeatureSimilarity( $E, F$ )

---

**Input:** Feature values in English ( $E$ ) and the foreign language ( $F$ )**Output:** Similarity of this feature in both these distributions

```

1: for  $i = 1 \rightarrow n$  do
2:   Generate random samples  $E_s^r, E_s \sim E$  and  $F_s \sim F$ 
3:   Estimate probability density function (pdf) from  $E - \{E_s^r \cup E_s\}$ .
4:   Let  $L_e^r, L_e$  &  $L_f$  be the average log-likelihood of the queries in each sample
5:    $\alpha_i^* \leftarrow$  FractionalSimilarity( $L_e^r, L_e, L_f$ )
6: end for
7: return Average( $\alpha_1^*, \dots, \alpha_n^*$ ).

```

---

## 5 Cross-Lingual Feature Selection

This section describes how we use Fractional Similarity to identify features that have a similar distribution in both English and foreign language training data. Algorithm 2 gives the pseudo code.

Let there be a total of  $m$  queries in our English training data ( $E$ ). We first generate two random samples ( $E_s^r$  and  $E_s$ ) from  $E$  with 10% of queries in each sample (line 2 of the pseudo code) leaving 80% of queries for training a pdf. We then generate a third sample ( $F_s$ ) of approximately the same number of queries from foreign language training data ( $F$ ). If we choose to include a query in any sample then we include feature values corresponding to all the results of the query. At this point, an instance in any of the above three samples corresponds to the feature value of a query-document pair.

However, simply comparing the means of the feature values would not test if the entire distributions are similar. If we have a pdf of the feature, though, we can use the value of the likelihood of a sample under that pdf to transform the problem of verifying the equality of means to the problem of verifying if the underlying distributions are same. This holds because probability at any point is the area of an infinitesimal rectangle considered around that point. So by comparing the average log-likelihood values we compare the area under the trained English pdf at random points drawn from English and foreign language distributions. Thus the similarity between these log-likelihood values is an indicator of the similarity between the underlying distributions from which the samples are drawn. Another advantage of using the likelihood is that it makes the approach amenable to different types of features with appropriate choice of pdf estimate.

Since we assume that English has more training data, we estimate the probability density function (pdf) of this feature from the remaining 80% of English training data (line 3) using kernel density estimation [20]. Now we compute log-likelihood of the feature values in all the three samples and then compute the average of all the instances per query (line 4). Let the average log-likelihood values of all the three samples be stored in  $L_e^r, L_e$  and  $L_f$  respectively. Here, each log-likelihood value is a property of a query which is an aggregate measure, the joint probability, over all the results of this query. Next, we use Fractional Similarity described in Sec. 4.2 to compute similarity between the distributions

that generated the log-likelihood values (line 5). We repeat this process multiple times and return the average of all the similarity values (line 7) as the similarity of this feature between English and foreign language training data.

There are two main key insights in the way we apply Fractional Similarity to IR. The first one is that, our sampling criterion is based on queries and not on query-document pairs, this implies while computing Fractional Similarity we try to find the fraction of English queries that can be replaced with foreign language queries. Because we are sampling queries as a unit, this enables us to deal with query-specific variance more directly. Secondly, we also deal with query-set variance better because the denominator of Eqn. 2 includes normalization by an estimate of the within language query-set variance.

Though we have developed our approach in the context of Information Retrieval, it is applicable in any scenario in which correlation arises among groups of instances. Furthermore, with appropriate choice of pdf estimation technique, it also straightforwardly extends to different types of features and to multivariate setting.

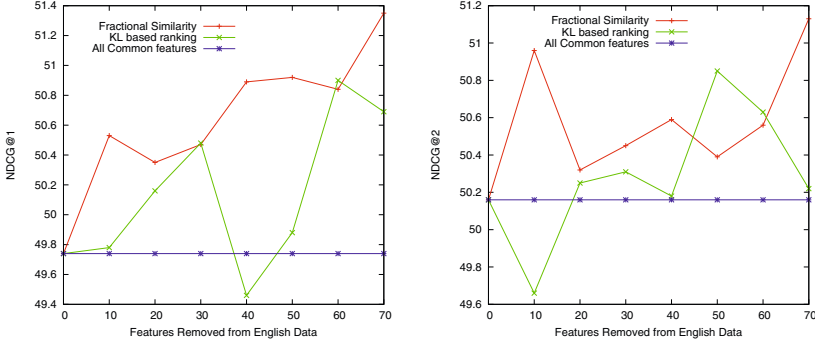
## 6 Experiments

We experimented with the task of improving a German language web search ranking model by using English training data. For both the languages, we took random query samples from the logs of a commercial search engine. Both these query sets differ in terms of the time spans and the markets they cover. English queries are from U.S. market while German queries are from German market. The English data set contains 15K queries with an average of 135 results per query while the German data set has 7K queries with an average of 55 results per query. The German data set has a total of 537 features, including some click-through features, of which 347 are common to the English data set. Our aim is to select features from these common features whose training data from English can also be used in training the German ranker. We only use a subset of features from English and always use all the 537 features in German data set. For each of those common features, we use Alg. 2 to compute its similarity across English and German data sets and then rank features according to their similarity and use them selectively. We also compute feature similarity using KL-divergence and use it as a baseline system to compare with Fractional Similarity. Note that KL-divergence based feature ranking doesn't take query-specific variance into account. We use a state-of-the-art learning algorithm, LambdaMART [3], to train the ranker.

In the first experiment, we compare different types of combination strategies to verify if English training data can improve the relevance ranking in German. On a smaller data set (of approximately half size), we try two adaptation strategies: the first one is to consider the data corresponding to all the common features from the English data set and simply add it to the German training data to learn the ranker ('Eng\_align+German' run). This strategy discards the fact that the new data is from a different language and treats it as if it were coming from German but with some missing feature values. We use model adaptation

**Table 1.** Performance with different combination strategies

	NDCG@1	NDCG@2	NDCG@10
German only	48.69	49.42	56.52
Eng_adapt+German	49.87	49.91	56.91
Eng_align+German	<b>50.77</b>	<b>50.63</b>	<b>57.47</b>



**Fig. 3.** Performance obtained by gradually removing noisy features from English data. X-axis denotes the number of features removed and Y-axis denotes the NDCG values.

[13] as another strategy. Here, we first learn a base ranker on the English data (out-of-domain) with only the common features and then adapt it to the German training data (in-domain). We also report the results of a baseline ranker learned only on the German training data. The NDCG scores [21] for these different runs are shown in Table 1. Though we also report NDCG@10, we are more interested in the performance among the first few positions where improved results often impact the user experience. The results show that both strategies improved upon the German only baseline – indicating the benefit of using English training data in training the German ranker. Also, we observe that simply selecting the common features from the English data and appending it to the German data (‘\_align’ strategy) showed considerable improvements compared to the adapted version. So in our further experiments we report results only using the ‘\_align’ strategy.

The previous experiment showed that using English training data improves the German ranker, but it is not clear if all the common features are needed or only a subset of them are really useful. To explore this further, we first identify the mismatched features using Fractional Similarity and then repeat the experiment multiple times while gradually removing the mismatched features. We also use KL-divergence to identify the mismatched features and compare it with the ranking obtained by Fractional Similarity. We divide the corpus into two random splits and then subdivide each split into 80%, 10% and 10% for training, validation and test sets respectively. The results reported in Fig. 3 are averaged over both the test sets. Each time we remove the least similar ten features

returned by the two feature ranking schemes and retrain two separate rankers. Here, the baseline system uses all the common features from the English data. Note that the baseline number is slightly different from the Eng\_align+German number in Table 1 as the latter was only over a subset of the data. The red line in Fig. 3 indicates removing the mismatched features found by Fractional Similarity while the green line uses feature rankings obtained by KL-divergence. Though removing features based on KL-divergence also improves the performance compared to the baseline system, its performance fell short sometimes while our method always performed better than the baseline system. Except at one point, our method always performed better than KL-divergence feature selection. Removing approximately 25% of the mismatched features gave an improvement of 1.6 NDCG points at rank 1 (left image of Fig. 3) which is on the order of notable improvements in competitive ranking settings reported by other researchers [23]. From the graphs, it is very clear that our method identifies the noisy features better than the KL-divergence based method. We observed this consistently at higher ranks, but as we go down (rank  $\geq 5$ ), feature selection using either method has, on average, either a neutral or negative effect. We believe this is because as we go deeper, the task becomes more akin to separating the relevant from the non-relevant rather than identifying the most relevant. While many of these feature distributions differ, they still provide a useful signal for general relevance, and thus transferring all of the data may prove superior when identifying general relevance is the goal.

Although feature selection can improve performance, naturally we expect that being too aggressive hurts performance. In fact, we observed that if we use only the 100 most similar features then the performance drops below the baseline ('Eng\_align+German'). In our experiments, we found that transferring 70-75% of the most similar features yielded the greatest improvements.

## 7 Discussion

Because of the high variance of a feature within a language and the challenges of comparing samples from two different query sets, traditional statistical significance tests output negligible probabilities – failing to correctly discriminate between feature distributions that truly differ and those that do not across languages. Fractional Similarity overcomes this problem by explicitly accounting for the within language variance of a feature by sampling in query blocks and normalizing by a within language factor. This increases the robustness of our approach and enables a fair comparison of the similarity scores between different features. Furthermore, empirical results demonstrate notable wins using a state-of-the-art ranker in a realistic setting.

## References

1. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: ICML 2005, pp. 89–96. ACM, New York (2005)

2. Burges, C.J.C., Ragno, R., Le, Q.V.: Learning to rank with nonsmooth cost functions. In: NIPS, pp. 193–200. MIT Press, Cambridge (2006)
3. Wu, Q., Burges, C.J., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. *Information Retrieval* 13(3), 254–270 (2010)
4. Gao, J., Qi, H., Xia, X., Yun Nie, J.: Linear discriminant model for information retrieval. In: Proceedings of the 28th International ACM SIGIR Conference, pp. 290–297. ACM Press, New York (2005)
5. Robertson, S., Zaragoza, H., Taylor, M.: Simple BM25 extension to multiple weighted fields. In: CIKM 2004: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, pp. 42–49. ACM, New York (2004)
6. Agichtein, E., Brill, E., Dumais, S.: Improving web search ranking by incorporating user behavior information. In: SIGIR 2006, pp. 19–26. ACM, New York (2006)
7. Broder, A.: A taxonomy of web search. *SIGIR Forum* 36(2), 3–10 (2002)
8. Rose, D.E., Levinson, D.: Understanding user goals in web search. In: WWW 2004: Proceedings of the 13th International Conference on World Wide Web, pp. 13–19. ACM, New York (2004)
9. Xue, G.R., Zeng, H.J., Chen, Z., Yu, Y., Ma, W.Y., Xi, W., Fan, W.: Optimizing web search using web click-through data. In: CIKM 2004: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, pp. 118–126. ACM, New York (2004)
10. Gao, W., Blitzer, J., Zhou, M.: Using english information in non-english web search. In: iNEWS 2008: Proceeding of the 2nd ACM Workshop on Improving Non English Web Searching, pp. 17–24. ACM, New York (2008)
11. Chinnakotla, M.K., Raman, K., Bhattacharyya, P.: Multilingual pseudo-relevance feedback: performance study of assisting languages. In: ACL 2010: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Morristown, NJ, USA, pp. 1346–1356 (2010)
12. Chinnakotla, M.K., Raman, K., Bhattacharyya, P.: Multilingual PRF: english lends a helping hand. In: SIGIR 2010, pp. 659–666. ACM, New York (2010)
13. Gao, J., Wu, Q., Burges, C., Svore, K., Su, Y., Khan, N., Shah, S., Zhou, H.: Model adaptation via model interpolation and boosting for web search ranking. In: EMNLP 2009: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Morristown, NJ, USA, pp. 505–513 (2009)
14. Bai, J., Zhou, K., Xue, G., Zha, H., Sun, G., Tseng, B., Zheng, Z., Chang, Y.: Multi-task learning for learning to rank in web search. In: CIKM 2009, pp. 1549–1552. ACM, New York (2009)
15. Carterette, B., Pavlu, V., Kanoulas, E., Aslam, J.A., Allan, J.: Evaluation over thousands of queries. In: SIGIR 2008, pp. 651–658. ACM, New York (2008)
16. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: Learning bounds and algorithms. CoRR abs/0902.3430 (2009)
17. Ben-david, S., Blitzer, J., Crammer, K., Sokolova, P.M.: Analysis of representations for domain adaptation. In: NIPS. MIT Press, Cambridge (2007)
18. Welch, B.L.: The generalization of ‘student’s’ problem when several different population variances are involved. *Biometrika* 34(1/2), 28–35 (1947)
19. Fisher Ronald, A.: Applications of “student’s” distribution. *Metron* 5, 90–104 (1925)
20. Parzen, E.: On estimation of a probability density function and mode. *The Annals of Mathematical Statistics* 33(3), 1065–1076 (1962)
21. Järvelin, K., Kekäläinen, J.: IR evaluation methods for retrieving highly relevant documents. In: SIGIR 2000, pp. 41–48. ACM, New York (2000)

# Is a Query Worth Translating: Ask the Users!

Ahmed Hefny, Kareem Darwish, and Ali Alkakhky

Microsoft Innovation Laboratory in Cairo  
{t-ahmehe, kareemd, t-aleka}@microsoft.com

**Abstract.** Users in many regions of the world are multilingual and they issue similar queries in different languages. Given a source language query, we propose query picking which involves finding equivalent target language queries in a large query log. Query picking treats translation as a search problem, and can serve as a translation method in the context of cross-language and multilingual search. Further, given that users usually issue queries when they think they can find relevant content, the success of query picking can serve as a strong indicator to the projected success of cross-language and multilingual search. In this paper we describe a system that performs query picking and we show that picked queries yield results that are statistically indistinguishable from a monolingual baseline. Further, using query picking to predict the effectiveness of cross-language results can have statistically significant effect on the success of multilingual search with improvements over a monolingual baseline. Multilingual merging methods that do not account for the success of query picking can often hurt retrieval effectiveness.

**Keywords:** cross-language search, multilingual search, query translation mining.

## 1 Introduction

Users in many countries and regions are multilingual<sup>1</sup>. In many regions, the second language is typically English or French and to a lesser extent German or Spanish. The interesting fact here is that the second language is often one with a larger web presence compared to local languages such as Filipino and Hindi.

Users in multilingual regions offer interesting opportunities for search applications, particularly cross-language and multilingual web search. One such opportunity is that users in a single region share common interests, and hence search for similar things in different languages. For example, in the Arab market, which includes all countries where most users identify Arabic as their primary language, 63% of queries are Arabic, 27% are in English, and the rest are mixed language queries. A casual inspection of a large query log from the Arab market confirms that many queries, even many tail queries, have equivalent Arabic and English versions in the logs. For example, one of the Arabic queries that we encountered in the log was “تحويل رصيد لحساب سوا” which had an equivalent English query “Sawabalance transfer” in the log and the English version yielded more relevant results than the Arabic version. This has interesting

---

<sup>1</sup>[http://en.wikipedia.org/wiki/List\\_of\\_multilingual\\_countries\\_and\\_regions](http://en.wikipedia.org/wiki/List_of_multilingual_countries_and_regions)



potential in cross-language and multilingual search: if a user issues a query in one language and we can ascertain that an equivalent exists in another language, then this can aid cross-language search and predict its effectiveness. The motivation for query picking is based on two assumptions, namely:

1. Given a very large query log in the target language and a source language query with relevant results in the target language, it is very likely that some users issued an equivalent query in the target language;
2. Users likely issue queries if they believe that they will find relevant results. Thus, if we can find an equivalent query in the target language query log, then users have in effect informed us that the cross-language results will likely have good results.

We refer to the process of finding equivalent queries in a target language query log given a source language query as query picking. In performing query picking, there are two main challenges, namely:

1) How to determine if a query in one language has an equivalent in another language. This typically involves some type of query translation mining. To tackle this problem, we explore the use of dictionary based translation, machine translation, and transliteration mining.

2) How to ascertain if a query would have good cross-language results. We address this problem in the context of query picking. Conceptually, we use the users' issuance of equivalent queries in the target language as a vote on the likelihood that cross-language results would produce reasonable results.

In this paper, we describe query picking and its effect on determining the success of cross-language and multilingual search. We show the effectiveness of query picking in a prototypical market, namely the Arab market, where bilingualism is prevalent and we have access to a large query log from a popular web search engine. Specifically, we examine the scenario where a bilingual user issues a query in Arabic and is searching for English document or where a user can benefit from seeing both Arabic and English results. This is motivated by the fact that the size of the English web is estimated to be at least 2 orders of magnitude larger than the Arabic web. Thus, significantly more information exists in English than in Arabic.

Our contributions in this paper are:

1. Applying a discriminative method for query picking to translate queries.
2. Predicting when cross-language and multilingual search would be effective.
3. Evaluating query picking and existing query translation techniques for cross-language web search. Most previously reported cross-language experiments in the literature were conducted on ad hoc collections.
4. Applying merging models with the aid of query picking to obtain effective multilingual web search. Though multilingual merge models are mentioned in the literature, our work applies multiple merge models that are informed by query picking. Multilingual search in this work involves Arabic and English.

The rest of the paper is organized as follows: Section 2 describes related work; Section 3 describes the query translation methods and query picking; Section 4 evaluates the

effectiveness of different translation schemes, their impact on cross-language search, and the power of query picking in predicting the success of cross-language search; Section 5 describes different results merging methodologies for multilingual search; Section 6 reports on the benefits of using query picking in results merging; and Section 7 concludes the paper.

## 2 Related Work

One of the central issues in this work pertains to query translation. Query translation has been explored extensively in the context of cross-language information retrieval, where a query is supplied in a source language to retrieve results in a target language. Two of the most popular query translation approaches are Dictionary Based Translation (DBT) methods [12] and Machine Translation (MT) [20]. DBT methods usually involve replacing each of the source language words with equivalent target language word(s). Since a source language word may have multiple translations, optionally the most popular translation or  $n$ best translations are used. Since web search engines typically use an AND operator by default, using multiple translations may cause translated queries to return no results. Another alternative is to use a synonym operator, which has been shown to be effective in CLIR [12][14]. A synonym operator can be approximated in web search by using an OR operator between different translations. The use of a weighted synonym operator, where each translation is assigned a confidence score, is not supported in popular web search engines, though it has been shown to be effective in cross-language search [19]. MT has been widely used for query translation [17][20]. Wu et al. [20] claim that MT outperforms DBT. Their claim is sensible in the context of web search for two reasons: a) MT attempts to optimally reorder words after translation and web search engines are typically sensitive to word order; and b) MT produces a single translation without any synonym or OR operators, for which the rankers of web search engines are not tuned. In our work, we experiment with DBT and MT.

Another approach of interest here is the so-called cross-lingual query suggestion [2][4]. This approach involves finding related translations for a source language query in a large web query log in the target language. Gao et al. [2] proposed a cross-language query suggestion<sup>2</sup> framework that used a discriminative model trained on cross-language query similarity, cross-language word co-occurrence in snippets of search results, co-clicks from both queries to the same URL, and monolingual query suggestion. Our work extends the work of Gao et al. [2] by using alternative features, such as phonetic similarity, relative query lengths, and cross-language coverage. Further, Gao et al. [2] restricted their experimentation to a set of manually selected queries for which cross-language equivalents are known to exist. In our work, we selected queries randomly without this restriction, making the problem more realistic.

The second issue involves merging multilingual results. For results merging in general, there are several simple techniques in the literature such as score-based

---

<sup>2</sup> Although cross-language query suggestion and query picking are similar as they both mine queries in a target language query log, we opted to use the term query picking to emphasize that we mine the target language log for a single query translation of the input query, not multiple translations of related queries.

merging, round-robin merging, and normalized score based merging [11]. Score-based merging assumes that scores in different ranked lists are comparable, which cannot be guaranteed. This is solved by normalizing scores from different ranked lists. Round robin merging assumes that different ranked lists have a comparable number of relevant documents [11]. Si and Callan[15] used a logistic regression based model to combine results from different multilingual ranked lists in the context of ad hoc search. Tsai et al. [16]also addressed the problem in the context of ad hoc search. They used document, query, and translation based word-level features to train an  $F^{\text{Rank}}$ -based ranker whose score was then linearly combined with the BM25 score of each document. Rankers of web search engine typically consider many more features such link-graph, document structure, and log based features. Gao et al. [3]used a Boltzman machine to learn a merging model that takes cross-language relations between retrieved documents into account. They tested their approach on ad hoc as well as web search. In the context of web search, they evaluated their approach using queries that have equivalents in the query log of the target language, which means that these queries would likely benefit from crosslanguage results merging. We apply and evaluate our work on an unbiased sample from the query log of a popular search engine. We thus handle the practical case where some queries would benefit from cross-language results merging while others would be hurt by suchmerging. This has to be taken into account when designing a merging model.

The third central issue is ascertaining when cross language web search would yield good results. The literature is relatively sparse on this issue. Kishida[6] examined “ease of search” and translation quality as means to predict cross language query prediction. Another less related work examined when and when not to translate query words is[9]. To the best of our knowledge, the proposed use ofquery logs, through query picking, to determine if cross language search would be effective is novel.

### 3 Query Translation

For our work, we used three different query translation methods, namely MT, DBT, and query picking.

**Machine Translation:** MT attempts to translate a string from one language to another [7]. References for this approach are too many to list here. The advantage of this technique is that an MT system can translate the vast majority of queries regardless of their frequency, and it attempts to find the most likely word order in translations, which could aid web search engines that take query word order as a feature. However, machine translation engine is tuned for natural language sentences and not queries. Further, MT provides a single translation, which may be incorrect particularly for short sequences that don’t have enough contexts.

We used an in-house statistical phrasal MT engine that is similar to Moses [7] for all our experiments. The engine was trained on a set of 14 million parallel Arabic-English sentence pairs. The alignment was performed using a Bayesian learner that was trained on word dependent transition models for HMM based word alignment [5]. Alignment produced a mapping of source word sequence to a target word sequence along with the probability of source given target and target given source. The parallel training data included United Nations records, parallel newswire stories, automatically mined web data, and parallel Wikipedia titles. We performed basic tokenization and

preprocessing of Arabic. The preprocessing included letter normalization [1] and the splitting of attached coordinating conjunction and preposition as in Lee et al. [10]. The MT system also transliterated out of vocabulary words using a statistical noisy channel model transliterator that was trained using 27,000 transliteration pairs and large English word and character language models.

**Dictionary Based Translation:** For DBT, we exploited the phrasal translation table that was produced by training the aforementioned MT engine. Basically, we attempted to find entries in the phrasal translation table that matched the longest word sub-sequences in the query and then combined the different translations with the original sub-string using a synonym operator [14]. The longest matching substring and multiword translations were treated as phrases. For example, the query “وادي الملوك” produced the query synonyms: (“وادي الملوك”, “valley of kings”, “valley of the kings”). In doing so, the query was multilingual, which would lead to finding results in both languages simultaneously. However, English pages generally had higher static rank than Arabic pages and consequently dominated search results. When no translations were found, we employed a transliteration miner that attempted to match out of vocabulary words to phonetically similar English words, which were extracted from the English queries in the log. The transliteration miner is similar to the baseline system in [1]. Briefly, the transliteration miner was trained on a set of 27,000 parallel transliteration pairs. The miner used character segment mappings to generate all possible transliteration in the target language while constraining generation to the existing words in the target language word list. The synonym operator was approximated using an OR operator between the different translations. Multiword translations were put between quotes to force exact matching. The web search engine that we used does not support the weighted synonym operator [19].

**Query Picking:** Instead of translating a query directly, query picking attempts to identify a target language query from a query log that matches the original query. To test query picking and to verify our assumption, we used a query log from a popular search engine. The query log was composed of all queries that were issued against the search engine from Arabic speaking countries between May 2009 and February 2010, inclusive. In total, there were slightly more than 186 million unique queries, of which 63% were Arabic, 27% were English, and the rest were mixed language queries. We examined a large sample of queries, and our estimates indicate that more than 50% of the Arabic queries have English equivalents in the query log.

Query picking was performed as follows: Given a source language query  $Q_S$ :

1. All queries from the log in the target language (English) were indexed by an IR system, namely Indri, which uses inference networks and supports the weighted synonym operator [13].
2. For  $Q_S$ , 3 queries were issued against the index, namely: 1) the machine translated version of the query; 2) a weighted synonym operator based query generated using the phrase table; and 3) a weighted synonym operator based query generated using the phrase table and transliterations that were mined in the aforementioned manner. The weighted synonym operator was used to include the confidence for each translation and transliteration equivalent. The queries 2 and 3 were constructed as in the DBT method without the inclusion of the source language strings.

3. A set  $Q_T$  of candidate translations was composed of the top 10 retrieved queries from the index for each translated query. Candidate translations were then classified using a Support Vector Machine (SVM) classifier based on the following features:

For each candidate translation  $Q_{T_i}$ :

- The relative difference in length between  $Q_S$  and  $Q_{T_i}$
- The percentage of words in  $Q_S$  that have translations in  $Q_{T_i}$
- The percentage of words in  $Q_{T_i}$  that have translations in  $Q_S$
- The mutual agreement score, which was calculated as follows:

$$Score(Q_{T_i}) = \frac{1}{|Q_{T_i}|} \sum_{w \in Q_{T_i}} \frac{f(w) - 1}{\max_{w' \in W(Q_T)} f(w') - 1}$$

Where  $|Q_{T_i}|$  is the number of unique words in that candidate,  $f(w)$  is the number of retrieved candidates that contain a word  $w$  and  $W(Q_T)$  is the set of all words in all candidates. In the case where  $\max_w f(w) = 1$ , the score is set to 0. This feature is based on the hypothesis that translated words that occur frequently in retrieved candidates are more likely to be correct translations of query terms.

For classification, we used the TinySVM<sup>3</sup> implementation of the SVM classifier with a linear kernel. The SVM classifier attempts to find the maximum margin separating hyperplane  $w^T x - b$ , where  $w$  is the feature weight vector,  $x$  is the feature vector, and  $b$  is the y-intercept. We considered a query to be a potential translation if  $w^T x - b > 0$ . If none of the queries were classified as potential translations, then no query was picked. If multiple queries were classified as potential translations, then the query that maximizes  $w^T x - b$  was chosen. The SVM was trained using a set of 70 queries and their corresponding 30 candidate translations (minus duplicates) for each query, which were manually judged.

On a separate 200 query validation set, our algorithm picked equivalent queries correctly with precision of 68% and recall of 58%. In computing precision and recall, proposed queries had to match initial query intent. For example, if query picking proposed “buying Sawacredit” and the intent was “Sawabalance transfer”, then the picked query was considered incorrect. When we relaxed the criterion to accept related queries (as in the example), precision climbed to 85%. The existence of related queries may indicate the presence of relevant results in the target language.

For the remainder of the paper, the “success” of query picking only indicates that query picking produced a candidate, without regard to whether the suggested candidate was correct or not.

## 4 Query Picking vs. MT and DBT in CLIR

This section presents the effectiveness of different translation techniques in cross language search and compares the results to source language query results. We conducted the experiments on a set of 200 randomly picked Arabic queries from the query log. Four variants for each query were issued: the original query and using the three aforementioned query translation techniques. The top 10 results for each of the

<sup>3</sup> <http://chasen.org/~taku/software/TinySVM/>

query variants were manually evaluated by independent judges and were assigned one of five relevance levels, namely: Perfect (4), Excellent (3), Good (2), Poor (1), and Bad (0). The judges only had access to the original queries and could not see the translations used to obtain the crosslanguage results. The judges were instructed to be languageneutral in their judgments. To avoid biases, the results were shuffled and translation methods were obfuscated. We used Normalized Discounted Cumulative

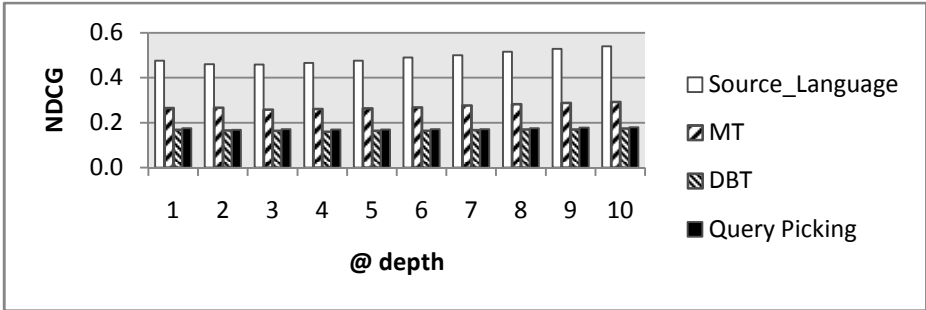


Fig. 1. Comparing the effectiveness of different translation methods

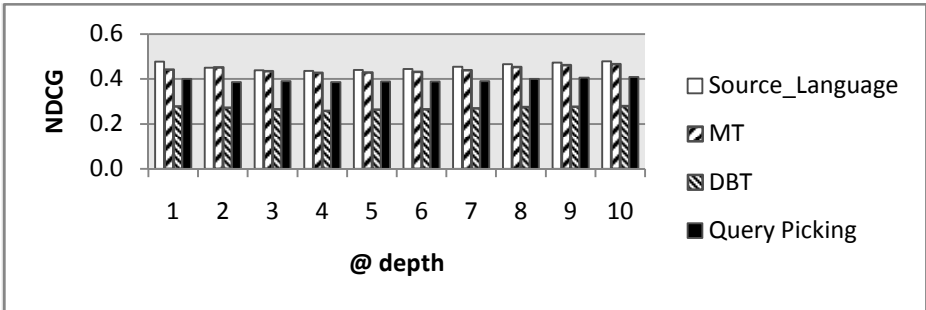


Fig. 2. Comparing the effectiveness of different translation methods when query picking succeeded (for 43% of the queries)

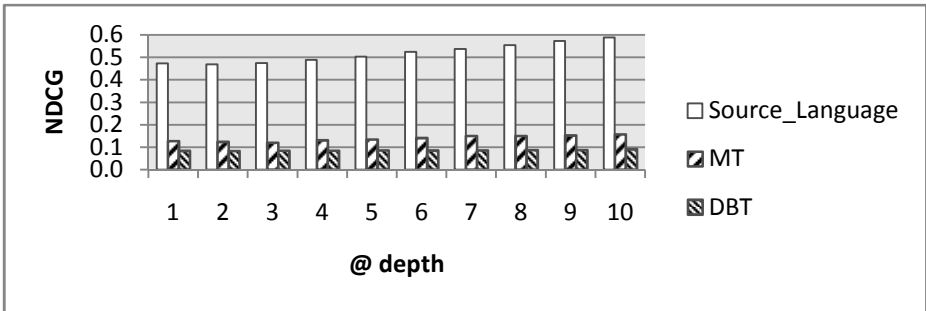


Fig. 3. Comparing the effectiveness of different translation methods when query picking did NOT succeed (for 57% of the queries)

Gain (NDCG) as the measure of goodness as described in [18]. Monolingual results served as a baseline. Figure 1 reports NDCG at depths ranging between 1 and 10. We restricted the depth to the top 10 results because users rarely see results beyond the first results page, which typically contains 10 results. Since query picking was successful for 43% of the queries (i.e. 57% of the queries had an NDCG of 0), Figure 2 and 3 compare the results when query picking was successful or not respectively.

As can be seen, monolingual results were generally better than cross language results. However, when the results were restricted to queries where query picking was successful, the differences in search effectiveness between the monolingual baseline and MT or query picking results were relatively small and statistically insignificant. Statistical significance in this paper was determined using a paired two-tailed t-test with  $p$ -value less than 0.05. When query picking was not successful, MT and DBT yielded very poor results. This seems to confirm our hypothesis that the success of query picking is a strong indicative feature in predicting the success of cross language search. When query picking was successful, the results indicate that cross language results were similar in quality to monolingual results. DBT queries fared poorly, which could be attributed to: web search engine ranker not being able to effectively rank results from complex queries; and the loss of proper word order.

## 5 Merging Models for Multilingual Search

The goal of the merging models is to merge the ranked results lists of multiple sources into a single ranked list that optimizes the overall retrieval effectiveness. In our bilingual setting, the source language query as well as each of the translated versions (MT, DBT, and picked) were issued independently against a popular web search engine resulting in four ranked lists to be merged. We used two methods for results merging, namely: Round Robin (RR) merging and a supervised ranking based on SVM<sup>Rank</sup>[8]. Monolingual results served as a baseline.

**Round Robin Merging:** We implemented the RR merging described in [11]. RR merging works by selecting a result from each ranked list in turn. We implemented 3 versions of RR merging based on combining monolingual results with MT, query picking, and DBT results independently. Monolingual results were always preferred over cross language results i.e. always included first in the merged list.

**Supervised Learning Based Merging:** We employed a supervised learning rank model, namely SVM<sup>Rank</sup>, to merge multiple ranked lists into a single list. SVM<sup>Rank</sup> was trained using the relevance judgments for query-result pairs which were used to extract pairwise order constraints. SVM<sup>Rank</sup> learned a linear combination of features that agrees with these constraints. We used 8 features: 4 boolean features indicating the presence of the URL in each of the four ranked lists and 4 real-valued features representing the score of a result in each list. The score was set to  $1/r$ , where  $r$  was the rank of the URL in the ranked list. If a result was not returned in a ranked list, it received a score of 0. Without overlap between ranked lists, SVM<sup>Rank</sup> would have produced a static merging scheme. However, Results routinely overlapped.

**Using Query Picking to Guide Merging:** To the best of our knowledge, most if not all reported work on multilingual results merging relied on a single merging model,

whether statistically trained or based on heuristics[16]. We have shown in Section 4 that queries where query picking succeeded yielded cross language results that were comparable to monolingual results. For the rest of the queries, cross language results were poor. So unlike previous work, we experimented with two merging models: one for queries for which query picking was successful and another when query picking was not. We applied this to RR and SVM<sup>Rank</sup> merging models. For RR:

*If query picking was successful*  
Then  
Merged List = RR Merge  
  
Else

Merged List = Original Query Results

For SVM<sup>Rank</sup> we trained two SVMs, one for queries for which picking was successful and another for queries for which picking was not successful.

## 6 Evaluation of Query Picking in Multilingual Search

We tested the aforementioned merging methods using the same 200 queries that were used for testing the effectiveness of translation methods in Section 4 along with their relevance judgments. A separate training set composed of 50 randomly selected queries was used for training SVM<sup>Rank</sup>. Specifically, one model was trained on all 50 queries, another was trained on queries for which picking was successful, and a third one was trained on queries for which picking was not successful. The percentages of queries where query picking was successful in the training and test sets were different by only a few percentage points. The results of training and test queries were obtained using the four previously described sources: monolingual search, MT, DBT, and query picking. We compared the results of the aforementioned merging methods to monolingual search results and we evaluated the effect of incorporating query picking in the merging model. Table 1 summarizes the ranked lists that we produced.

**Table 1.** Different merging models used for evaluation

Model	Description
Source_Language	Monolingual search results only
RR_{MT,DBT, Query_Pick}	RR merging used to merge between monolingual results and MT, DBT, and Query Picking results independently
RR_MT_Dual	RR merging used to merge between monolingual and MT result only if query picking is successful, and Source_Language otherwise
SVM <sup>Rank</sup> _Single	SVM <sup>Rank</sup> trained on all training queries
SVM <sup>Rank</sup> _Pick	SVM <sup>Rank</sup> trained on training queries when picking was successful
SVM <sup>Rank</sup> _No_Pick	SVM <sup>Rank</sup> trained on training queries when picking was not successful
SVM <sup>Rank</sup> _Dual	SVMRank_Pick used when picking was successful and SVMRank_No_Pick used when picking was not successful

Figure 4 reports NDCG for results depths ranging from 1 to 10 for all queries. When using RR merging, NDCG@1 did not change from the monolingual case because we always preferred monolingual results over cross language results. RR merging with MT actually hurt overall NDCG with up to 2.7 basis points (for NDCG@2). RR\_Query\_Pick slightly improved overall NDCG with improvements ranging from 0.6 to 1.8 basis points. The improvements realized by RR\_Query\_Pick were



statistically significant over the monolingual baseline for  $NDCG@{4-7,9,10}$ . In effect,  $RR\_Query\_Pick$  was guided by query picking, since a non-picked query will not have cross-language results.  $RR\_DBT$  consistently produced poor results. As for  $SVM^{Rank\_Single}$  it consistently produced slightly better results than the monolingual baseline, but all the differences were not statistically significant.

To demonstrate the correlation between success of query picking and effective merging, Figures 5 and 6 report on  $NDCG$  when query picking was successful or not respectively. Beyond  $NDCG@1$ , when query picking was successful,  $RR$  merging in  $RR\_MT$  and  $RR\_Query\_Pick$  consistently improved overall  $NDCG$ . The improvements ranged from 2.5 to 5.6 basis points and from 1.4 to 4.2 basis points for  $RR\_MT$  and  $RR\_Query\_Pick$  respectively. When query picking was not successful,  $RR$  merging consistently hurt overall  $NDCG$ . This was not surprising given that cross language results were generally poor when query picking was not successful. As for  $SVM^{Rank\_Pick}$  based merging, it consistently improved  $NDCG$  by 5.6 to 7.5 basis points. For  $SVM^{Rank\_No\_Pick}$ ,  $NDCG$  changed marginally by values less than  $\pm 0.4$  basis points. Except for  $NDCG@1$ , all  $SVM^{Rank\_Pick}$  improvements over source language,  $RR\_MT$ , and  $RR\_DBT$  were statistically significant.

Finally, Figure 7 reports results where query picking was used to guide merging strategies.  $SVM^{Rank\_Dual}$  used  $SVM^{Rank\_Pick}$  and  $SVM^{Rank\_No\_Pick}$  collaboratively.  $SVM^{Rank\_Single}$ , which is the same as in Figure 4, is also shown for comparison. For  $RR\_MT\_Dual$  in this figure,  $RR$  merging was used when query picking was successful, and monolingual results were used as when query picking was not successful. The results of  $RR\_MT\_Dual$  slightly edged those for  $RR\_Query\_Pick$ , but the differences were very small and not statistically significant.  $SVM^{Rank\_Dual}$  produced statistically significantly better results than  $SVM^{Rank\_Single}$  for  $NDCG@{3-10}$ . Further,  $SVM^{Rank\_Dual}$  was statistically significantly better than  $RR\_Query\_Pick$  for  $NDCG@{3,4,7-10}$ , but statistically better than  $RR\_MT\_Dual$  only for  $NDCG@3$ .  $SVM^{Rank\_Dual}$  produced results that were better than monolingual results by 2.7 to 3.4 basis points.

All the results suggest the success of query picking as a strong indication on the quality of cross language results. In examining  $SVM^{Rank}$  feature weights, we found that:  $SVM^{Rank\_Dual}$  preferred monolingual results, and cross language result were ranked higher than monolingual result only if they were returned by multiple sources. As expected, the weights assigned to cross language sources were significantly lower in the case of non-picked queries than in the case of picked queries.

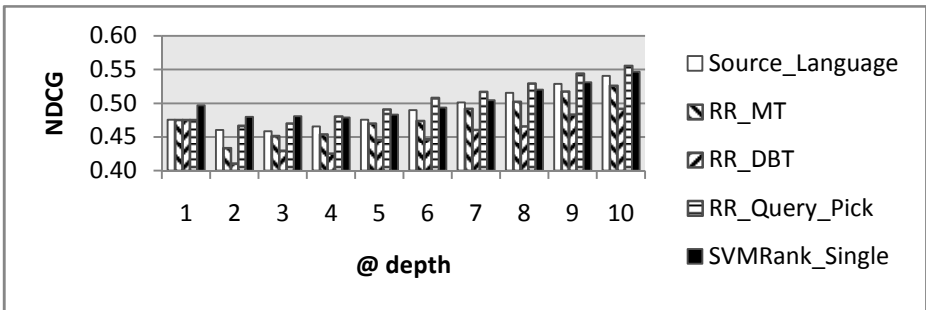


Fig. 4. Comparing monolingual results to merged results

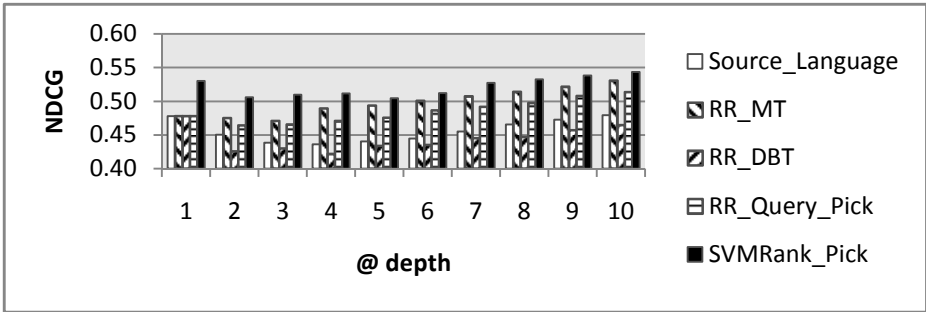


Fig. 5. Comparing monolingual results to merged results when query picking succeeded

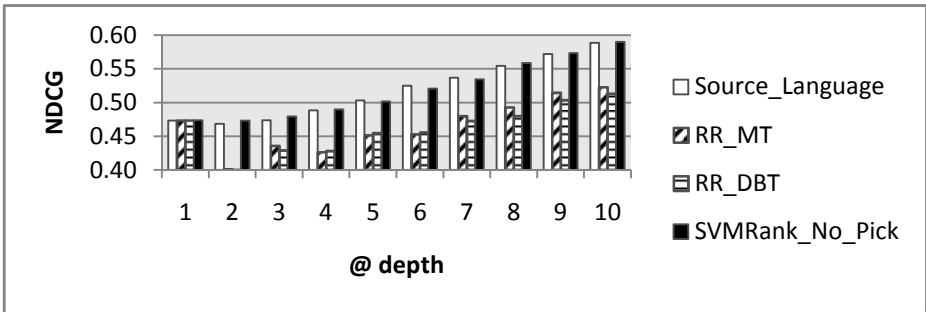


Fig. 6. Comparing monolingual Results to merged results when query picking failed

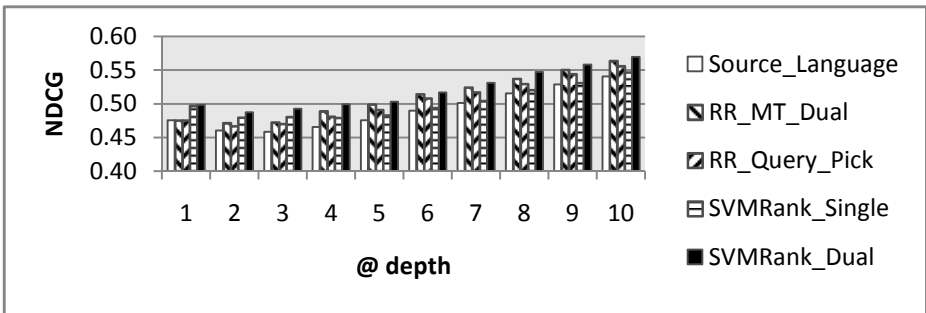


Fig. 7. Comparing monolingual results to RR\_Dual, SVM<sup>Rank</sup>\_Single, and SVM<sup>Rank</sup>\_Dual

## 7 Conclusion

In this paper we examined the use of query picking in performing query translation and in predicting the success of cross language and multilingual web search. For query translation we experimented with different translation techniques, namely machine translation, dictionary-based translation, and query picking. When query picking was successful, picked queries yielded results that were statistically indistinguishable from the results of source language queries and machine translated queries.

The results show that the success of query picking is a strong indicator to whether cross language search and subsequent multilingual results merging would be successful. Exploiting this finding, we have shown that merging results from query picking with monolingual results in a round robin manner yielded statistically significant improvements over a monolingual baseline. Using a merging scheme based on two SVM<sup>Rank</sup> models yielded further statistically significant improvements over monolingual results and over all round robin based merge techniques. SVM<sup>Rank</sup> yielded up to 3.4 basis points improvement in NDCG over monolingual search across all queries and up to 7.5 basis points when query picking was successful.

For future work, we would like to examine features that would further improve query picking, such as features that capture cross language similarity between multilingual results. We would also like to extend the proposed framework to more than two languages, where query picking may have variable success across different languages.

## References

1. Darwish, K.: Transliteration Mining with Phonetic Conflation and Iterative Training. In: NEWS Workshop, ACL 2010 (2010)
2. Gao, W., Niu, C., Nie, J.-Y., Zhou, M., Hu, J., Wong, K.-F., Hon, H.-W.: Cross-lingual query suggestion using query logs of different languages. In: SIGIR 2007, pp. 463–470 (2007)
3. Gao, W., Niu, C., Zhou, M., Wong, K.F.: Joint Ranking for Multilingual Web Search. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 114–125. Springer, Heidelberg (2009)
4. Gao, W., Niu, C., Nie, J.-Y., Zhou, M., Wong, K.-F., Hon, H.-W.: Exploiting query logs for cross-lingual query suggestions. ACM Trans. Inf. Syst. 28, 1–33 (2010)
5. He, X.: Using Word-Dependent Transition Models in HMM based Word Alignment for Statistical Machine Translation. In: ACL 2007 2nd SMT Workshop (2007)
6. Kishida, K.: Prediction of performance of cross-language information retrieval using automatic evaluation of translation. Library & Info. Science Research 30(2), 138–144 (2008)
7. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open Source Toolkit for Statistical Machine Translation. In: ACL 2007, demo. session, Prague, Czech Republic (June 2007)
8. Joachims, T.: Training Linear SVMs in Linear Time. In: Proceedings of the ACM Conference on Knowledge Discovery and Data Mining, KDD (2006)
9. Lee, C.J., Chen, C.H., Kao, S.H., Cheng, P.J.: To translate or not to translate? In: SIGIR 2010 (2010)
10. Lee, Y.S., Papineni, K., Roukos, S., Emam, O., Hassan, H.: Language Model Based Arabic Word Segmentation. In: ACL 2003, pp. 399–406 (2003)
11. Lin, W.C., Chen, H.H.: Merging Mechanisms in Multilingual Information Retrieval. In: Peters, C.A. (ed.) CLEF 2002. LNCS, vol. 2785, pp. 175–186. Springer, Heidelberg (2003)
12. Levow, G.A., Oard, D.W., Resnik, P.: Dictionary-based techniques for cross-language information retrieval. Info. Processing and Management Journal 41(3) (May 2005)
13. Metzler, D., Croft, W.B.: Combining the language model and inference network approaches to retrieval. Information Processing and Management 40(5), 735–750 (2004)
14. Pirkola, A.: The Effects of Query Structure and Dictionary setups in Dictionary-Based Cross-language Information Retrieval. In: SIGIR 1998, pp. 55–63 (1998)

15. Si, L., Callan, J.: CLEF 2005: Multilingual Retrieval by Combining Multiple Multilingual Ranked Lists. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 121–130. Springer, Heidelberg (2006)
16. Tsai, M.F., Wang, T.Y., Chen, H.H.: A Study of Learning a Merge Model for Multilingual Information Retrieval. In: SIGIR 2008 (2008)
17. Udupa, R., Saravanan, K., Bakalov, A., Bhole, A.: They Are Out There, If You Know Where to Look: Mining Transliterations of OOV Query Terms for Cross-Language Information Retrieval. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 437–448. Springer, Heidelberg (2009)
18. Voorhees, E.M.: Evaluation by highly relevant documents. In: Proceedings of SIGIR, pp. 74–82 (2001)
19. Wang, J., Oard, D.W.: Combining Bidirectional Translation and Synonymy for Cross-language Information Retrieval. In: SIGIR 2006, pp. 202–209 (2006)
20. Wu, D., He, D., Ji, H., Grishman, R.: A study of using an out-of-box commercial MT system for query translation in CLIR. In: Workshop on Improving non-English Web Searching, CIKM 2008 (2008)

# Balancing Exploration and Exploitation in Learning to Rank Online

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke

ISLA, University of Amsterdam

{K.Hofmann, S.A.Whiteson, deRijke}@uva.nl

**Abstract.** As retrieval systems become more complex, *learning to rank* approaches are being developed to automatically tune their parameters. Using *online* learning to rank approaches, retrieval systems can learn directly from implicit feedback, while they are running. In such an online setting, algorithms need to both explore new solutions to obtain feedback for effective learning, and exploit what has already been learned to produce results that are acceptable to users. We formulate this challenge as an *exploration-exploitation dilemma* and present the first online learning to rank algorithm that works with implicit feedback and balances exploration and exploitation. We leverage existing learning to rank data sets and recently developed click models to evaluate the proposed algorithm. Our results show that finding a balance between exploration and exploitation can substantially improve online retrieval performance, bringing us one step closer to making online learning to rank work in practice.

## 1 Introduction

Information retrieval (IR) systems are becoming increasingly complex. For example, web search engines combine hundreds of ranking features that each capture a particular aspect of a query, candidate documents, and the match between the two. In heavily used search engines these combinations are carefully tuned to fit users' needs.

For automatically tuning the parameters of such a system, machine learning algorithms are invaluable [13]. Most methods employ supervised learning, i.e., algorithms are trained on examples of relevant and non-relevant documents for particular queries.

While for some applications, such as web search, large amounts of data are available for training, for many environments such data is not available. For example, when deploying a search engine for a local library or company intranet, collecting large amounts of training data required for supervised learning may not be feasible [19]. Even in environments where training data is available, it may not capture typical information needs and user preferences perfectly [15], and cannot anticipate future changes in user needs.

A promising direction for addressing this problem are online approaches for learning to rank [9, 25, 26]. These work in settings where no training data is available before deployment. They learn directly from implicit feedback inferred from user interactions, such as clicks, making it possible to adapt to users throughout the lifetime of the system.

However, collecting a broad enough set of implicit feedback to enable effective online learning is difficult in practice. An online algorithm can observe feedback only on the document lists it presents to the user. This feedback is strongly biased towards the

top results, because users are unlikely to examine lower ranked documents [20]. Therefore, effective learning is possible only if the system experiments with new rankings.

Recent online learning to rank approaches address this problem through exploration, for example by interleaving a document list produced by the current best solution with that of a (randomly selected) exploratory solution [25, 26]. However, this purely exploratory behavior may harm the quality of the result list presented to the user. For example, once the system has found a reasonably good solution, most exploratory document lists will be worse than the current solution.

In this paper we frame this fundamental problem as an *exploration–exploitation dilemma*. If the system presents only document lists that it expects will satisfy the user, it cannot obtain feedback on other, potentially better, solutions. However, if it presents document lists from which it can gain a lot of new information, it risks presenting bad results to the user during learning. Therefore, to perform optimally, the system must *explore* new solutions, while also maintaining satisfactory performance by *exploiting* existing solutions. To make online learning to rank for IR work in a realistic setting, we need to find ways to balance exploration and exploitation.

We present the first algorithm that balances exploration and exploitation in a setting where only implicit feedback is available. Our approach augments a recently developed purely exploratory algorithm that learns from implicit feedback [25] with a mechanism for controlling the rate of exploration. We assess the resulting algorithm using a novel evaluation framework that leverages standard learning to rank datasets and models of users’ click behavior. Our experiments are the first to confirm that finding a proper balance between exploration and exploitation can improve online performance. We also find that surprisingly little exploration is needed for effective learning. These results bring us one step closer to making online learning to rank work in practice.

## 2 Related Work

While our method is the first to balance exploration and exploitation in a setting where only implicit feedback is available, a large body of research addresses related problems.

Most work in learning to rank has focused on supervised learning approaches that learn from labeled training examples [13]. A limitation of these approaches is that they cannot use the copious data that can be easily collected while users interact with the search engine. Such implicit feedback directly captures information from the actual users of the system [15]. In particular, preferences between documents [10] and between document lists [18] can be inferred and have been shown to contain sufficient information for learning effective ranking functions [9].

To make learning from implicit feedback effective, online approaches need to explore. Methods based on active learning systematically select document pairs so as to maximize the expected information gain [16, 24]. Two recently developed stochastic methods use interleaved document lists to infer relative preferences between an exploratory and an exploitative ranking function [25, 26]. One algorithm compares a fixed set of ranking functions and selects the best one [26]. The other algorithm, on which our approach is based, uses relative feedback about two ranking functions for stochastic gradient descent [25].

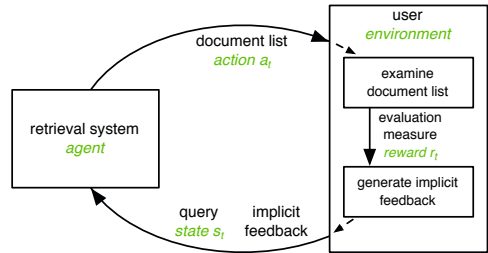
While recent online learning to rank methods provide ways to explore, they do not address how to balance exploration and exploitation. Related research on ad placement and result list diversification has investigated how to balance these factors, but assumes explicit feedback and does not generalize over queries and documents [12, 17].

### 3 Method

In this section, we formalize the problem of online learning to rank for IR, describe a baseline learning algorithm, and extend it to balance exploration and exploitation.

**Problem formulation.** Our formulation of learning to rank for IR differs from most other work in learning to rank in that we consider a continuous cycle of interactions between users and the search engine. A natural fit for this problem are formalizations from reinforcement learning (RL), a branch of machine learning in which an algorithm learns by trying out actions (e.g., document lists) that generate rewards (e.g., an evaluation measure such as AP or NDCG) from its environment (e.g., users) [21]. Using this formalization allows us to describe this problem in a principled way and to apply concepts and solutions from this well-studied area.

Figure 1 shows the interaction cycle. A user submits a query to a retrieval system, which generates a document list and presents it to the user. The user interacts with the list, e.g., by clicking on links, from which the retrieval system infers feedback about the quality of the presented document list. This problem formulation directly translates to an RL problem (cf., Figure 1, terminology in italics) in which the retrieval system tries, based only on implicit feedback, to maximize a hidden reward signal that corresponds to some evaluation measure. We make the simplifying assumption that queries are independent, i.e., queries are submitted by different users and there are no sessions. This renders the problem a *contextual bandit problem*, a well-studied type of RL problem [1, 11].



**Fig. 1.** The IR problem modeled as a contextual bandit problem, with IR terminology in black and corresponding RL terminology in green and italics

Since our hypothesis is that balancing exploration and exploitation improves retrieval performance *while learning*, we need to measure this aspect of performance. Previous work in learning to rank for IR has considered only final performance, i.e., performance on unseen data after training is completed [13], and, in the case of active learning, learning speed in terms of the number of required training samples [24].

As is common in RL, we measure *cumulative reward*, i.e., the sum of rewards over all queries addressed during learning [21]. Many definitions of cumulative reward are possible, depending on the modeling assumptions. We assume an *infinite horizon problem*, a model that is appropriate for IR learning to rank problems that run indefinitely. Such problems include a *discount factor*  $\gamma \in [0, 1)$  that weights immediate rewards

higher than future rewards. One way to interpret the discount factor is to suppose that there is a  $1 - \gamma$  probability that the task will terminate at each timestep (e.g., users may abandon the retrieval system). Rewards are thus weighted according to the probability that the task will last long enough for them to occur. Then, cumulative reward is defined as the discounted infinite sum of rewards  $r_i$ :  $C = \sum_{i=1}^{\infty} \gamma^{i-1} r_i$ .

**Baseline learning approach.** Our approach builds off a gradient-based policy search algorithm called Dueling Bandit Gradient Descent (DBGD) [25]. This algorithm is particularly suitable for online learning to rank for IR because it generalizes over queries, requires only relative evaluations of the quality of two document lists, and infers such comparisons from implicit feedback [18].

This approach learns a ranking function consisting of a weight vector  $w$  for a linear weighted combinations of feature vectors. Thus, to rank a set of documents  $D$  given a query  $q$ , feature vectors  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_D\}$  that describe the relation between  $D$  and  $q$  are produced. Next, scores  $S$  for each document are produced using  $S = wX$ . Finally, documents are ranked by their scores to generate a ranked document list  $l_w$ .

Algorithm 1 summarizes this approach. It takes as input a comparison method  $f(l_1, l_2)$ , that compares two document lists, and three parameters, the step sizes  $\alpha$  and  $\delta$ , and an initial weight vector  $w_0$ . At each timestep  $t$ , the algorithm observes a query  $q_t$  from which two document lists are produced: one exploitative, one exploratory. The exploitative list is produced from

---

**Algorithm 1** Baseline algorithm, based on [25]

---

```

1: Input:  $f(l_1, l_2), \alpha, \delta, w_0$ 
2: for query  $q_t$  ( $t = 1..T$ ) do
3:   Sample unit vector  $u_t$  uniformly.
4:    $w'_t \leftarrow w_t + \delta u_t$  // generate exploratory  $w$ 
5:   if  $f(l(w_t), l(w'_t))$  then
6:      $w_{t+1} \leftarrow w_t + \alpha w_t$  // update exploitative  $w$ 
7:   else
8:      $w_{t+1} \leftarrow w_t$ 
9: return  $w_{t+1}$ 

```

---

the current exploitative weight vector  $w_t$ , found to perform best up to the current timestep  $t$ . The exploratory list is produced from an exploratory weight vector  $w'_t$ , which is generated by moving  $w_t$  in a random direction  $u_t$  by a step of size  $\delta$ . The exploitative and exploratory lists are then compared using a function  $f(l_1, l_2)$ . If the exploratory weight vector  $w'_t$  is judged to have produced the better document ranking, the current exploitative weight vector  $w_t$  is updated by moving it towards  $w'_t$  by a step size  $\alpha$ .

For the comparison method  $f(l_1, l_2)$ , several implementations have been suggested [8, 18]. We chose a variant of the *balanced interleave method* as it is efficient, easy to implement, and was found to be more reliable than the similar *team-draft method* both in [8] and in our own preliminary experiments. This method takes as input two document lists and constructs an interleaved result list by randomly selecting a starting list and then interleaving the two lists so that presentation bias between the two lists is minimized. After observing user clicks on the result list, a preference between the lists is inferred as follows. The rank of the lowest clicked document  $N$  is identified. Then, for each list the number of clicked documents within the top  $N$  is counted. The list that received more clicks in its top  $N$  is preferred. Ties are ignored.

---

<sup>1</sup> In [25],  $\gamma$  denotes the exploitation step size. We use  $\alpha$  to avoid confusion with the discount factor  $\gamma$ .



**Balancing exploration and exploitation.** Given an appropriate function for comparing document lists, the baseline algorithm described above learns effectively from implicit feedback. However, the algorithm always explores, i.e., it constructs the result list in a way that minimizes bias between the exploratory and exploitative document lists, which is assumed to produce the best feedback for learning. We now present a comparison function  $f(l_1, l_2)$  that does allow balancing exploration and exploitation.

In contrast to previous work, we alter the balanced interleave function to interleave documents probabilistically. Instead of randomizing only the starting position and then interleaving documents deterministically, we randomly select the list to contribute the document at each rank of the result list. In expectation, each list contributes documents to each rank equally often.

We employ a method for balancing exploration and exploitation that is inspired by  $\epsilon$ -greedy, a commonly used exploration strategies in RL [22]. In  $\epsilon$ -greedy exploration, the agent selects an action with probability  $\epsilon$  at each timestep. With probability  $1 - \epsilon$ , it selects the greedy action, i.e., the action with the highest currently estimated value.

Our probabilistic interleave algorithm, which supplies the comparison method required by DBGD, is shown in Algorithm 2. The algorithm takes as input two document lists  $l_1$  and  $l_2$ , and an exploration rate  $k$ . For each rank of the result list to be filled, the algorithm randomly picks one of the two result lists (biased by the exploration rate  $k$ ). From the selected list, the highest-ranked document that is not yet in the combined result list is added at this rank. The result list is displayed to the user and clicks  $C$  are observed. Then, for each clicked document, a click is attributed to that list if the document is in the top  $N$  of the list, where  $N$  is the lowest-ranked click.

The exploration rate  $k \in [0.0, 0.5]$  controls the relative amount of exploration and exploitation, similar to  $\epsilon$ . It determines the probability with which a list is selected to contribute a document to the interleaved result list at each rank. When  $k = 0.5$ , an equal number of documents are presented to the user in expectation. As  $k$  decreases, more documents are contributed by the exploitative list, which is expected to improve the quality of the result list but produce noisier feedback.

As  $k$  decreases, more documents from the exploitative list are presented, which introduces bias for inferring feedback. The bias linearly increases the expected number of clicks on the exploitative list and reduces the expected number of clicks on the exploratory list. We can partially compensate for this bias since  $E[c_2] = \frac{n_1}{n_2} * E[c_1]$ , where  $E[c_i]$  is the expected number of clicks within the top  $N$  of list  $l_i$ , and  $n_i$  is the number

---

**Algorithm 2**  $f(l_1, l_2) - k$ -greedy comparison of document lists

---

```

1: Input:  $l_1, l_2, k$ 
2: initialize empty result list  $I$ 
   // construct result list
3: for rank  $r$  in  $(1..10)$  do
4:    $L \leftarrow l_1$  with probability  $k, l_2$  with probability  $1 - k$ 
5:    $I[r] \leftarrow$  first element of  $L \notin I$ 
6: display  $I$  and observe clicked elements  $C$ 
7:  $N = \text{length}(C); c_1 = c_2 = 0$ 
8: for  $i$  in  $(1..N)$  do
9:   if  $C[i] \in l_1[1 : N]$  then
10:     $c_1 = c_1 + 1$  // count clicks on  $l_1$ 
11:   if  $C[i] \in l_2[1 : N]$  then
12:     $c_2 = c_2 + 1$  // count clicks on  $l_2$ 
13:  $n_1 = |l_1[1 : N] \cap I[1 : N]|$  // compensate for bias
14:  $n_2 = |l_2[1 : N] \cap I[1 : N]|$ 
15:  $c_2 = \frac{n_1}{n_2} * c_1$ 
16: return  $c_1 < c_2$ 

```

---

of documents from  $l_i$  that were displayed in the top  $N$  of the interleaved result list. This compensates for the expected number of clicks, but leaves some bias in the expected number of times each document list is preferred. While perfectly compensating for bias is possible, it would require making probabilistic updates based on the observed result. This would introduce additional noise, creating a bias/variance trade-off. Preliminary experiments show that the learning algorithm is less susceptible to increased bias than to increased noise. Therefore we use this relatively simple, less noisy bias correction.

## 4 Experiments

Evaluating the ability of an algorithm to maximize cumulative performance in an online IR setting poses unique experimental challenges. The most realistic experimental setup – in a live setting with actual users – is risky because users may get frustrated with bad search results. The typical TREC-like setup used in supervised learning to rank for IR is not sufficient because information on user behavior is missing.

To address these challenges, we propose an evaluation setup that simulates user interactions. This setup combines datasets with explicit relevance judgments that are typically used for supervised learning to rank with recently developed click models. Given a dataset with queries and explicit relevance judgments, interactions between the retrieval system and the user are simulated (c.f., the box labeled “user/environment” in Figure 1). Submitting a query is simulated by random sampling from the set of queries. After the system has generated a result list for the query, feedback is generated using a click model and the relevance judgments provided with the dataset. Note that the explicit judgments from the dataset are not directly shown to the retrieval system but rather used to simulate the user feedback and measure cumulative performance.

**Click model.** Our click model is based on the Dependent Click Model (DCM) [6, 7], a generalization of the cascade model [3]. The model posits that users traverse result lists from top to bottom, examining each document as it is encountered. Based on this examination, the user decides whether to click on the document or skip it. After each clicked document the user decides whether or not to continue examining the document list. Since the DCM has been shown to effectively predict users’ click behavior [7], we believe it is a good model for generating implicit feedback.

When a user examines a document in the result list, they do not know the true relevance label of the document. However, aspects of the document’s representation in the result list (e.g., title) make it more likely that a document is clicked if it is relevant. Using this assumption, the ground truth relevance judgments provided in explicitly annotated learning to rank datasets, and the process put forward by the DCM, we define the following model parameters. Relevant documents are clicked with a probability  $p(c|R)$ , the probability of a click given that a document is relevant. Non-relevant documents can attract (noisy) clicks, with probability  $p(c|NR)$ . After clicking a document, the user may be satisfied with the results and stop examination with probability  $p(s|R)$ , the probability of stopping examination after clicking on a relevant document. The probability of stopping after visiting a non-relevant document is denoted by  $p(s|NR)$ .

To instantiate this click model we need to define click and stop probabilities. When DCM is trained on large click logs, probabilities are estimated for individual query-

document pairs, while marginalizing over the position at which documents were presented in the training data. In our setting, learning these probabilities directly is not possible, because no click log data is available. Therefore we instantiate the model heuristically, making choices that allow us to study the behavior of our approach in various settings. Setting these probabilities heuristically is reasonable because learning outcomes for the gradient descent algorithm used in this paper are influenced mainly by how much more likely users are to click on relevant and non-relevant documents. Thus, this ratio is more important than the actual numbers used to instantiate the model.

Table 1 gives an overview of the click models used in our experiments. First, to obtain an upper bound on the performance that could be obtained if feedback was deterministic, we define a *perfect* model, where all relevant documents are clicked and no non-relevant documents are clicked. The two realistic models are based on typical user behavior in web search [2, 6], because 8 of the 9 datasets we use implement web search tasks (see below). In a navigational task, users look for a specific document they know to exist in a collection, e.g., a company’s homepage. Typically, it is easy to distinguish relevant and non-relevant documents and the probability of stopping examination after a relevant hit is high. Therefore, our *navigational* model is relatively reliable, with a high difference between  $p(c|R)$  and  $p(c|NR)$ . In an informational task, users look for information about a topic, which can be distributed over several pages. Here, users generally know less about what page(s) they are looking for and clicks tend to be noisier. This behavior leads to the *informational* model, which is much noisier than the navigational model.

**Table 1.** Overview of the click models used

<i>model</i>	$p(c R)$	$p(c NR)$	$p(s R)$	$p(s NR)$
<i>perfect</i>	1.0	0.0	0.0	0.0
<i>navigational</i>	0.95	0.05	0.9	0.2
<i>informational</i>	0.9	0.4	0.5	0.1

**Data.** We conducted our experiments using two standard collections for learning to rank: letor 3.0 and letor 4.0 [14]. In total, these two collections comprise 9 datasets. Each consists of queries for which features were extracted from a document collection, together with relevance judgements for the considered query-document pairs.

The datasets were compiled from different sources: the 106 queries in OHSUMED are based on a log of a search engine for scientific abstracts drawn from the MedLine database. The remaining datasets are based on Web Track collections run between 2003 and 2008 at TREC. HP2003, HP2004, NP2003, NP2004, TD2003 and TD2004 implement homepage finding, named-page finding, and topic distillation tasks, using a crawl of web pages within the .gov domain. These datasets contain between 50–150 queries each, with about 1000 judged documents per query. MQ2007 and MQ2008 are based on the 2007 and 2008 Million Query track at TREC and use the “.GOV2” collection. These two datasets contain substantially more queries, 1700 and 800 respectively, but much fewer judged documents per query.

The datasets based on the TREC Web track use binary relevance judgments, while OHSUMED, MQ2007 and MQ2008 are judged on a 3-point scale from 0 (non-relevant) to 2 (highly relevant). In all experiments we use binary relevance judgments. For the three datasets that originally contain graded judgments, we treat all judgments greater

than zero as relevant. In preliminary experiments with graded relevance, we obtained results nearly identical to those with the simpler binary judgments<sup>2</sup>

Each dataset comes split up for machine learning experiments using 5-fold cross-validation. We use the training sets for training during the learning cycle and for calculating cumulative performance, and the test sets for measuring final performance.

**Runs.** In all experiments we initialize the starting weight vector  $w_0$  randomly, and use the best performing parameter settings from [25]:  $\delta = 1$  and  $\alpha = 0.01$ . Our *baseline* is Algorithm 1 based on [25], which corresponds to a purely exploratory setting of  $k = 0.5$  in our extended method. Against this baseline we compare *exploit* runs that balance exploration and exploration by varying the exploration rate  $k$  between 0.4 and 0.1 as shown in Algorithm 2. All experiments are run for 1000 iterations.

**Discounting.** Because our problem formulation assumes an infinite horizon, cumulative performance is defined as an infinite sum of discounted rewards (cf. §3). Since experiments are necessarily finite, we cannot compute this infinite sum exactly. However, because the sum is discounted, rewards in the far future have little impact and cumulative performance can be approximated with a sufficiently long finite experiment.

In our experiments, we set the discount factor  $\gamma = 0.995$ . This choice can be justified in two ways. First, it is typical of discount factors used when evaluating RL methods [21]. Choosing a value close to 1 ensures that future rewards have significant weight and thus the system must explore in order to perform well. Second, at this value of  $\gamma$ , cumulative performance can be accurately estimated with the number of queries in our datasets. Since rewards after 1000 iterations have a weight of 1% or less, our finite runs are good approximations of true cumulative performance.

**Evaluation measures.** We use cumulative NDCG on the result list presented to the user to measure cumulative performance of the system. We define cumulative reward as the discounted sum of NDCG that the retrieval system accrues throughout the length of the experiment. Final performance is reported in terms of NDCG on the test set. Though omitted here due to lack of space, we also conducted experiments measuring cumulative and final performance based on MAP and MRR and observed similar results.

For each dataset we repeat all runs 25 times and report results averaged over folds and repetitions. We test for significant differences with the baseline runs ( $k = 0.5$ , the first column of Table 3) using a two-sided student's t-test. Runs that significantly outperform the exploratory baseline are marked with  $\Delta$  ( $p < 0.05$ ) or  $\blacktriangle$  ( $p < 0.01$ ).

## 5 Results and Discussion

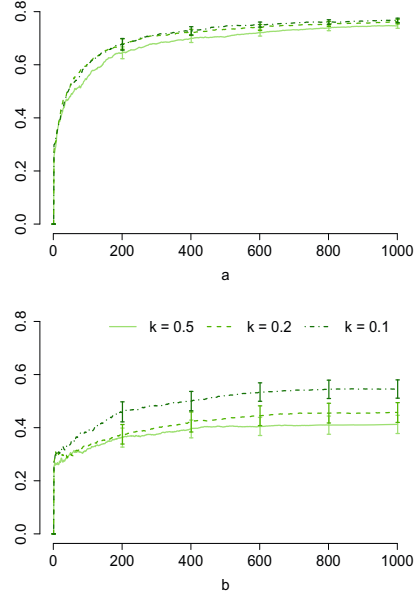
The main goal of this paper is to show that balancing exploration and exploitation in online learning to rank for IR can improve cumulative performance. However, such a result is meaningful only if our baseline learning algorithm learns effectively in this setting. Therefore, before turning to our main results, we assess our baseline algorithm.

<sup>2</sup> The reason appears to be that the learning algorithm works with very coarse feedback, so more finely grained feedback has little influence on the reliability of inferred judgments.

**Baseline learning approach.** Figure 2 shows example learning curves for the dataset *NP2003* at different settings of  $k$  and for all click models. Learning curves for all other datasets are qualitatively similar, and we omit those due to space restrictions. The figure shows final performance in terms of NDCG on the test sets after each learning step. We see that final performance improves over time in all settings. As expected, learning is faster when feedback is more reliable. For the idealized *perfect* click model, final performance after 1000 iterations ranges between 0.777 and 0.785 for different settings of  $k$ . For the noisy *informational* click model at the same settings, final performance is between 0.412 and 0.546. Although final performance drops substantially as implicit feedback becomes extremely noisy, we find that as long as there is a signal, i.e., relevant documents are more likely to be clicked than non-relevant ones, performance improves over time for all datasets.

We find an interaction effect between click model and exploration rate. When the click model is reliable, there is no significant difference between the final performance at different settings of  $k$ . However, in the *informational* click model, variance increases, and there is a large difference between final performance at different settings of  $k$ . This is a direct and expected consequence of the noise in inferred feedback. More surprising is that final performance improves for smaller  $k$ , since we expected feedback to be the most reliable for the fully exploratory setting  $k = 0.5$ . Instead, it appears that, since bias is only partially compensated for (cf., §3), the remaining bias at lower values of  $k$  smoothes over some of the noise in the click model. At lower exploration rates, fewer results from the exploratory list are presented and it becomes harder for the exploratory list to win the comparison. Thus, instead of noisier updates, the algorithm makes fewer, more reliable updates that on average result in greater performance gains.

As a final sanity check, we calculate standard evaluation measures that are typically used to evaluate supervised learning to rank methods. Results for the *perfect* click model



**Fig. 2.** Final performance (with 5% confidence intervals) over time for the dataset *NP2003* for a) *navigational*, and b) *informational* click models and  $k \in \{0.1, 0.2, 0.5\}$

**Table 2.** NDCG@10, Precision@10, and MAP for the baseline algorithm

	NDCG@10	P@10	MAP
<i>HP2003</i>	0.792	0.102	0.721
<i>HP2004</i>	0.770	0.096	0.676
<i>NP2003</i>	0.761	0.090	0.649
<i>NP2004</i>	0.787	0.093	0.659
<i>TD2003</i>	0.296	0.152	0.231
<i>TD2004</i>	0.298	0.236	0.206
<i>OHSUMED</i>	0.422	0.488	0.437
<i>MQ2007</i>	0.375	0.335	0.410
<i>MQ2008</i>	0.488	0.238	0.447

and  $k = 0.5$  after 1000 iterations are listed in Table 2. Despite the limited information available to the algorithm (relative quality of the result list instead of explicit relevance judgment per document), performance is competitive with current supervised learning to rank algorithms [13]. Note that we did not tune parameters of the algorithm for final performance, so further improvements may be possible.

**Balancing exploration and exploitation.** We now turn to our main research question: *Can balancing exploration and exploitation improve online performance?* Our results are shown in Table 3. Cumulative performance for all *exploit* runs ( $k \in [0.1, 0.4]$ ) is compared to the purely exploratory baseline ( $k = 0.5$ ). Best runs per row are highlighted in bold and significant differences are marked as described above.

Our focus is on comparing relative performance per dataset. Starting with the *perfect* click model, we see that for all datasets the baseline is outperformed by all lower settings of  $k$ . For  $k < 0.4$  all improvements over the baseline are statistically significant. The improvements range from 4.1% (*OHSUMED*) to 12.35% (*NP2004*).

We observe similar results for the *navigational* click model. For all datasets, there are several lower settings of  $k$  where performance improves over the baseline. For all but one dataset (*MQ2007*), these improvements are statistically significant. Improvements range from 0.54% (*MQ2007*) to 21.9% (*NP2003*).

The trend continues for the *informational* click model. Again, the purely exploratory baseline is outperformed by more exploitative settings in all cases. For 7 out of 9 cases the improvements are statistically significant. The improvement ranges up to 35.9% for the dataset *HP2004*.

We find that for all click models and all datasets balancing exploration and exploitation can significantly improve online performance over the purely exploratory baseline. Comparing cumulative performance listed in Table 3 with final performance in Table 2, we find that cumulative performance does not depend only on final performance. For example, NDCG@10 and MAP for *HP2003* are much higher than for *OHSUMED*, but cumulative performance is very similar (precision scores are low for *HP2003* because

**Table 3.** Results. Cumulative NDCG for *baseline* ( $k = 0.5$ ) and *exploit* ( $k \in [0.1, 0.4]$ ) runs.

$k$	0.5	0.4	0.3	0.2	0.1
<i>click model: perfect</i>					
<i>HP2003</i>	119.91	125.71 <sup>▲</sup>	129.99 <sup>▲</sup>	<b>130.55<sup>▲</sup></b>	128.50 <sup>▲</sup>
<i>HP2004</i>	109.21	111.57	118.54 <sup>▲</sup>	<b>119.86<sup>▲</sup></b>	116.46 <sup>▲</sup>
<i>NP2003</i>	108.74	113.61 <sup>▲</sup>	117.44 <sup>▲</sup>	<b>120.46<sup>▲</sup></b>	119.06 <sup>▲</sup>
<i>NP2004</i>	112.33	119.34 <sup>▲</sup>	124.47 <sup>▲</sup>	<b>126.20<sup>▲</sup></b>	123.70 <sup>▲</sup>
<i>TD2003</i>	82.00	84.24	88.20 <sup>▲</sup>	<b>89.36<sup>▲</sup></b>	86.20 <sup>▲</sup>
<i>TD2004</i>	85.67	90.23 <sup>▲</sup>	91.00 <sup>▲</sup>	<b>91.71<sup>▲</sup></b>	88.98 <sup>△</sup>
<i>OHSUMED</i>	128.12	130.40 <sup>▲</sup>	131.16 <sup>▲</sup>	<b>133.37<sup>▲</sup></b>	131.93 <sup>▲</sup>
<i>MQ2007</i>	96.02	97.48	98.54 <sup>▲</sup>	<b>100.28<sup>▲</sup></b>	98.32 <sup>▲</sup>
<i>MQ2008</i>	90.97	92.99 <sup>▲</sup>	94.03 <sup>▲</sup>	<b>95.59<sup>▲</sup></b>	95.14 <sup>▲</sup>
<i>click model: navigational</i>					
<i>HP2003</i>	102.58	109.78 <sup>▲</sup>	<b>118.84<sup>▲</sup></b>	116.38 <sup>▲</sup>	117.52 <sup>▲</sup>
<i>HP2004</i>	89.61	97.08 <sup>▲</sup>	99.03 <sup>▲</sup>	103.36 <sup>▲</sup>	<b>105.69<sup>▲</sup></b>
<i>NP2003</i>	90.32	100.94 <sup>▲</sup>	105.03 <sup>▲</sup>	108.15 <sup>▲</sup>	<b>110.12<sup>▲</sup></b>
<i>NP2004</i>	99.14	104.34 <sup>△</sup>	110.16 <sup>▲</sup>	112.05 <sup>▲</sup>	<b>116.00<sup>▲</sup></b>
<i>TD2003</i>	70.93	75.20 <sup>▲</sup>	<b>77.64<sup>▲</sup></b>	77.54 <sup>▲</sup>	75.70 <sup>△</sup>
<i>TD2004</i>	78.83	80.17	82.40 <sup>△</sup>	<b>83.54<sup>▲</sup></b>	80.98
<i>OHSUMED</i>	125.35	126.92 <sup>△</sup>	127.37 <sup>▲</sup>	<b>127.94<sup>▲</sup></b>	127.21
<i>MQ2007</i>	95.50	94.99	95.70	<b>96.02</b>	94.94
<i>MQ2008</i>	89.39	90.55	91.24 <sup>△</sup>	<b>92.36<sup>▲</sup></b>	92.25 <sup>▲</sup>
<i>click model: informational</i>					
<i>HP2003</i>	59.53	63.91	61.43	70.11 <sup>△</sup>	<b>71.19<sup>▲</sup></b>
<i>HP2004</i>	41.12	52.88 <sup>▲</sup>	48.54 <sup>△</sup>	<b>55.88<sup>▲</sup></b>	55.16 <sup>▲</sup>
<i>NP2003</i>	53.63	53.64	57.60	58.40	<b>69.90<sup>▲</sup></b>
<i>NP2004</i>	60.59	63.38	64.17	63.23	<b>69.96<sup>△</sup></b>
<i>TD2003</i>	52.78	52.95	51.58	55.76	<b>57.30</b>
<i>TD2004</i>	58.49	61.43	59.75	62.88 <sup>△</sup>	<b>63.37</b>
<i>OHSUMED</i>	121.39	123.26	124.01 <sup>△</sup>	<b>126.76<sup>▲</sup></b>	125.40 <sup>▲</sup>
<i>MQ2007</i>	91.57	<b>92.00</b>	91.66	90.79	90.19
<i>MQ2008</i>	86.06	87.26	85.83	<b>87.62</b>	86.29

there are few relevant documents in general, and are not a good indicator of the relative quality of result rankings). We find that the main factors affecting cumulative performance are the speed of learning and how effectively early learning gains are exploited. This confirms that measuring final performance is not enough when evaluating online learning to rank algorithms.

The best setting for exploration rate  $k$  is 0.1 or 0.2 in all but two cases. A setting of  $k = 0.2$  means that by injecting, on average, only two documents from an exploratory list, the algorithm learns effectively and achieves good cumulative performance while learning. This means that surprisingly little exploration is sufficient for good performance and that the original algorithm (our baseline) explores too much.

While balancing exploration and exploitation improves performance for all datasets, the magnitude of these improvements differs substantially. For example, for the *navigational* click model, the relative improvement between baseline and best setting for *NP2003* is 21.9%, while for *MQ2007* the difference is only 0.54%. This reflects a general difference between datasets obtained from the 2003 and 2004 TREC web tracks and the remaining datasets. The first contain 1000 candidate documents per query, but few relevant documents. Therefore, it is relatively difficult to find a good ranking and minor changes in weights can result in substantially worse result lists. Consequently, the differences between exploratory and exploitative document lists are large, leading to large improvements when more documents from the exploitative list are selected. In contrast, the datasets *MQ2007*, *MQ2008*, and *OHSUMED* contain fewer candidate documents but many more relevant ones. Therefore, exploring does not hurt as much as in other settings and differences between exploratory and exploitative settings tend to be smaller. Note that in realistic settings it is likely that more candidate documents are considered, so the effect of exploiting more is likely to be stronger.

The different instantiations of the click model also result in qualitative differences in cumulative performance. Performance is higher with *perfect* feedback, and decreases as feedback becomes noisier. Performance on some datasets is more strongly affected by noisy feedback. For the *HP*, *NP*, and *TD* datasets, performance for the *informational* model drops substantially. This may again be related to the large number of non-relevant documents in these datasets. As finding a good ranking is harder, noise has a stronger effect. Despite this drop in performance, balancing exploration and exploitation consistently leads to better cumulative performance than the purely exploratory baseline.

## 6 Conclusion

In this paper, we formulated online learning to rank for IR as an RL problem, casting the task as a contextual bandit problem in which only implicit feedback is available. We argued that, in such problems, learning to rank algorithms must balance exploration and exploitation in order to maximize cumulative performance. We proposed the first algorithm to do so in an IR setting with only implicit feedback. This algorithm extends a stochastic gradient descent algorithm with a mechanism for controlling the rate of exploration. Since assessing the performance of such algorithms poses unique challenges, we introduced a evaluation framework based on simulated interaction that can measure the cumulative performance of online methods.

The performance of our method was compared to a purely exploratory baseline, using three click models and nine datasets. We demonstrate that a proper balance between exploration and exploitation can significantly and substantially improve cumulative performance, which confirms our hypothesis. Surprisingly little exploration is needed for good performance, and we analyzed how the reliability of user feedback and differences between datasets affect the balance between exploration and exploitation.

Given this initial evidence of the benefit of balancing exploration and exploitation in online IR, developing new algorithms for this problem is an important goal for future research. In addition to the approach developed in this paper, approaches combining active learning [4, 23] with exploration strategies from RL could be developed.

Our evaluation framework is based on existing learning to rank data collections and a probabilistic model of how users examine result lists and decide whether to follow a link to a document. An interesting future direction is to leverage click log data for evaluation. The main challenge is to account for the fact that not all possible rankings are contained in logs. Possible solutions could be based on click models estimated from such data, like the one underlying the click model used in this paper [5, 7].

Like all simulations, our experiments are based on specific modeling assumptions. In the future, experiments in real-life settings will be indispensable for verifying these assumptions. Interesting questions include how a discount factor should be set for the requirements of specific search environments. In preliminary experiments, we found that our results do not depend on the infinite horizon model, as we obtained qualitatively similar results in a finite horizon setting. Also, while our click model is based on specific validated models, particular instantiations of this model had to be chosen heuristically. Therefore, it would be useful to investigate what instantiations of the click model best reflect the behavior of real users in such environments.

**Acknowledgements.** This research was partially supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430, by the PROMISE Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement no. 258191, by the DuOMAn project carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project nr STE-09-12, by the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.061.814, 612.061.815, 640.004.802, 380-70-011, and by the Center for Creation, Content and Technology (CCCT).

## References

- [1] Barto, A.G., Sutton, R.S., Brouwer, P.S.: Associative search network: A reinforcement learning associative memory. *IEEE Trans. Syst., Man, and Cybern.* 40, 201–211 (1981)
- [2] Broder, A.: A taxonomy of web search. *SIGIR Forum* 36(2), 3–10 (2002)
- [3] Craswell, N., Zoeter, O., Taylor, M., Ramsey, B.: An experimental comparison of click position-bias models. In: *WSDM 2008*, pp. 87–94 (2008)
- [4] Donmez, P., Carbonell, J.G.: Active sampling for rank learning via optimizing the area under the ROC curve. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) *ECIR 2009*. LNCS, vol. 5478, pp. 78–89. Springer, Heidelberg (2009)



- [5] Dupret, G.E., Piwowarski, B.: A user browsing model to predict search engine click data from past observations. In: SIGIR 2008, pp. 331–338 (2008)
- [6] Guo, F., Li, L., Faloutsos, C.: Tailoring click models to user goals. In: WSCD 2009, pp. 88–92 (2009)
- [7] Guo, F., Liu, C., Wang, Y.M.: Efficient multiple-click models in web search. In: WSDM 2009, pp. 124–131 (2009)
- [8] He, J., Zhai, C., Li, X.: Evaluation of methods for relative comparison of retrieval systems based on clickthroughs. In: CIKM 2009, pp. 2029–2032 (2009)
- [9] Joachims, T.: Optimizing search engines using clickthrough data. In: KDD 2002, pp. 133–142 (2002)
- [10] Joachims, T., Granka, L., Pan, B., Hembrooke, H., Gay, G.: Accurately interpreting click-through data as implicit feedback. In: SIGIR 2005, pp. 154–161. ACM Press, New York (2005)
- [11] Langford, J., Zhang, T.: The epoch-greedy algorithm for multi-armed bandits with side information. In: NIPS 2008, pp. 817–824 (2008)
- [12] Langford, J., Strehl, A., Wortman, J.: Exploration scavenging. In: ICML 2008, pp. 528–535 (2008)
- [13] Liu, T.Y.: Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3(3), 225–331 (2009)
- [14] Liu, T.-Y., Xu, J., Qin, T., Xiong, W., Li, H.: Letor: Benchmark dataset for research on learning to rank for information retrieval. In: LR4IR 2007 (2007)
- [15] Radlinski, F., Craswell, N.: Comparing the sensitivity of information retrieval metrics. In: SIGIR 2010, pp. 667–674 (2010)
- [16] Radlinski, F., Joachims, T.: Active exploration for learning rankings from clickthrough data. In: KDD 2007, pp. 570–579 (2007)
- [17] Radlinski, F., Kleinberg, R., Joachims, T.: Learning diverse rankings with multi-armed bandits. In: ICML 2008, pp. 784–791. ACM, New York (2008)
- [18] Radlinski, F., Kurup, M., Joachims, T.: How does clickthrough data reflect retrieval quality? In: CIKM 2008, pp. 43–52 (2008)
- [19] Sanderson, M.: Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval* 4(4), 247–375 (2010)
- [20] Silverstein, C., Marais, H., Henzinger, M., Moricz, M.: Analysis of a very large web search engine query log. *SIGIR Forum* 33(1), 6–12 (1999)
- [21] Sutton, R.S., Barto, A.G.: *Introduction to Reinforcement Learning*. MIT Press, Cambridge (1998)
- [22] Watkins, C.: *Learning from Delayed Rewards*. PhD thesis, Cambridge University (1989)
- [23] Xu, Z., Akella, R., Zhang, Y.: Incorporating diversity and density in active learning for relevance feedback. In: Amati, G., Carpineto, C., Romano, G. (eds.) *ECIR 2007*. LNCS, vol. 4425, pp. 246–257. Springer, Heidelberg (2007)
- [24] Xu, Z., Kersting, K., Joachims, T.: Fast active exploration for link-based preference learning using gaussian processes. In: *ECML PKDD 2010*, pp. 499–514 (2010)
- [25] Yue, Y., Joachims, T.: Interactively optimizing information retrieval systems as a dueling bandits problem. In: *ICML 2009*, pp. 1201–1208 (2009)
- [26] Yue, Y., Broder, J., Kleinberg, R., Joachims, T.: The k-armed dueling bandits problem. In: *COLT 2009* (2009)

# ReFER: Effective Relevance Feedback for Entity Ranking

Tereza Iofciu<sup>1</sup>, Gianluca Demartini<sup>1,\*</sup>,  
Nick Craswell<sup>2</sup>, and Arjen P. de Vries<sup>3</sup>

<sup>1</sup> L3S Research Center, Hannover, Germany  
{iofcIU,demartini}@L3S.de

<sup>2</sup> Microsoft Redmond, WA, USA  
nickcr@microsoft.com

<sup>3</sup> Centrum Wiskunde & Informatica, Amsterdam, Netherland  
arjen@acm.org

**Abstract.** Web search increasingly deals with structured data about people, places and things, their attributes and relationships. In such an environment an important sub-problem is matching a user’s unstructured free-text query to a set of *relevant* entities. For example, a user might request ‘Olympic host cities’. The most challenging general problem is to find relevant entities, of the correct type and characteristics, based on a free-text query that need not conform to any single ontology or category structure. This paper presents an entity ranking relevance feedback model, based on example entities specified by the user or on pseudo feedback. It employs the Wikipedia category structure, but augments that structure with ‘smooth categories’ to deal with the sparseness of the raw category information. Our experiments show the effectiveness of the proposed method, whether applied as a pseudo relevance feedback method or interactively with the user in the loop.

## 1 Introduction

Finding entities of different types is a challenging search task which goes beyond classic document retrieval and also single-type entity retrieval such as expert search [1]. The motivation for this task is that many ‘real searches’ are not looking for documents to learn about a topic, but really seek a list of specific entities: restaurants, countries, films, songs, etc. [10]. Example needs include ‘Formula 1 drivers that won the Monaco Grand Prix’, ‘Female singer and songwriter born in Canada’, ‘Swiss cantons where they speak German’, and ‘Coldplay band members’, just to name few.

This is a new interesting task that goes beyond standard search engine’s matching between user query and document features. In the Entity Ranking (ER) scenario the user is looking for a set of entities of the same type with

---

\* This work is partially supported by the EU Large-scale Integrating Projects Living-Knowledge (contract no. 231126) and Arcomem (contract no. 270239).

some common properties, e.g., ‘countries where I can pay in Euro’. This query is answered by current web search engines with a list of pages on the topic ‘Euro zone’, or ways to pay in Euros, but not with a list of country names.

The complexity of this novel search task lays in the multi-step solution that should be adopted. Firstly, the system has to understand the user query, what is the entity type and which are its properties. Similarly to expert search, the index should contain entities instead of just documents, and the entity type should be represented in and matched against the user query. Therefore, several techniques from research fields such as Information Extraction and Natural Language Processing (NLP) could be used as well in order to first identify entities in a document collection. Moreover, a hierarchy of possible entity types and relations among entities and their types has to be considered [13][16]. Initial attempts to ER have recently been presented. The main approaches build on top of the link structure in the set of entities [12], use passage retrieval techniques, language models [13], or NLP based solutions [6].

In this paper, we propose ReFER: a graph-based method to take advantage of relevance feedback (RFB) in entity retrieval, exploiting either example entities provided by the user, or the top- $k$  results from an ER system. We show how the combination of RFB results with the initial system improves search effectiveness for all runs submitted to the Initiative for the Evaluation of XML Retrieval (INEX)<sup>1</sup> 2008 XML Entity Ranking track. The proposed method is designed based on the Wikipedia setting used at INEX but it could be adapted to other settings such as the one of tag recommendation (i.e., tagged web pages compared to Wikipedia articles belonging to Wikipedia categories).

The rest of this paper is structured as follows. We start with a summary of related work in the area of entity ranking. Section 3 then discusses the two main properties of Wikipedia and how these are exploited in this paper. The proposed algorithm is presented in Section 4. Section 5 then discusses our experimental results, showing how our graph-based method improves performance of previously proposed entity ranking techniques. The final Section summarizes our main conclusions.

## 2 Related Work

The first proposed approaches for ER [3,4,5] mainly focus on scaling efficiently on Web dimension datasets but not much on search quality while we aim to improve effectiveness of the ER task. Approaches for finding entities have also been developed in the Wikipedia context. Pehcevski et al. [12] use link information for improving effectiveness of ER in Wikipedia. Demartini et al. [6] improve ER effectiveness by leveraging on an ontology for refining the Wikipedia category hierarchy. Compared to these approaches, we propose an orthogonal view on the problem that can be applied to any ER approach via RFB. Balog et al. [2] propose a model that considers RFB techniques on the category structure

<sup>1</sup> <http://www.inex.otago.ac.nz/>

of Wikipedia to improve effectiveness of ER. In this paper we show how our method can be effectively applied also on top of their system.

A different approach to the problem is to rank document passages that represent entities. In [20] the authors present an ER system that builds on top of an entity extraction and semantic annotation step followed by running a passage retrieval system using appropriate approaches to re-rank entities. In [7] a similar approach is used to rank entities over time.

A related task is the *entity type ranking* defined in [17]. The goal is to retrieve the most important entity types for a query, e.g., Location, Date, and Organization for the query Australia. Our algorithm exploits entity type information from the entity-category graph in order to find the most important entity types. Another related task is *expert finding* which has been mainly studied in the context of the TREC Enterprise Track [1]. The entity type is fixed to people and the query is finding knowledgeable people about a given topic. Entity ranking goes beyond the single-typed entity retrieval and relevance is more loosely defined. More recently, at the TREC Entity track the task of finding entities related to a given one was studied [15]. As our approach works on the entity category structure, we focus on ER performed in the Wikipedia setting.

### 3 Category Expansion in Wikipedia

This paper presents an entity ranking model based on assigning entities to ‘smooth categories’. This in turn is based on the Wikipedia link and category structure. This section describes the two key properties of Wikipedia we rely on to develop our model. The next section describes the smooth category model.

Wikipedia is a free encyclopedia with 2.7 million English articles written by volunteers.<sup>2</sup> It is a collaborative website with an editorial process governed by a series of policies and guidelines.<sup>3</sup> Wikipedia has two properties that make it particularly useful for ER. The first is that many of its articles are dedicated to an entity, so the entity ranking problem reduces to the problem of ranking such articles. The Wikipedia guidelines prescribe that an entity should have at most one article dedicated to it, according to the *content forking* guidelines. Thus the entity ranking model does not need to eliminate duplicates. Many real-world entities have no Wikipedia page, according to the *notability* guidelines. To be included, an entity should have significant coverage in multiple independent, reliable sources. For example, the model can rank major-league baseball players according to some entity-ranking query, but not players in youth baseball leagues, since youth players rarely meet the notability criteria.

In this setting, a simple ER solution is to rank Wikipedia pages in a standard IR system. If we search in a List Completion manner (i.e. query by example), for ‘John F. Kennedy’ in an index of Wikipedia pages, the top-ranked articles are: ‘John F. Kennedy’, ‘John F. Kennedy International Airport’, ‘John F. Kennedy Jr.’, ‘John F. Kennedy Library’ and ‘John F. Kennedy assassination in popular

<sup>2</sup> <http://en.wikipedia.org/wiki/Wikipedia>

<sup>3</sup> [http://en.wikipedia.org/wiki/Wikipedia:Policies\\_and\\_guidelines](http://en.wikipedia.org/wiki/Wikipedia:Policies_and_guidelines)

culture’. The IR system has succeeded in finding pages relevant to the topic of JFK. However, if the information need were related to finding US presidents, the system has not succeeded. It did not find entities of a similar type. As a concluding remark, note, some articles do not pertain to an entity (e.g., ‘Running’); we have to rely on the entity ranking model to avoid retrieving these.

The second useful property of Wikipedia is its rich link and category structure, with the category structure being of particular interest when finding entities of similar type. Intuitively, one would say that if two entities are related by satisfying an information need, they should have at least one common category. The more common categories two entities belong to, the more related they are likely to be. The usefulness of Wikipedia’s link structure has been confirmed in the INEX entity ranking experiments: participants found that category information, associations between entities and query-dependent link structure improved results over their baselines [18]. However, as Wikipedia is a collaborative effort, no strict rules enforce the guidelines for linking between entities or assigning entities to categories. Entities may belong to many categories describing its different aspects, and no limit exists on the number of categories an entity could get assigned. For example the Wikipedia page describing ‘Albert Einstein’ links to a wide variety of entities, including specific information such as ‘Theory of relativity’ and ‘Nobel Prize in Physics’, but also more generic facts like ‘Germany’ and ‘Genius’. Considering the Wikipedia category structure, ‘Albert Einstein’ belongs to some sixty categories, varying from ‘German Nobel laureates’ and ‘German immigrants to the United States’ to ‘1879 births’.

The categories of a page are not weighted by their importance, so we do not know which is more important, and a page may also be missing from important categories. For example, in our snapshot of Wikipedia the article on South Korea is in the categories: ‘South Korea’, ‘Liberal democracies’ and ‘Divided regions’. There are attributes of South Korea that are not described by categories.

## 4 Link-Based Relevance Feedback for Entity Ranking

In this section we describe ReFER, our RFB algorithm based on the link structure of the Wikipedia model, and we then present ways of integrating it with existing ER systems.

In our model we assume a collection of categories  $C = \{c_1, \dots, c_n\}$  and a collection of entities  $E = \{e_1, \dots, e_m\}$ . Each entity  $e_i$  corresponds to a Wikipedia page, and each category  $c_i$  is a tag describing an entity.

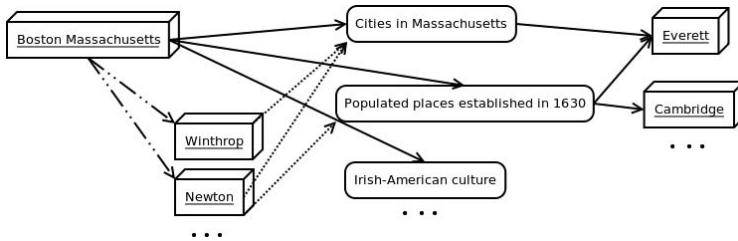
**Definition 1.** *An entity  $e_i$  is a tuple  $\langle uri, desc, C_{e_i}, R_{e_i} \rangle$  where  $uri$  is the entity’s identifier,  $desc$  is a string describing  $e_i$  (given by the Wikipedia article),  $C_{e_i} \subseteq C$  is the set of categories describing the entity and  $R_{e_i} \subseteq E \setminus \{e_i\}$  is the set of entities that are referred to from the Wikipedia url of  $e_i$ .*

One can easily see that given a collection of entities and categories, we can retrieve two types of connections. The first type is between two entities and we denote it with  $link = \langle e_i, e_j \rangle$ . The second type of connection is between an entity

and a category and we denote it with  $edge = \langle e_i, c_j \rangle$ . In addition we distinguish between two types of edges according to the process that created them. The ‘hard’ edges of entity  $e_i$  are the ones that can be directly generated using  $C_{e_i}$ , i.e.,  $C_H(e_i) = \{c | c \in C_{e_i}\}$ . The ‘smooth’ edges can be inferred through the categories of the referred entities, i.e.,  $C_S(e_i) = \{c | c \in C_{e_j} \forall e_j \in R_{e_i}\}$ .

### 4.1 The ReFER Algorithm

Our entity ranking algorithm can be described as propagation of weights through a directed acyclic graph. The graph has nodes in three layers: an ‘input’ layer of entities, an ‘intermediate’ layer of hard and smooth categories and a ranked ‘output’ layer of entities connected to the ‘intermediate’ categories. Weights propagate through graph and is proportional to the number of links, hard edges and smooth edges.



**Fig. 1.** Three layer graph, with input node entity ‘Boston, Massachusetts’. Solid edges indicate hard categories, dashed edges indicate smooth categories.

For the example in Figure 1, if the article on *Boston Massachusetts* is in the category *Cities in Massachusetts*, and links to several pages that are also in that category, then the article’s input node is connected to *Cities in Massachusetts* node via both a hard edge and a smooth edge. In our example, category *Cities in Massachusetts* will be weighted higher than category *Irish-American culture*, as the latter has no smooth edges leading to it. Smooth categories can add extra weight to hard categories, and also make associations with new categories. For ‘South Korea’, the original category that is most strongly supported is ‘Liberal Democracies’, since seven of the articles linked-to by the ‘South Korea’ article are ‘Liberal Democracies’. The page is associated to 26 smooth categories, out of which 14 contain the word Korea. There is though some noise in the smooth categories, like ‘Constitutional Monarchies’ and ‘History of Japan’. In order to reduce the amount of noisy smooth categories for an entity  $e_i$  we filter out the ones with less than 2 entities from  $R_{e_i}$  belonging to them.

Given a query  $q$  we activate a certain set of nodes  $E_q$  as input for our algorithm. Then for each category node in  $C_H(e_i) \cup C_S(e_i)$ , where  $e_i \in E_q$ , we sum the incident edge weights from active input nodes from  $E_q$ . For category  $c_j$  let us denote the total incoming hard-edge weight as  $h_{c_j}$  and smooth-edge weight as  $s_{c_j}$ . In our initial experiments, we noticed that the hard-category ‘coordination’

between the input nodes is important. If there is one category that is common to most of the active input nodes, then that category is extremely important, and should massively outweigh the other categories. This led us to develop the following exponential category weighting heuristic:

$$cw(c_j) = \frac{\alpha^{h_{c_j}} + s_{c_j}}{\log(\text{catsize}(c_j) + \beta)}, \quad (1)$$

where  $\text{catsize}(c_j)$  is the number of Wikipedia pages in the category  $c_j$  and  $\alpha$  and  $\beta$  are parameters<sup>4</sup>,  $\beta$  being used so that the logarithm does not return negative values. The log down-weights very large categories, since these are unlikely to be discriminative. Akin to stopword removal, we eliminate categories with many entities (in our setup we considered a threshold of 1000 entities).

If there is a category that is common to all input nodes in  $E_q$ , then it will have high  $h$  and a much higher weight than any other category. For example, if the input nodes are a number of entities in the category *Cities in Massachusetts*, then that category will dominate the rest of the entity ranking process. If there is not a dominant category, then both hard and smooth categories come into play under this weighting scheme.

To rank entities, we propagate and sum the category weights to the output layer. The final entity ranking weight of output node  $e_k$  includes a popularity weight  $P(e_k)$ :

$$ew(e_k) = \left( \sum_{j=1}^n cw(c_j) \right) * P(e_k). \quad (2)$$

The popularity weight is based on the Wikipedia link graph where node  $e_k$  has indegree  $IN_k$ , such that  $P(e_k) = \min(\theta, \log(IN_k))$ ,  $\theta$  being a parameter<sup>5</sup>. Static rank, a well-known concept from Web search, is a query-independent indicator that a certain search result is more likely to be relevant (see, for example, PageRank [11]). We found that connectivity in Wikipedia is an indicator that an entity is well-known, and therefore possibly a good search result.

## 4.2 ReFER Bootstrap and Its Application to ER Systems

The algorithm we propose is query independent as it just needs an initial set of entities where to start from. ER systems start from keyword queries provided by the user in order to generate a ranked list of results. We propose three ways of running our algorithm and combining it with existing ER systems.

In the first scenario the user provides also a small set of example relevant entities. We can use such set as the active nodes  $E_q$  from input layer I. We would thus obtain a ranked list of entities ordered by decreasing  $ew(e_k)$  scores. It is then possible to merge, for example by means of a linear combination, the obtained

<sup>4</sup> Experimentally exploring the parameter space we obtained best results with  $\alpha = 10$  and  $\beta = 50$ .

<sup>5</sup> Experimentally exploring the parameter space we obtained best results with  $\theta = 5$ .

ranking with one produced by an ER system which uses keywords provided by the user. In this paper we perform ranking combination in the following way<sup>6</sup>:

$$\text{rank}(e_k, q) := \lambda \cdot \text{baseline}(e_k, q) + (1 - \lambda) \cdot \text{ReFER}(e_k), \quad (3)$$

where  $\text{rank}(e_k, q)$  is the new rank for entity  $e_k$  on query  $q$ ,  $\lambda \in [0, 1]$ ,  $\text{baseline}(e_k, q)$  is the rank assigned by the baseline system, and  $\text{ReFER}(e_k)$  is the rank assigned to  $e$  based on the scores computed by Formula 2.

A second approach would be to use results of an ER system in order to bootstrap our algorithm (i.e., as elements of the input layer). Thus, in a pseudo-RFB fashion, we consider top-k retrieved entities as being part of  $E_q$ . Again, in this way we would obtain a ranked list of entities by running the ReFER algorithm. We can now combine the two available rankings by, for example, in a linear combination.

A third approach, is the RFB one. After the ER system retrieves results for a query, the user selects relevant results present in top-k. We can use selected relevant results as elements of active input layer  $E_q$ . Again, we can combine the two rankings (the original one and the one generated based on Formula 2) by a linear combination.

## 5 Experimental Results

In this section we present an experimental evaluation of the proposed model for RFB in ER. We start describing the test collection we use and we then evaluate effectiveness of different applications to existing ER baseline systems.

### 5.1 Experimental Setting

The Entity Ranking track at INEX has developed a test collection based on Wikipedia. We perform our experiments on this test collection, for an objective and comparable evaluation. We will consider our RFB approach successful if it improves consistently upon the measured performance for most (or all) of the runs submitted to the track, essentially using the participant runs as baselines. This is an especially challenging goal in case of runs that already use the Wikipedia link structure between entities and/or categories.

The document collection used for evaluating our approach is the 2006 Wikipedia XML Corpus<sup>8</sup> containing 659,338 English Wikipedia articles. In INEX 2008, 35 topics have been selected and manually assessed by the participants<sup>7</sup>. An example of an INEX 2008 Entity Ranking Topic is presented in Table 1. The

<sup>6</sup> A different option would be to combine RSVs of the baseline ER system with  $ew(e_k)$  scores. Due to the variety of approaches that lead to the scores in different ER systems, we could estimate such scores transforming the rank of entity  $e_k$  for query  $q$ ; we carried out experiments computing the rank-based scores as  $(1000 - \text{rank})$  and  $(1/\text{rank})$ . As the conclusions resulting from both transformations turned out identical we perform a simpler combination of ranks.

<sup>7</sup> The test collection we used is available at: <http://www.L3S.de/~demartini/XER08/>.



**Table 1.** INEX Entity Ranking Topic example

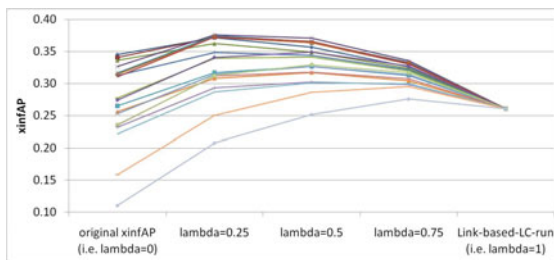
Title	Italian Nobel prize winners
Categories	#924: Nobel laureates
Examples	#176791: Dario Fo; #744909: Renato Dulbecco; #44932: Carlo Rubbia

track distinguishes between the XML Entity Ranking (XER) and the List Completion (LC) tasks. In the XER task, participants use topic category and topic title; in the LC case, the example entities provided in the topics can be used by the system (and should not be presented in the results). Because the assessment pool has been created using stratified sampling, the evaluation metric used is xinfAP [19], an estimation of Average Precision (AP) for pools built with a stratified sampling approach. The ranked list of entities produced by ER systems is divided into disjoint contiguous subsets (strata) and then entities are randomly selected (sampled) from each stratum for relevance judgement. The metric xinfAP is then computed exploiting the estimation of Precision at each relevant document for each stratum.

## 5.2 Using Topic Examples

In order to evaluate the combination of ReFER with previously proposed ER systems, we decided to apply our algorithm to all the submitted runs at INEX 2008 as baselines as well as to the top performing runs of a later method tested on the same collection [2]. We then combine the results with baseline systems following Formula 3.

We performed such experiment with both XER and LC runs. The values of xinfAP for the original runs and the combination with the ReFER run are presented in Figure 2 for the XER task. The Figure shows how in all cases the

**Fig. 2.** Comparison of INEX 2008 XER runs merged with ReFER using topic examples

combination of the baseline with ReFER improves the quality of the original ER system. For the runs where the initial baseline performs well (a high xinfAP), the best average value for lambda is close to 0.25 (i.e., giving more importance to the baseline). Baselines that did not perform that well require a higher  $\lambda$  of 0.75, giving more importance to ReFER results. For both tasks, the value of  $\lambda$  that yields best absolute improvement (i.e. 6.4% for XER and 5.2% for LC) is 0.5, so we present the following experiment results only for this combination strategy.

### 5.3 Content Based Pseudo Relevance Feedback

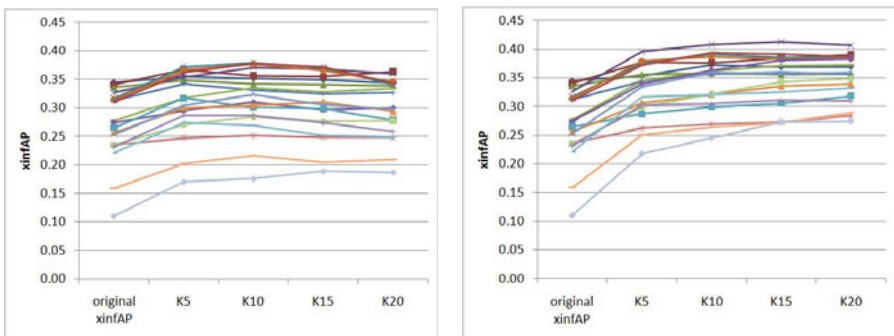
How does the ReFER approach perform as compared to standard content based pseudo-RFB? As we do not have access to the retrieval systems used to create the various runs, we implemented a system independent method. From each run we start from the top  $k$  retrieved results, from which we take top  $n$  common terms. The terms are ranked based on the cumulated TF-IDF score from the  $k$  documents. Next, we search with both the topic title and the top  $n$  common terms in our index of the INEX Wikipedia and retrieve ranked lists of results for each run. We then combine such result set with the corresponding original run by applying Formula 3 with  $\lambda = 0.5$ .

Experimental findings show that this method performed best on average when using top 5 common terms from top 10 retrieved documents. The maximum absolute improvement achieved by the content based approach is of 2% on average. Also, the content based method improved only 79% of the 19 runs.

### 5.4 Pseudo Relevance Feedback

Instead of using the example entities provided in the topic we can use top- $k$  retrieved results from each run. In this way, we build a system that requires no user involvement, but that just builds on top of another method for ER.

For each query  $q$  we activate the  $k$  nodes in the input layer that correspond to the top- $k$  retrieved results from the baseline run. Figure 3(a) shows the xinfAP values for the original runs and for the combination (i.e., Formula 3 with  $\lambda = 0.5$ ) with such pseudo-RFB run, for different values of  $k$ .



**Fig. 3.** Improvement of xinfAP for each run using all (a) or only relevant results (b) in top- $k$  retrieved as seed of the algorithm, combining with  $\lambda = 0.5$

The effectiveness is always improved for each  $k$ . In Table 2 it is possible to see that, on average,  $K = 10$  gives best improvement both for xinfAP and for the expected P@20 (as used in [19]). A t-test shows that the xinfAP improvement using  $k = 10$  and  $\lambda = 0.5$  over each baseline is statistical significant ( $p \leq 0.05$ ) for all systems but one, where  $p = 0.53$ .

**Table 2.** xinfAP and expected P@20 measured for different values of  $k$  for pseudo-RFB and the relative improvement obtained by the combination over the original runs

	xinfAP				expected P@20			
	K=5	K=10	K=15	K=20	K=5	K=10	K=15	K=20
Original	0.270	0.270	0.270	0.270	0.307	0.307	0.307	0.307
pseudo-RFB	0.266	0.275	0.267	0.256	0.284	0.290	0.277	0.269
Combination $\lambda = 0.5$	0.308	0.313	0.307	0.300	0.327	0.328	0.319	0.315
Relative Improvement	14%	16%	14%	11%	7%	7%	4%	3%

**Table 3.** Average unique contribution of relevant results (pseudo-RFB)

	K=5	K=10	K=15	K=20
Relevant in baseline	5.158	4.654	4.557	4.495
Relevant in pseudo-RFB	3.289	3.544	3.555	3.425
Relevant in both	10.694	11.198	11.296	11.358
Missed relevant	4.010	3.754	3.744	3.873

The results show how a small but effective seed leads to good results after applying the score propagation. When analysing the contribution of unique relevant results from the baseline and the pseudo-RFB we can see (Table 3) that most of the relevant results are present in both runs while only 4 relevant entities out of 21, on average, are not retrieved.

## 5.5 Relevance Feedback

In the next scenario we assume entity ranking in an interactive setting where the user can click on the relevant entities in the top- $k$  results returned by the baseline system (i.e., RFB). Because assessing the relevance of entities returned can be considered to take a much lower effort than reading documents in a traditional information retrieval setting, we believe the ER setting justifies measuring the improvement in quality of the full displayed list (as opposed to the rank freezing or residual ranking methodologies that are more appropriate in the ad-hoc retrieval case [14]). When performing an entity retrieval task, the user’s aim is not to read new relevant documents, but rather to obtain a precise and complete list of entities that answers the query. Thus, we use only relevant entities in top- $k$  as

**Table 4.** xinfAP and expected P@20 measured for different values of  $k$  for RFB and the relative improvement obtained by the combination over the original runs

	xinfAP				expected P@20			
	K=5	K=10	K=15	K=20	K=5	K=10	K=15	K=20
Original	0.270	0.270	0.270	0.270	0.307	0.307	0.307	0.307
RFB	0.281	0.310	0.320	0.327	0.295	0.332	0.339	0.347
Combination $\lambda = 0.5$	0.327	0.341	0.347	0.350	0.386	0.382	0.380	0.381
Relative Improvement	21%	26%	29%	30%	26%	24%	24%	24%

**Table 5.** Average unique contribution of relevant results (RFB)

	K=5	K=10	K=15	K=20
Relevant in baseline	7.14	5.78	5.32	4.95
Relevant in RFB	2.02	2.65	2.96	3.11
Relevant in both	8.71	10.07	10.54	10.91
Missed relevant	5.28	4.65	4.34	4.19

seed to our algorithm. For xinfAP, it is possible to see how the algorithm obtains best performances with  $k = 20$  (cf. Table 4).

If we compare Table 2 and Table 4 we can see that in the pseudo-RFB case, the best improvement is obtained using the first 10 retrieved results. In the RFB scenario, given that input entities are all relevant, the higher the value of  $k$ , the better the improvement. We did not study the effect of  $k > 20$  because we do not expect a user to select relevant results lower than rank 20. A t-test confirms statistical significance ( $p \leq 0.05$ ) of the improvement in xinfAP between the run using  $k = 20$  and  $\lambda = 0.5$  and each of the baselines.

If we analyze the contribution of unique relevant results from the baseline and the RFB results (Table 5) we see that the baseline contributes more than the pseudo-RFB part. Compared to the contribution of uniquely relevant entities in the pseudo-RFB scenario (see Table 3), we find however that blind feedback works better with respect to this aspect. This result can be explained by the fact that when considering system-topic pairs in almost 20% of the cases there are no relevant results in top- $k$  retrieved results. There are only 7 topics for which all systems had relevant results in top 5 retrieved results. Thus in the RFB scenario we cannot apply our algorithm for all the system-topic pairs, whereas for pseudo-RFB the algorithm is applied also using only non-relevant entities.

## 5.6 Hard vs. Smooth Categories

What is the benefit of using hard and smooth categories? In order to observe the effect of using smoothed categories along with hard categories we experimented with various sets of categories both in the pseudo-RFB and RFB cases (see Table 6). We used as input nodes top  $k=10$  retrieved results from the baseline (for the RFB case we only used the relevant from top 10 retrieved results, amounting to 3.63 results per topic). In both cases the use of smooth categories improves the

**Table 6.** xinfAP measured for  $k=10$  in the pseudo-RFB and RFB cases. The relative improvement obtained by the combination over the baseline is also shown.

	pseudo-RFB			RFB		
	$C_H$	$C_S$	$C_H \cup C_S$	$C_H$	$C_S$	$C_H \cup C_S$
Baseline	0.270	0.270	0.270	0.270	0.270	0.270
Pseudo-RFB/RFB	0.269	0.126	0.2753	0.306	0.097	0.310
Combination $\lambda = 0.5$	0.308	0.213	0.313	0.338	0.220	0.341
Relative Improvement	14%	-21%	16%	25%	-19%	26%

overall performance of the analyzed systems. Furthermore, in the pseudo-RFB case, where also non-relevant entities are used as seed, the smoothed categories have a higher impact on the overall improvement.

## 6 Conclusions and Further Work

Entity Ranking is a novel search task that goes over document search by finding typed entities in a collection. The retrieved entities can be used, for example, for a better presentation of web search results. In this paper, we presented a model for RFB in the entity retrieval scenario. The proposed model is based on weight propagation in a directed acyclic graph that represents links between entity descriptions. We have used as experimental setting the Wikipedia as a repository of such entity descriptions and have evaluated our approach on the INEX 2008 benchmark.

We have used the submitted runs as baselines and have shown, firstly, that performing fusion with the result of our algorithm using relevant entity examples as initial seed always improves over the baseline effectiveness. We have also evaluated our algorithm using as seed the top- $k$  retrieved results in a pseudo-RFB fashion. The experiments demonstrate that, while in all cases the baselines were improved, using top 10 results yields the best improvement. Finally, we have shown how an emulated interactive feedback session (by using only the relevant entities in the top- $k$  retrieved results) leads to an even higher improvement when performing a fusion with the baseline (i.e., a 0.12 absolute improvement in xinfAP using the relevant entities encountered in top 20).

We conclude that the proposed approach can be easily applied to any ER system in order to improve search effectiveness, and that the model performs well on the test collection we used. A limitation of this work is the use of a single test collection. As future work, we aim at evaluating our approach on a different ER setting such as, for example, graph-based tag recommendation [9].

## References

1. Bailey, P., Craswell, N., De Vries, A., Soboroff, I.: Overview of the TREC 2007 Enterprise Track. In: Proceedings of TREC 2007, Gaithersburg, MD (2008)
2. Balog, K., Bron, M., de Rijke, M.: Category-based query modeling for entity search. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 319–331. Springer, Heidelberg (2010)
3. Bast, H., Chitea, A., Suchanek, F., Weber, I.: Ester: efficient search on text, entities, and relations. In: SIGIR 2007, pp. 671–678. ACM, New York (2007)
4. Cheng, T., Chang, K.: Entity Search Engine: Towards Agile Best-Effort Information Integration over the Web. In: CIDR 2007, pp. 108–113 (2007)
5. Cheng, T., Yan, X., Chang, K.C.-C.: Entityrank: Searching entities directly and holistically. In: Proceedings of VLDB, pp. 387–398 (2007)
6. Demartini, G., Firan, C., Iofciu, T., Krestel, R., Nejdl, W.: Why finding entities in wikipedia is difficult, sometimes. *Information Retrieval* 13(5), 534–567 (2010)

7. Demartini, G., Missen, M.M.S., Blanco, R., Zaragoza, H.: Entity summarization of news articles. In: SIGIR, pp. 795–796 (2010)
8. Denoyer, L., Gallinari, P.: The Wikipedia XML corpus. *ACM SIGIR Forum*. 40(1), 64–69 (2006)
9. Huang, Z., Chung, W., Ong, T., Chen, H.: A graph-based recommender system for digital library. In: Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 65–73. ACM, New York (2002)
10. Kumar, R., Tomkins, A.: A characterization of online search behavior. *IEEE Data Eng. Bull.* (2009)
11. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web (1998)
12. Pehcevski, J., Vercoustre, A.-M., Thom, J.A.: Exploiting locality of wikipedia links in entity ranking. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008*. LNCS, vol. 4956, pp. 258–269. Springer, Heidelberg (2008)
13. Rode, H., Serdyukov, P., Hiemstra, D.: Combining document- and paragraph-based entity ranking. In: SIGIR, pp. 851–852 (2008)
14. Ruthven, I., Lalmas, M.: A Survey on the Use of Relevance Feedback for Information Access Systems. *Knowl. Eng. Rev.* 18(2), 95–145 (2003)
15. Serdyukov, P., Balog, K., Thomas, P., Vries, A., Westerveld, T.: Overview of the TREC 2009 Entity Track (2009)
16. Tsikrika, T., Serdyukov, P., Rode, H., Westerveld, T., Aly, R., Hiemstra, D., de Vries, A.P.: Structured document retrieval, multimedia retrieval, and entity ranking using pF/Tijah. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) *INEX 2007*. LNCS, vol. 4862, pp. 306–320. Springer, Heidelberg (2008)
17. Vallet, D., Zaragoza, H.: Inferring the most important types of a query: a semantic approach. In: SIGIR, pp. 857–858 (2008)
18. de Vries, A.P., Vercoustre, A.-M., Thom, J.A., Craswell, N., Lalmas, M.: Overview of the INEX 2007 entity ranking track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) *INEX 2007*. LNCS, vol. 4862, pp. 245–251. Springer, Heidelberg (2008)
19. Yilmaz, E., Kanoulas, E., Aslam, J.A.: A simple and efficient sampling method for estimating ap and ndcg. In: SIGIR, pp. 603–610 (2008)
20. Zaragoza, H., Rode, H., Mika, P., Atserias, J., Ciaramita, M., Attardi, G.: Ranking very many typed entities on wikipedia. In: CIKM, pp. 1015–1018 (2007)

# The Limits of Retrieval Effectiveness

Ronan Cummins<sup>1,3</sup>, Mounia Lalmas<sup>2</sup>, and Colm O’Riordan<sup>3</sup>

<sup>1</sup> School of Computing Science, University of Glasgow, UK

<sup>2</sup> Yahoo! Research, Barcelona, Spain

<sup>3</sup> Dept. of Information Technology, National University of Ireland, Galway, Ireland

ronan.cummins@nuigalway.ie

**Abstract.** Best match systems in Information Retrieval have long been one of the most predominant models used in both research and practice. It is argued that the effectiveness of these types of systems for the ad hoc task in IR has plateaued. In this short paper, we conduct experiments to find the upper limits of performance of these systems from three different perspectives. Our results on TREC data show that there is much room for improvement in terms of term-weighting and query reformulation in the ad hoc task given an entire information need.

## 1 Background

Best match systems in IR are the predominant model in both research and industry for developing search engines. From library searches to Internet search, these best match systems aim at returning only relevant documents given a user query. It has been stated in the past few years that the performance of ad hoc retrieval has plateaued or even that the performance of IR systems has failed to improve since 1994 [1]. This is one of the reasons that there has been a shift away from more traditional views of IR, to examine, among others, the querying process. The overall aim of our project (ACQUIRE - AutomatiC Query formUlation in Information REtrieval) is to learn how best to extract good queries given an information need from statistical and linguistic features of the terms and queries. However, for this short paper, we focus on finding the upper bound on performance from three different perspectives.

In a typical search scenario, a user who has an information need (*IN*) in mind, formulates this need into a query (*Q*). This is similar to the TREC formulation for the ad hoc task, where the topic *description* and *narrative* are a natural language description of the information need (*IN*) and the *title* is a sample query (*Q*). However, this sample query is only one of a myriad of queries that might be posed for the same information need.

In web search, usually a user poses short queries mainly because they have adapted their own behaviour for use with the system. Ultimately however, a user should to be able to communicate (and search) using an IR system in his/her own natural language and thus, automatic query extraction is an important goal in IR. If web systems provided a much better performance for longer queries, users may adapt their behaviour further. At present, there are many IR domains

in which a user already provides longer type queries. For example, in spoken retrieval, a user may utter a few sentences of an information need [4]. Similarly, in patent search [2], queries are often extracted from a document that has been filed for patent. In this paper, we report on experiments that aim to find the best query (Q) for a given information need (IN). The contribution of this paper is three-fold:

- Firstly, we determine the effectiveness of humans at manually extracting queries from an information need (IN). It is important to understand how good people are at the task of query generation.
- Secondly, we determine the effectiveness of the *best* possible query that might be extracted given an information need (IN).
- Thirdly, we determine the effectiveness of the *best* possible query for each topic (i.e. the universal upper bound of system performance for a topic).

The paper is organised as follows: Section 2 formally outlines the problem of query extraction and draws comparison to that of query term-weighting. We also outline a method for finding near optimal queries given a set of terms. Section 3 presents experiments in three subsections that map to the three different research questions above. Our conclusions are outlined in Section 4.

## 2 Query Generation

In this section, we describe the task of query formulation and outline a method to find near optimal queries given an IN. Research into tasks of query sampling [5], query modification, query re-weighting, query reduction [3] and query extraction, can be thought of in similar ways (usually the query is modelled as a vector of terms, and the weights are modified to change retrieval behaviour).

Given a best match IR system, a user interacts with it by formulating and entering a query for a specific IN<sup>1</sup>. More formally, for a IN of  $N$  terms, there are  $2^N - 1$  possible queries that exist (ignoring the empty query). The example below shows all the possible queries for a three term IN.

$$\left( \begin{array}{c|ccccccc} & Q_1 & Q_2 & Q_3 & Q_4 & Q_5 & Q_6 & Q_7 \\ \hline t_1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ t_2 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ t_3 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array} \right)$$

Given that the number of possible queries that can be created for even a relatively small IN (e.g. 20 terms) is so large, we can suppose that a user creates sub-optimal queries. Essentially, given an IR system, we can reformulate the IR problem into that of finding the best query (Q) for an IN. As already stated, exhaustively finding the optimally performing query is infeasible given even a

<sup>1</sup> This information need may, or may not, be written down but for the purposes of this study we can assume that there is a written description of the IN.



small  $IN$  of 20 or 30 terms for each topic (as we would have to submit  $2^N$  queries to the system). However, by adopting a standard ‘greedy’ approach, we can build near optimal queries. A ‘greedy’ approach finds the best one term query. Then, the approach finds the best two term query, assuming that it includes the best one term query. This approach requires submitting  $N^2$  queries to the system. The query search space may be deceptive, but this approach finds high performing queries (as our results will show) and can be thought of as a conservative upper bound on performance given an  $IN$ .

From the example three term  $IN$  above, it should be noted that the problem of query extraction can also be viewed as a query term-weighting problem. For example, given the  $IN$ , a discrete weighting of 1 or 0 for the query terms (i.e. query term-weights) can be applied to the terms. Equally the problem might be viewed as a binary classification problem, classifying a term in the  $IN$  to be used in the query, or not used in the query. Nevertheless, reformulating the problem in this user centered way, allows us to view the problem as a classic AI search problem and allows us to specify the difficulty of the problem.

Given this user or query focused view of the IR problem, we now address the following questions; (1) How good are users at formulating queries given an  $IN$ ? (2) What is the performance of the best possible query that could be extracted from the  $IN$ ? and (3) What is the performance of the best possible query that could be presented to the system? The experiments that are outlined in the rest of this short paper attempt to answer these questions.

### 3 Experiments

In this section, we compare three approaches to generating queries. For the experiments in this paper we use the FT, AP, WSJ, and FR TREC sub-collections. These collections have different sizes and properties making our results more general. For the  $IN$ , we use the *description* and *narrative*. On average the  $IN$  contains 20 to 35 unique terms. The system we used for all our experiments is an implementation of the *BM25* ranking function<sup>2</sup>.

#### 3.1 Manual Extraction

For the first experiment, we investigate the effectiveness of humans at manually extracting queries from a given  $IN$ . We gave the topic *descriptions* and *narratives* to a number of people in the broad area of IR (i.e. experts) who were asked to manually extract a good query (Q). The *title* was presented to them as a example query. Furthermore, we used the *title* of the information need as another pseudo expert. Table 1 presents the effectiveness of each set of manually extracted queries. Most users performed significantly ( $\downarrow$  denotes 0.05 confidence level for Wilcoxon test) worse than simply entering in the entire  $IN$  (i.e. *description* and *narrative*) into the system. The *Max\_User* label is the performance of

<sup>2</sup> We also ran experiments using a dirichlet prior language model and obtained very similar results.

**Table 1.** MAP for Manual Query Extraction Task

	FT	FR	AP	WSJ
#Docs	210,158	55,630	242,918	130,837
#Topics	188 (251-450)	91 (251-450)	149 (051-200)	150 (051-200)
desc+narr	0.2529	0.2930	0.2098	0.3018
User_1 (title)	0.2281	0.2841	0.1632 ↓	0.2223 ↓
User_2	0.2482	0.2673	0.1773 ↓	0.2496 ↓
User_3	0.2212 ↓	0.2226 ↓	0.1833 ↓	0.2501 ↓
User_4	0.2302 ↓	0.2152 ↓	0.1888 ↓	0.2674 ↓
Avg_User	0.2319	0.2473	0.1782	0.2473
Max_User	0.3173	0.3572	0.2311	0.3163

**Table 2.** Optimal Performance (MAP) for Query Extraction Task

	FT	FR	AP	WSJ
desc+narr	0.2529	0.2930	0.2098	0.3018
Avg_Length	(23)	(24)	(32)	(32)
Avg_User	0.2319	0.2473	0.1782	0.2473
Avg_Length	(3.9)	(3.8)	(4.7)	(4.7)
Opt	0.4832	0.5838	0.3776	0.4712
Avg_Length	(4.5)	(3.85)	(6.4)	(6.3)

the best of the four user generated queries for each topic. From this we can see that users can often choose queries that surpass the effectiveness of the entire *IN*. This experiments tells us that human extracted queries are, on average, less effective than simply entering the entire *IN* into the system<sup>3</sup>, but could surpass the effectiveness of the *IN* in a best case scenario. A repeated-measures ANOVA performed on average precision of the queries across the four users showed no significant difference (on all but the FR collection<sup>4</sup>), telling us that the four users are likely to perform similarly.

### 3.2 Optimal Extraction

In this section, we present the results from the experiment that aims to find a conservative upper bound (i.e. near optimal query) on the performance given an *IN* (i.e. only using terms from the *description* and *narrative*). Table 2 shows the performance of the best query (Opt) using the *greedy* approach outlined in Section 2. It can be seen that if we could extract the best query from the *IN*, we could double the effectiveness (i.e. *MAP*) compared to the average user. This informs us that there is a lot more that might be achieved using the *IN* given. This might be useful in scenarios where a user poses a longer query or in situations where the *IN* is available. Also shown in Table 2 is the average length of the optimal queries found using our approach. We can see that the optimal

<sup>3</sup> We do acknowledge that entering the entire *IN* into the system is an added effort for the user for only a marginal extra benefit.

<sup>4</sup> This difference was not present using the language model as the IR system.

queries are short compared to the entire *IN*. This result has implications for term-weighting schemes for longer type queries. This is because the extraction task can also be viewed as a query term-weighting problem. By taking account of terms already in the query, term dependent term-weighting scheme may be a fruitful avenue of research.

### 3.3 System Limit

In the final experiment of this short paper, we aim to find the upper bound on the performance of the system (i.e. is a *MAP* of 1 possible for a set of topics?). To find the upper bound on performance for individual topics, terms are extracted from the relevant documents. Again we use the same greedy approach outlined in Section 2 to find high performing queries. However, because there is a larger number of terms (i.e. those extracted from relevant documents), we only find optimal queries up to length of 10 terms to illustrate the general trend. Figure 1 shows the performance of the best queries found for each query length for a set of topics. The key labelled “SYSTEM LIMIT” is the conservative upper bound for the system for a set of topics. The other curves (labelled “IN LIMIT”) show the performance of the optimal queries extracted from the *IN* (as per section 3.2). Firstly, we can see that perfect IR performance (i.e. *MAP* of 1) is achievable on one of the collections for a set of topics using queries of only five terms. Although, this collection is the smallest collection, our results would tend to suggest that near perfect retrievability is possible using best match systems. This further enhances the view that we might better improve IR effectiveness by concentrating on modifying the input to these systems.

Figure 1 also outlines the performance of the best query of each length extracted from the *IN*. We can see that the performance peaks at about six terms

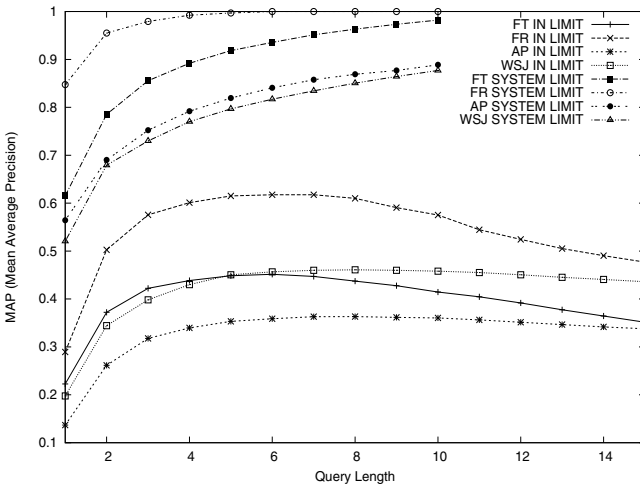


Fig. 1. Upper Limit (MAP) for System

and decreases afterwards. This curve will decrease to the same performance of the entire *IN* once all terms are selected for use in the query. This evidence might help explain why users often simply submit short queries (i.e. short queries can be powerful).

## 4 Conclusion and Future Work

Overall, we have found that although people are good at extracting terms from an *IN*, entering the entire *IN* into a system is better. The average person achieves over 80% performance by entering a few terms compared to typing in the entire *IN*. Interestingly, we have found that the upper bound on the performance for query extraction is more than twice that of the average person, and close to twice the performance of the entire *IN*. Furthermore, we have found that given a fixed number of terms (as a person may formulate for an *IN*), optimal performance is achieved by only entering a small number of those query terms. Typically, given 20-30 terms that describe an information need, the optimal queries lie in the range of three to six terms. Finally, we show that best match systems are very powerful, as if the near perfect query is entered, these systems can achieve near perfect retrieval for small to medium sized databases.

Future work includes applying machine learning algorithms to learn the best query extraction methods given an information need. We plan to release the data and features gathered so that the task of query extraction can become a standard machine learning task for others in the community to research.

**Acknowledgments.** Ronan Cummins is funded by the Irish Research Council (IRCSET), co-funded by Marie Curie Actions under FP7. The authors are grateful to the annotators from Galway and Glasgow who helped in the query extraction process. This paper was written when Mounia Lalmas was a Microsoft Research/RAEng Research Professor at the University of Glasgow.

## References

1. Armstrong, T.G., Moffat, A., Webber, W., Zobel, J.: Improvements that don’t add up: ad-hoc retrieval results since 1998. In: CIKM 2009, pp. 601–610. ACM, New York (2009)
2. Graf, E., Azzopardi, L., van Rijsbergen, K.: Automatically generating queries for prior art search. In: Peters, C., Caputo, B., Gonzalo, J., Jones, G.J.F., Kalpathy-Cramer, J., Müller, H., Tsirikla, T. (eds.) CLEF 2009. LNCS, vol. 6242, pp. 480–490. Springer, Heidelberg (2010)
3. Kumaran, G., Carvalho, V.R.: Reducing long queries using query quality predictors. In: SIGIR, pp. 564–571 (2009)
4. Popescu-Belis, A., Kilgour, J., Poller, P., Nanchen, A., Boertjes, E., de Wit, J.: Automatic content linking: speech-based just-in-time retrieval for multimedia archives. In: SIGIR 2010, p. 703. ACM, New York (2010)
5. Yang, L., Wang, L., Geng, B., Hua, X.-S.: Query sampling for ranking learning in web search. In: SIGIR 2009, pp. 754–755. ACM, New York (2009)

# Learning Conditional Random Fields from Unaligned Data for Natural Language Understanding

Deyu Zhou<sup>1</sup> and Yulan He<sup>2</sup>

<sup>1</sup> School of Computer Science and Engineering  
Southeast University, China

<sup>2</sup> Knowledge Media Institute, Open University  
Milton Keynes MK7 6AA, UK  
d.zhou@seu.edu.cn, y.he@open.ac.uk

**Abstract.** In this paper, we propose a learning approach to train conditional random fields from unaligned data for natural language understanding where input to model learning are sentences paired with predicate formulae (or abstract semantic annotations) without word-level annotations. The learning approach resembles the expectation maximization algorithm. It has two advantages, one is that only abstract annotations are needed instead of fully word-level annotations, and the other is that the proposed learning framework can be easily extended for training other discriminative models, such as support vector machines, from abstract annotations. The proposed approach has been tested on the DARPA Communicator Data. Experimental results show that it outperforms the hidden vector state (HVS) model, a modified hidden Markov model also trained on abstract annotations. Furthermore, the proposed method has been compared with two other approaches, one is the hybrid framework (HF) combining the HVS model and the support vector hidden Markov model, and the other is discriminative training of the HVS model (DT). The proposed approach gives a relative error reduction rate of 18.7% and 8.3% in F-measure when compared with HF and DT respectively.

## 1 Introduction

One of the key tasks in natural language understanding is semantic parsing which maps natural language sentences to complete formal meaning representations. For example, the following sentence could be represented by the predicate formula (also called abstract semantic annotation) as shown below:

I want to return to Dallas on Thursday.

RETURN ( TOLOC ( CITY ( Dallas ) ) ON ( DATE ( Thursday ) ) )

Early approaches to semantic parsing rely on hand-crafted semantic grammar rules to fill slots in semantic frames using word pattern and semantic tokens. Such rule-based approaches are typically domain-specific and often fragile. In contrast, statistical approaches are able to accommodate the variations found in real data and hence can in principle be more robust. They can be categorized into three types: generative approaches, discriminative approaches and a hybrid of the two.

Generative approaches learn the joint probability model,  $P(W, C)$ , of input sentence  $W$  and its semantic tag sequence  $C$ , compute  $P(C|W)$  using the Bayes rule, and then

take the most probable semantic tag sequence  $C$ . The hidden Markov model (HMM), being a generative model, has been predominantly used in statistical semantic parsing. It models sequential dependencies by treating a semantic parse sequence as a Markov chain, which leads to an efficient dynamic programming formulation for inference and learning. The hidden vector state (HVS) model [4] is a discrete HMM model in which each HMM state represents the state of a push-down automaton with a finite stack size. State transitions are factored into separate stack pop and push operations constrained to give a tractable search space. The result is a model which is complex enough to capture hierarchical structure but which can be trained automatically from only lightly annotated data. Discriminative approaches directly model posterior probability  $P(C|W)$  and learn mappings from  $W$  to  $C$ . Conditional random fields (CRFs), as one representative example, define a conditional probability distribution over label sequence given an observation sequence, rather than a joint distribution over both label and observation sequences [5]. Another example is the hidden Markov support vector machines (HM-SVMs) [1] which combine the flexibility of kernel methods with the idea of HMMs to predict a label sequence given an input sequence. However, such discriminative methods require fully annotated corpora for training which are difficult to obtain in practical applications. On the other hand, the HVS model can be easily trained from only lightly annotated corpora. However, unlike discriminative models such as the CRFs, it cannot include a large number of correlated lexical or syntactic features in input sentences. It is thus interesting to explore the feasibility to train CRFs from abstract semantic annotations. It is a highly challenge task since the derivation from each sentence to its abstract semantic annotation is not annotated in the training data and is considered hidden.

In this paper, we propose a learning approach based on expectation maximization (EM) to train the CRFs from abstract annotations. This approach works as follows, the CRFs compute expectation based on initial parameters in first step. Based on the expectation results, the CRFs are then constrainedly trained using some general learning algorithms such as stochastic gradient descent (SGD). With re-estimated parameters, the CRFs go to the next iteration until no more improvements could be achieved. Our proposed learning approach has two advantages, one is that the CRFs can be trained from abstract semantic annotations without expensive treebank style annotation data, and the other is that the learning approach is applicable to other discriminative models such as SVMs. To evaluate the performance of the proposed approach, we conducted experiments on the DARPA Communicator Data. Experimental results show that our proposed approach outperforms the HVS model trained also on abstract annotations. Furthermore, the proposed approach outperforms the other two approaches, one is the hybrid framework (HF) combining HVS and HM-SVMs, and the other is discriminative training of the HVS model (DT). The proposed approach gives a relative error reduction rate of 18.7% and 8.3% in F-measure when compared with HF and DT respectively.

The rest of this paper is organized as follows. Section 2 introduces CRFs and the parameter estimation and inference procedures of training CRFs. Our proposed learning procedure to train CRFs from abstract annotations is presented in Section 3. Experimental setup and results are discussed in Section 4. Finally, Section 5 concludes the paper.

## 2 Conditional Random Fields

Linear-chain conditional random fields (CRFs), as a discriminative probabilistic model over sequences of feature vectors and label sequences, have been widely used to model sequential data. This model is analogous to maximum entropy models for structured outputs. By making a first-order Markov assumption on states, a linear-chain CRF defines a distribution over state sequence  $\mathbf{y} = y_1, y_2, \dots, y_T$  given an input sequence  $\mathbf{x} = x_1, x_2, \dots, x_T$  ( $T$  is the length of the sequence) as

$$p(\mathbf{y}|\mathbf{x}) = \frac{\prod_t \Phi_t(y_{t-1}, y_t, \mathbf{x})}{Z(\mathbf{x})} \quad (1)$$

where the partition function  $Z(\mathbf{x})$  is the normalization constant that makes the probability of all state sequences sum to one.

### 2.1 Inference

The most probable labeling sequence can be calculated by  $\operatorname{argmax}_Y P(Y|X; \Theta)$ . It can be efficiently calculated using the Viterbi algorithm. Similar to the forward-backward procedure for HMM, the marginal probability of states at each position in the sequence can be computed as,

$$P(y_t = s|\mathbf{x}) = \frac{\alpha_t(y_t = s|\mathbf{x})\beta_t(y_t = s|\mathbf{x})}{Z(\mathbf{x})} \quad (2)$$

where  $Z(\mathbf{x}) = \sum_y \alpha_t(y|\mathbf{x})$ .

The forward values  $\alpha_t(y_t = s|\mathbf{x})$  and backward values  $\beta_t(y_t = s|\mathbf{x})$  are defined in iterative form as follows,

$$\alpha_t(y_t = s|\mathbf{x}) = \sum_{y'} \alpha_{t-1}(y_{t-1} = y'|\mathbf{x}) \exp \sum_k \theta_k f_k(y_{t-1} = y', y_t = s, \mathbf{x}) \quad (3)$$

$$\beta_t(y_t = s|\mathbf{x}) = \sum_{y'} \beta_{t+1}(y_{t+1} = y'|\mathbf{x}) \exp \sum_k \theta_k f_k(y_{t+1} = y', y_t = s, \mathbf{x}) \quad (4)$$

## 3 Training CRFs from Abstract Annotations

To train CRFs from abstract annotations, the expectation maximization (EM) algorithm can be extended to efficiently estimate model parameters. The EM algorithm is an efficient iterative procedure to compute the maximum likelihood (ML) estimate in the presence of missing or hidden data [3]. The EM algorithm is divided into two-step iterations: The E-step, and the M-step. The missing data are estimated given the observed data and current estimate of the model parameters in E-step. In the M-step, the likelihood function is maximized under the assumption that the missing data are known. We now explain how to train CRFs from abstract annotations.

Given a sentence labeled with an abstract semantic annotation as shown in Table II, we first expand the annotation to the flattened semantic tag sequence as in Table II(a). In order to cater for irrelevant input words, a DUMMY tag is allowed everywhere in

**Table 1.** Abstract semantic annotation

Sentence:	I want to return to Dallas on Thursday.
Abstract annotation:	RETURN (TOLOC (CITY (Dallas) ) ON (DATE (Thursday) ) )
(a) Flattened semantic tag list:	RETURN RETURN+TOLOC RETURN+TOLOC+CITY (Dallas) RETURN+ON RETURN+ON+DATE (Thursday)
(b) Expanded semantic tag list:	RETURN RETURN+DUMMY RETURN+TOLOC RETURN+TOLOC+DUMMY RETURN+TOLOC+CITY (Dallas) RETURN+TOLOC+CITY (Dallas) +DUMMY RETURN+ON RETURN+ON+DUMMY RETURN+ON+DATE (Thursday) RETURN+ON+DATE (Thursday) +DUMMY

preterminal positions. Hence, the flattened semantic tag sequence is finally expanded to the semantic tag sequence as in Table 1(b).

We first calculate the log likelihood of  $L(\theta)$  with expectation over the abstract annotation as follows,

$$\begin{aligned}
 L(\theta; \theta^t) &= \sum_i^M \sum_{Y_i^u} P(Y_i^u | X_i; \theta^t) \log P(Y_i^u | X_i; \theta) \\
 &= \sum_i^M \sum_{Y_i^u} P(Y_i^u | X_i; \theta^t) \sum_t \sum_k \theta_k f_k(y', y, X_i) - \sum_i^k \log Z(X_i),
 \end{aligned}$$

where  $Y_i^u$  is the unknown semantic tag sequence of the  $i$ -th word sequence, and

$$Z(X_i) = \sum_y \exp\left(\sum_t \sum_k \theta_k f_k(y_{t-1}, y_t, X_i)\right) \tag{5}$$

It can be optimized using the same optimization method as in standard CRFs training.

Then, to infer the word-level semantic tag sequences based on abstract annotations, Equations 3 and 4 are modified as shown in Equations 6 and 7.

$$\alpha_t(y_t = s | \mathbf{x}) = \begin{cases} 0, & \text{when } g(s, x_t) = 1 \\ \sum_{y'} \left\{ \alpha_{t-1}(y_{t-1} = y' | \mathbf{x}) \right. \\ \left. \exp \sum_k \theta_k f_k(y_{t-1} = y', y_t = s, \mathbf{x}) \right\}, & \text{otherwise} \end{cases} \tag{6}$$

$$\beta_t(y_t = s | \mathbf{x}) = \begin{cases} 0, & \text{when } g(s, x_t) = 1 \\ \sum_{y'} \left\{ \beta_{t+1}(y_{t+1} = y' | \mathbf{x}) \right. \\ \left. \exp \sum_k \theta_k f_k(y_{t+1} = y', y_t = s, \mathbf{x}) \right\}, & \text{otherwise} \end{cases} \tag{7}$$

where  $g(s, x_t)$  is defined as follows,

$$g(s, x_t) = \max \begin{cases} 1, s \text{ is not in the allowable semantic tag list of } \mathbf{x} \\ 1, s \text{ is not of class type and } x_t \text{ is of class type} \\ 0, \text{ otherwise} \end{cases} \tag{8}$$

$g(s, x_t)$  in fact encodes the two constraints implied from abstract annotations. Firstly, state transitions are only allowed if both incoming and outgoing states are listed in the semantic annotation defined for the sentence. Secondly, if there is a lexical item attached to a preterminal tag of a flattened semantic tag, that semantic tag must appear bound to that lexical item in the training annotation.



## 4 Experiments

Experiments have been conducted on the DARPA Communicator data [2] which are available for public download. The data contain utterance transcriptions and the semantic parse results from the rule-based Phoenix parser. After cleaning up the data, the training set consist of 12702 utterances while the test set contains 1178 utterances. The abstract annotation used for training and the reference annotation needed for testing were derived by hand correcting the Phoenix parse results. For example, for the sentence “Show me flights from Boston to New York”, the abstract annotation would be `FLIGHT (FROMLOC (CITY) TOLOC (CITY) )`. Such an annotation need only list a set of valid semantic concepts and the dominance relationships between them without considering the actual realized concept sequence or attempting to identify explicit word/concept pairs. Thus, it avoids the need for expensive tree-bank style annotations.

In all the subsequent experiments, the proposed learning approach is implemented by modifying the source code of the CRF suite<sup>1</sup>. The features such as word features (current word, previous word, next word and so on) and POS features (current POS tag, previous one, next one and so on) are employed. To estimate the parameters of CRFs, the stochastic gradient descent (SGD) iterative algorithm [6] was employed. As discussed in Section 1 that while CRFs can easily incorporate arbitrary features into training, HVS model cannot include a large number of correlated lexical or syntactic features in input sentences. It would be interested to see how CRFs compared to HVS when both are trained from abstract annotations. The proposed CRFs learning approach achieved 92.37% of F-measure, which significantly outperforms HVS. Employing SGD gives a relative error reduction of 36.6%, when compared with the performance of the HVS model where only 87.97% was achieved.

**Table 2.** Performance comparison between the proposed approach and the two other approaches

Measurement	Performance			Relative Error Reduction	
	HF	DT	Our Approach	Compared with HF	Compared with DT
Recall (%)	90.99	91.47	92.27	14.2	9.4
Precision (%)	90.25	91.87	92.48	22.9	7.5
F-measure (%)	90.62	91.68	92.37	18.7	8.3

We further compare our proposed learning approach with two other methods. One is a hybrid generative/discriminative framework (HF) [7] which combines HVS with HM-SVMs so as to allow the incorporation of arbitrary features as in CRFs. The same features as listed above were used in HF training. The other is a discriminative approach (DT) based on parse error measure to train the HVS model [8]. The generalized probabilistic descent (GPD) algorithm was employed for adjusting the HVS model to achieve minimum parse error rate. Table 2 shows that our proposed learning approach outperforms both HF and DT. Training CRFs on abstract annotations allows the calculation of conditional likelihood and hence results in direct optimization of the objective function to reduce the error rate of semantic labeling. In the contrary, the hybrid framework

<sup>1</sup> <http://www.chokkan.org/software/crfsuite/>

firstly uses the HVS parser to generate full annotations for training HM-SVMs. This process involves the optimization of two different object functions (one for HVS and another for HM-SVMs). Although DT also uses an objective function which aims to reduce the semantic parsing error rate. It is in fact employed for supervised re-ranking where the input is the  $N$ -best parse results generated from the HVS model.

## 5 Conclusions and Future Work

In this paper, we proposed an effective learning approach which can train the conditional random fields without the expensive treebank style annotation data. Instead, it trains the CRFs from only abstract annotations in a constrained way. Experimental results show that 36.6% relative error reduction in F-measure was obtained using the proposed approach on the DARPA Communicator Data when compared with the performance of the HVS model. Furthermore, the proposed learning approach also outperforms two other methods, one is the hybrid framework (HF) combining both HVS and HM-SVMs, and the other is discriminative training (DT) of the HVS model, with a relative error reduction rate of 18.7% and 8.3% being achieved when compared with HF and DT respectively.

## References

1. Altun, Y., Tsochantaridis, I., Hofmann, T.: Hidden markov support vector machines. In: International Conference in Machine Learning, pp. 3–10 (2003)
2. CUData. Darpa communicator travel data. university of colorado at boulder (2004), <http://communicator.colorado.edu/phoenix>
3. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B 39(1), 1–38 (1977)
4. He, Y., Young, S.: Semantic processing using the hidden vector state model. Computer Speech and Language 19(1), 85–106 (2005)
5. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML 2001: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289. Morgan Kaufmann Publishers Inc., San Francisco (2001)
6. Shai Shalev-Shwartz, Y.S., Srebro, N.: Pegasos: Primal estimated sub-gradient solver for svm. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007), pp. 807–814 (2007)
7. Zhou, D., He, Y.: A Hybrid Generative/Discriminative Framework to Train a Semantic Parser from an Un-annotated Corpus. In: Proceedings of 22nd International Conference on Computational Linguistics (COLING 2008), Manchester, UK, pp. 1113–1120 (2008)
8. Zhou, D., He, Y.: Discriminative Training of the Hidden Vector State Model for Semantic Parsing. IEEE Transaction on Knowledge and Data Engineering 21(1), 66–77 (2008)

# Subspace Tracking for Latent Semantic Analysis

Radim Řehůřek

NLP lab, Masaryk University in Brno  
radimrehurek@seznam.cz

**Abstract.** Modern applications of Latent Semantic Analysis (LSA) must deal with enormous (often practically infinite) data collections, calling for a single-pass matrix decomposition algorithm that operates in constant memory w.r.t. the collection size. This paper introduces a *streamed distributed algorithm for incremental SVD updates*. Apart from the theoretical derivation, we present experiments measuring numerical accuracy and runtime performance of the algorithm over several data collections, one of which is the whole of the English Wikipedia.

## 1 Introduction

The purpose of *Latent Semantic Analysis (LSA)* is to find hidden (*latent*) structure in a collection of texts represented in the Vector Space Model [10]. LSA was introduced in [4] and has since become a standard tool in the field of Natural Language Processing and Information Retrieval. At the heart of LSA lies the *Singular Value Decomposition (SVD)* algorithm, which makes LSA (sometimes also called Latent Semantic Indexing, or LSI) really just another member of the broad family of applications that make use of SVD's robust and mathematically well-founded approximation capabilities [1]. In this way, although we will discuss our results in the perspective and terminology of LSA and Natural Language Processing, our results are in fact applicable to a wide range of problems and domains across much of the field of Computer Science.

In this paper, we will be dealing with the practical issues of computing SVD efficiently in a distributed manner. For a more gentle introduction to SVD and LSA, its history, pros and cons and comparisons to other methods, see elsewhere [4,5,7].

### 1.1 SVD Characteristics

In terms of practical ways of computing SVD, there is an enormous volume of literature [12,3,5,14]. The algorithms are well-studied and enjoy favourable numerical properties and stability, even in the face of badly conditioned input. They differ in their focus on what role SVD performs—batch algorithms vs. online updates, optimizing FLOPS vs. number of passes, accuracy vs. speed etc.

---

<sup>1</sup> Another member of that family is the *discrete Karhunen–Loève Transform*, from Image Processing; or Signal Processing, where SVD is commonly used to separate signal from noise. SVD is also used in solving shift-invariant differential equations, in Geophysics, in Antenna Array Processing, . . .

**Table 1.** Selected SVD algorithms for truncated (partial) factorization and their characteristics. In the table, “—” stands for *no/not found*. See text for details.

Algorithm	Distribu- table	Incremental docs	Incremental terms	Matrix structure	Subspace tracking	Implementations
Krylov subspace methods (Lanczos)	yes	—	—	sparse	—	<a href="#">PROPACK</a> , <a href="#">ARPACK</a> , <a href="#">SVDPACK</a> , <a href="#">MAHOUT</a>
Halko et al. <a href="#">[8]</a>	yes	—	—	sparse	—	<a href="#">redsvd</a> , our own
Gorrell & Webb <a href="#">[6]</a>	—	—	—	sparse	—	<a href="#">LingPipe</a> , our own
Zha & Simon <a href="#">[14]</a>	—	yes	yes	dense	yes	—, our own
Levy & Lindenbaum <a href="#">[9]</a>	—	yes	—	dense	yes	—, our own
Brand <a href="#">[2]</a>	—	yes	yes	dense	—	—, our own
this paper	yes	yes	—	sparse	yes	<a href="#">our own</a> , <a href="#">open-sourced</a>

In Table [1](#), we enumerate several such interesting characteristics, and evaluate them for a selected set of known algorithms.

**Distributable.** Can the algorithm run in a distributed manner (without major modifications or further research)? Here, we only consider distribution of a very coarse type, where each computing node works autonomously. This type of parallelization is suitable for clusters of commodity computers connected via standard, high-latency networks, as opposed to specialized hardware or supercomputers.

**Incremental Updates.** Is the algorithm capable of updating its decomposition as new data arrives, without recomputing everything from scratch? The new data may take form of new observations (documents), or new features (variables). Note that this changes the shape of the  $A$  matrix. With LSA, we are more interested in whether we can efficiently add new documents, rather than new features. The reason is that vocabulary drift (adding new words; old words acquiring new meanings) is a relatively slow phenomena in natural languages, while new documents appear all the time.

**Matrix Structure.** Does the algorithm make use of the structure of the input matrix? In particular, does the algorithm benefit from sparse input? Algorithms that can be expressed in terms of *Basic Linear Algebra Subprograms (BLAS)* routines over the input matrix are relatively easily adapted to any type of input structure.

**Subspace Tracking.** In online streaming environments, new observations come in asynchronously and the algorithm cannot in general store all the input documents in memory (not even out-of-core memory). The incoming observations must be immediately processed and then discarded<sup>2</sup>.

Being online has implication on the decomposition itself, because we cannot even afford to keep the truncated right singular vectors  $V$  in memory. The size of  $V_{n \times m}$  is  $O(n)$ , linear in the number of input documents, which is prohibitive. Therefore, only the  $U, S$  matrices are retained and the decomposition is used as a *predictive* (rather than descriptive) model. We call the

<sup>2</sup> This is in contrast to offline, *batch* algorithms, where the whole dataset is presented at once and the algorithm is allowed to go back and forth over the dataset many times.

$P_{m \times m} \equiv S^{-1}U^T$  matrix the *projection matrix*, and the projection process  $V_x = P \cdot x$  is called *folding-in* in the context of LSA<sup>3</sup>.

In such subspace tracking scenario, the input data stream can be assumed to be non-stationary. This allows us to introduce an explicit factor for “forgetting” old observations and adjusting the decomposition in favour of new data. This is realized by introducing a parameter  $\gamma \in \langle 0.0, 1.0 \rangle$ , called the *decay factor*, which dictates the rate of discounting the relevancy of old observations.

**Available Implementations.** While secondary from a theoretical point of view, we consider the availability of a real, executable reference implementation critical for a method’s adoption. The application of LSA is relevant to a wider audience who simply do not possess the time or motivation to disentangle terse mathematical equations into functional programs. We also observe that most of the existing SVD implementations (with the notable exceptions of the [Apache MAHOUT project](#) and [LingPipe](#)) are written in somewhat opaque, FORTRANish style of coding, even when implemented in other languages.

## 2 Distributed LSA

In this section, we derive a novel algorithm for distributed online computing of LSA over a cluster of computers, in a single pass over the input matrix.

### 2.1 Overview

Distribution will be achieved by column-partitioning the input matrix  $A$  into several smaller submatrices, called *jobs*,  $A^{m \times n} = [A_1^{m \times c_1}, A_2^{m \times c_2}, \dots, A_j^{m \times c_j}]$ ,  $\sum c_i = n$ . Since columns of  $A$  correspond to documents, each job  $A_i$  amounts to processing a chunk of  $c_i$  input documents. The sizes of these chunks are chosen to fit available resources of the processing nodes: bigger chunks mean faster overall processing but on the other hand consume more memory.

Jobs are then distributed among the available cluster nodes, in no particular order, so that each node will be processing a different set of column-blocks from  $A$ . The nodes need not process the same number of jobs, nor process jobs at the same speed; the computations are completely asynchronous and independent. Once all jobs have been processed, the decompositions accumulated in each node will be merged into a single, final decomposition  $P = (U, S)$  (see section [1.1](#) on subspace tracking for where  $V^T$  disappeared). As a reminder,  $U$  and  $S$  are respectively an orthonormal and a diagonal matrix such that  $A = USV^T$ , or equivalently and perhaps more naturally for avoiding mentioning the unused

<sup>3</sup> Note that folding-in is different to updating the decomposition: during folding-in, the  $U, S$  matrices stay intact and an existing model is only used to predict positions of documents in the latent space. In particular,  $V_{m \times n}^T = S^{-1}U^T A = P_{m \times m} A_{m \times n}$ , so that even though we cannot store the right singular vectors  $V^T$  during computations, they can still be recovered in a streaming fashion if needed, provided one has access to the projection matrix  $P$  and the original collection  $A$ .

$V^T$ , such that  $AA^T = US^2U^T$ . The former factorization is called the Singular Value Decomposition, the latter is its related eigen decomposition.

What is needed are thus two algorithms:

- 1. Base decomposition.** In main memory, find  $P_i = (U_i^{m \times c_i}, S_i^{c_i \times c_i})$  eigen decomposition of a single job  $A_i^{m \times c_i}$  such that  $A_i A_i^T = U_i S_i^2 U_i^T$ .
- 2. Merge decompositions.** Merge  $P_i = (U_i, S_i), P_j = (U_j, S_j)$  of two jobs  $A_i, A_j$  into a single decomposition  $P = (U, S)$  such that  $[A_i, A_j] [A_i, A_j]^T = US^2U^T$ .

We would like to highlight the fact that the first algorithm will perform decomposition of a *sparse* input matrix, while the second algorithm will merge two *dense* decompositions into another dense decomposition. This is in contrast to incremental updates discussed in the literature [2,9,14], where the existing decomposition and the new documents are mashed together into a single matrix, losing any potential benefits of sparsity as well as severely limiting the possible size of a job. The explicit merge procedure also makes the distributed version of the algorithm straightforward, so that the computation can be split across a cluster of computers. Another volume of literature on efficient SVD concerns itself with Lanczos-based iterative solvers (see e.g. [11] for a large-scale batch approach). These are not applicable to the streaming scenario, as they require a large number of  $O(k)$  passes over the input and are not incremental.

## 2.2 Solving the Base Case

There exist a multitude of partial sparse SVD solvers that work in-core. We view the particular implementation as “black-box” and note that the Lanczos-based implementations mentioned in Table 1 are particularly suitable for this task.

## 2.3 Merging Decompositions

No efficient algorithm (as far as we know) exists for merging two truncated eigen decompositions (or SVD decompositions) into one. We therefore propose our own, novel algorithm here, starting with its derivation and summing up the final version in the end.

The problem can be stated as follows. Given two truncated eigen decompositions  $P_1 = (U_1^{m \times k_1}, S_1^{k_1 \times k_1}), P_2 = (U_2^{m \times k_2}, S_2^{k_2 \times k_2})$ , which come from the (by now lost and unavailable) input matrices  $A_1^{m \times c_1}, A_2^{m \times c_2}, k_1 \leq c_1$  and  $k_2 \leq c_2$ , find  $P = (U, S)$  that is the eigen decomposition of  $[A_1, A_2]$ .

Our first approximation will be the direct naive

$$U, S^2 \stackrel{\text{eigen}}{\leftarrow} [U_1 S_1, U_2 S_2] [U_1 S_1, U_2 S_2]^T. \quad (1)$$

This is terribly inefficient, and forming the matrix product of size  $m \times m$  on the right hand side is prohibitively expensive. Writing  $SVD_k$  for truncated SVD that returns only the  $k$  greatest singular numbers and their associated singular vectors, we can equivalently write

$$U, S, V^T \stackrel{SVD_k}{\leftarrow} [\gamma U_1 S_1, U_2 S_2]. \quad (2)$$

This is more reasonable, with the added bonus of increased numerical accuracy over the related eigen decomposition. Note, however, that the computed right singular vectors  $V^T$  are not needed at all, which is a sign of further inefficiency. Also, the fact that  $U_1, U_2$  are orthonormal is completely ignored. This leads us to break the algorithm into several steps:

---

**Algorithm 1.** Baseline merge

---

- Input:** Truncation factor  $k$ , decay factor  $\gamma$ ,  $P_1 = (U_1^{m \times k_1}, S_1^{k_1 \times k_1})$ ,  $P_2 = (U_2^{m \times k_2}, S_2^{k_2 \times k_2})$
  - Output:**  $P = (U^{m \times k}, S^{k \times k})$
  - 1  $Q, R \leftarrow^{QR} [\gamma U_1 S_1, U_2 S_2]$
  - 2  $U_R, S, V_R^T \leftarrow^{SVD_k} R$
  - 3  $U^{m \times k} \leftarrow Q^{m \times (k_1 + k_2)} U_R^{(k_1 + k_2) \times k}$
- 

On line 1, an orthonormal subspace basis  $Q$  is found which spans both of the subspaces defined by columns of  $U_1$  and  $U_2$ ,  $\text{span}(Q) = \text{span}([U_1, U_2])$ . Multiplications by  $S_1, S_2$  and  $\gamma$  provide scaling for  $R$  only and do not affect  $Q$  in any way, as  $Q$  will always be column-orthonormal. Our algorithm of choice for constructing the new basis is QR factorization, because we can use its other product, the upper trapezoidal matrix  $R$ , to our advantage. Now we are almost ready to declare  $(Q, R)$  our target decomposition  $(U, S)$ , except  $R$  is not diagonal. To diagonalize the small matrix  $R$ , we perform an SVD on it, on line 2. This gives us the singular values  $S$  we need as well as the rotation of  $Q$  necessary to represent the basis in this new subspace. The rotation is applied on line 3. Finally, both output matrices are truncated to the requested rank  $k$ . The costs are  $O(m(k_1 + k_2)^2)$ ,  $O((k_1 + k_2)^3)$  and  $O(m(k_1 + k_2)^2)$  for line 1, 2 and 3 respectively, for a combined total of  $O(m(k_1 + k_2)^2)$ .

Although more elegant than the direct decomposition given by Equation 2, the baseline algorithm is only marginally more efficient than the direct SVD. This comes as no surprise, as the two algorithms are quite similar and SVD of rectangular matrices is often internally implemented by means of QR in exactly this way. Luckily, we can do better.

First, we observe that the QR decomposition makes no use of the fact that  $U_1$  and  $U_2$  are already orthogonal. Capitalizing on this will allow us to represent  $U$  as an update to the existing basis  $U_1$ ,  $U = [U_1, U']$ , dropping the complexity of the first step to  $O(mk_2^2)$ . Secondly, the application of rotation  $U_R$  to  $U$  can be rewritten as  $UU_R = [U_1, U'] U_R = U_1 R_1 + U' R_2$ , dropping the complexity of the last step to  $O(mkk_1 + mkk_2)$ . Plus, the algorithm can be made to work by modifying the existing matrices  $U_1, U_2$  in place inside BLAS routines, which is a considerable practical improvement over Algorithm 1, which requires allocating additional  $m(k_1 + k_2)$  floats.

The first two lines construct the orthonormal basis  $U'$  for the component of  $U_2$  that is orthogonal to  $U_1$ ;  $\text{span}(U') = \text{span}((I - U_1 U_1^T)U_2) = \text{span}(U_2 - U_1(U_1^T U_2))$ .

As before, we use QR factorization because the upper trapezoidal matrix  $R$  will come in handy when determining the singular vectors  $S$ .

---

**Algorithm 2.** Optimized merge

---

**Input:** Truncation factor  $k$ , decay factor  $\gamma$ ,  $P_1 = (U_1^{m \times k_1}, S_1^{k_1 \times k_1})$ ,  $P_2 = (U_2^{m \times k_2}, S_2^{k_2 \times k_2})$   
**Output:**  $P = (U^{m \times k}, S^{k \times k})$

- 1  $Z^{k_1 \times k_2} \leftarrow U_1^T U_2$
- 2  $U', R \xleftarrow{QR} U_2 - U_1 Z$
- 3  $U_R, S, V_R^T \xleftarrow{SVDk} \begin{bmatrix} \gamma S_1 & Z S_2 \\ 0 & R S_2 \end{bmatrix}^{(k_1+k_2) \times (k_1+k_2)}$
- 4  $\begin{bmatrix} R_1^{k_1 \times k} \\ R_2^{k_2 \times k} \end{bmatrix} = U_R$
- 5  $U \leftarrow U_1 R_1 + U' R_2$

---

Line 3 is perhaps the least obvious, but follows from the requirement that the updated basis  $[U, U']$  must satisfy

$$[U_1 S_1, U_2 S_2] = [U_1, U'] X, \tag{3}$$

so that

$$X = [U_1, U']^T [U_1 S_1, U_2 S_2] = \begin{bmatrix} U_1^T U_1 S_1 & U_1^T U_2 S_2 \\ U'^T U_1 & U'^T U_2 S_2 \end{bmatrix}. \tag{4}$$

Using the equalities  $R = U'^T U_2$ ,  $U'^T U_1 = 0$  and  $U_1^T U_1 = I$  (all by construction) we obtain

$$X = \begin{bmatrix} S_1 & U_1^T U_2 S_2 \\ 0 & U'^T U_2 S_2 \end{bmatrix} = \begin{bmatrix} S_1 & Z S_2 \\ 0 & R S_2 \end{bmatrix}. \tag{5}$$

Line 4 is just a way of saying that on line 5,  $U_1$  will be multiplied by the first  $k_1$  rows of  $U_R$ , while  $U'$  will be multiplied by the remaining  $k_2$  rows. Finally, line 5 seeks to avoid realizing the full  $[U_1, U']$  matrix in memory and is a direct application of the equality

$$[U_1, U']^{m \times (k_1+k_2)} U_R^{(k_1+k_2) \times k} = U_1 R_1 + U' R_2. \tag{6}$$

As for complexity of this algorithm, it is again dominated by the matrix products and the dense QR factorization, but this time only of a matrix of size  $m \times k_2$ . The SVD of line 3 is a negligible  $O(k_1 + k_2)^3$ , and the final basis rotation comes up to  $O(mk \max(k_1, k_2))$ . Overall, with  $k_1 \approx k_2 \approx k$ , this is an  $O(mk^2)$  algorithm.

In Section 3, we will compare the runtime speed of both these proposed merge algorithms on real corpora.

### 2.4 Effects of Truncation

While the equations above are exact when using matrices of full rank, it is not at all clear how to justify truncating all intermediate matrices to rank  $k$  in each update. What effect does this have on the merged decomposition? How do these effects stack up as we perform several updates in succession?

In [15], the authors did the hard work and identified the conditions under which operating with truncated matrices produces exact results. Moreover, they show by way of perturbation analysis that the results are stable (though no longer



exact) even if the input matrix only approximately satisfies this condition. They show that matrices coming from natural language corpora do indeed possess the necessary structure and that in this case, a rank- $k$  approximation of  $A$  can be expressed as a combination of rank- $k$  approximations of its submatrices without a serious loss of precision.

### 3 Experiments

In this section, we will describe two sets of experiments. The first set concerns itself with numerical accuracy of the proposed algorithm, the second with its performance.

In all experiments, the decay factor  $\gamma$  is set to 1.0, that is, there is no discounting in favour of new observation. The number of requested factors is  $k = 200$  for the small and medium corpus and  $k = 400$  for the large Wikipedia corpus.

#### 3.1 Algorithms

We will be comparing four implementations for partial Singular Value Decomposition:

**SVDLIBC.** A direct sparse SVD implementation due to Douglas Rohde<sup>4</sup>.

SVDLIBC is based on the SVDPACK package by Michael Berry [1]. We use its LAS2 routine to retrieve only the  $k$  dominant singular triplets.

**ZMS.** implementation of the incremental one-pass algorithm from [13]. All the operations involved can be expressed in terms of Basic Linear Algebra Subroutines (BLAS). For this reason we use the NumPy library, which makes use of whatever LAPACK library is installed in the system, to take advantage of fast blocked routines. The right singular vectors and their updates are completely ignored so that our implementation of their algorithm also realizes subspace tracking.

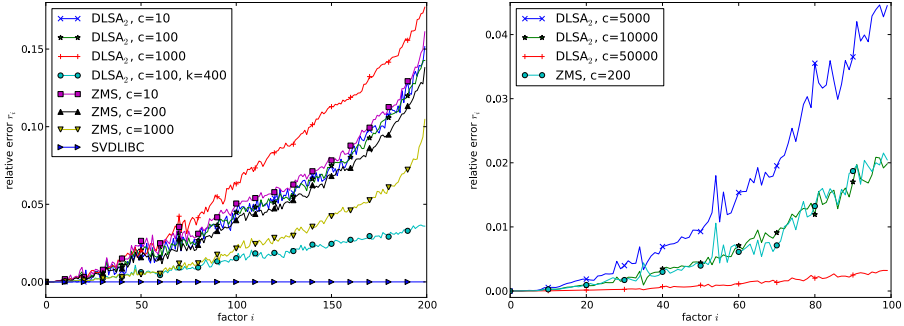
**DLSA.** Our proposed method. We will be evaluating two different versions of the merging routine, Algorithms 1 and 2, calling them  $DLSA_1$  and  $DLSA_2$  in the tables. We will also observe effects of varying the job sizes  $c$  and the number of cluster nodes  $p$  (see Section 2). Again, NumPy is used for dense matrix operations. The base case decomposition is realized by an adapted LAS2 routine from SVDLIBC.

**HEBB.** Streamed stochastic Hebbian algorithm from [6] which loops over the input dataset, in  $k * epochs$  passes, to converge at the singular triplets. The straightforward implementation suffered from serious convergency issues that we couldn't easily fix, so we only include it in our comparisons for the smallest dataset. This algorithm internally updates the singular triplets with explicit array loops (no BLAS).

#### 3.2 Datasets

For the experiments, we will be using three datasets.

<sup>4</sup> <http://tedlab.mit.edu/~dr/SVDLIBC/>



**Fig. 1.** Accuracy of singular values for various decomposition algorithms on the small corpus (left) and medium corpus (right)

**Medium size corpus.** A corpus of 61,293 mathematical articles collected from the digital libraries of [NUMDAM](#), [arXiv](#) and [DML-CZ](#). Together these comprise about 270 million corpus positions, with over 6 million unique word types (we parse out mathematical equations and use them as separate word types). After the standard procedure of pruning out word types that are too infrequent (hapax legomena, typos, OCR errors, etc.) or too frequent (stop words), we are left with 315,002 distinct features. The final matrix  $A^{315,002 \times 61,293}$  has 33.8 million non-zero entries, with density less than 0.18%. This corpus was chosen so that it fits into core memory of a single computer and its decomposition can therefore be computed directly. This will allow us to establish the “ground-truth” decomposition and set an upper bound on achievable accuracy and speed.

**Small corpus.** A subset of 3,494 documents from the medium size corpus. It contains 39,022 features and the sparse  $A^{39,022 \times 3,494}$  matrix has 1,446,235 non-zero entries, so that it is about 23 times smaller than the medium size corpus.

**Large corpus.** The last corpus is the English Wikipedia<sup>5</sup>. This corpus contains 3.2 million documents covering almost 8 million distinct word types. We clip the vocabulary size to the 100,000 most frequent word types, after discarding all words that appear in more than 10% of the documents (“stop-list”). This leaves us with a sparse term-documents matrix with 0.5G non-zero entries, or 14 times the size of the medium corpus.

### 3.3 Accuracy

Figure 1 plots the relative accuracy of singular values found by DLSA, ZMS, SVDLIBC and HEBB algorithms compared to known, “ground-truth” values  $S_G$ . We measure accuracy of the computed singular values  $\bar{S}$  as  $r_i = |\bar{s}_i - s_{G_i}|/s_{G_i}$ , for  $i = 1, \dots, k$ . The ground-truth singular values  $S_G$  are computed directly with

<sup>5</sup> The latest static dump as downloaded from <http://download.wikimedia.org/enwiki/latest>, June 2010.

**Table 2.** Decomposition accuracy on the small corpus, measured by RMSE of document similarities based on ground truth vs. given algorithm

Algorithm	Job size	RMSE	Algorithm	Job size	RMSE
SVDLIBC	3,494	0.0	DLSA <sub>2</sub>	10	0.0204
ZMS	10	0.0204	DLSA <sub>2</sub>	100	0.0199
ZMS	200	0.0193	DLSA <sub>2</sub>	1,000	0.0163
ZMS	1,000	0.0162	DLSA <sub>2</sub>	100, $k = 400$	0.0094

LAPACK’s DGESVD routine for the small corpus and with SVDLIBC’s LAS2 routine for the medium corpus.

We observe that the largest singular values are practically always exact, and accuracy quickly degrades towards the end of the returned spectrum. This leads us to the following refinement: When requesting  $x$  factors, compute the truncated updates for  $k > x$ , such as  $k = 2x$ , and discard the extra  $x - k$  factors only when the final projection is actually needed. This approach is marked as “DLSA<sub>2</sub>,  $c=100, k=400$ ” in Figure 3.3 and then used as default in the larger experiments on the medium corpus. The error is then below 5%, which is comparable to the ZMS algorithm (while DLSA is at least an order of magnitude faster, even without any parallelization).

Even with this refinement, the error is not negligible, so we are naturally interested in how it translates into error of the whole LSA application. This way of testing has the desirable effect that errors in decomposition which do not manifest themselves in the subsequent application do not affect our evaluation, while decomposition errors that carry over to the application are still correctly detected.

To this end, we conducted another set of accuracy experiments. In Latent Semantic Analysis, the most common application is measuring cosine similarity between documents represented in the new, “latent semantic” space. We will compute inter-document similarity of the entire input corpus, forming an  $n \times n$  matrix  $C$ , where each entry  $c_{i,j} = \text{cossim}(\overline{doc}_i, \overline{doc}_j)$ . We do the same thing for the corpus represented by the ground truth decomposition, obtaining another  $n \times n$  matrix. Difference between these two  $n \times n$  matrices (measured by Root Mean Square Error, or RMSE) then gives us a practical estimate of the error introduced by the given decomposition<sup>6</sup>. Note that in addition to testing the magnitude of singular values, this also tests accuracy of the singular vectors at the same time. In Table 2, we can observe that the error of DLSA<sub>2</sub> is around 2%, which is usually acceptable for assessment of document similarity and for document ranking.

### 3.4 Performance

Performance was measured as wall-clock time on a cluster of four dual-core 2GHz Intel Xeons, each with 4GB of RAM, which share the same Ethernet segment

<sup>6</sup> This is a round-about sort of test—to see how accurate a decomposition is, we use it to solve a superordinate task (similarity of documents), then compare results of this superordinate task against a known result.

**Table 3.** Performance of selected partial decomposition algorithms on the small corpus,  $A^{39,022 \times 3,494}$ . Times are in seconds, on the serial setup.

(a) Serial algorithms			(b) DLSA variants		
Algorithm	Job size	Time taken	Job size	DLSA <sub>2</sub>	DLSA <sub>1</sub>
HEBB	N/A	> 1h	10	190	2,406
SVDLIBC	3,494	16	100	122	350
ZMS	10	346	1,000	38	66
ZMS	100	165	3,494	21	21
ZMS	200	150			
ZMS	500	166			
ZMS	1,000	194			
ZMS	2,000	out of memory			

**Table 4.** Performance of selected partial decomposition algorithms on the medium corpus,  $A^{315,002 \times 61,293}$ . Times are in minutes.

(a) Serial algorithms, serial setup.			(b) distributed DLSA <sub>2</sub>				
Algorithm	Job size $c$	Time taken [min]	Job size $c$	No. of nodes $p$			
				serial	1	2	4
SVDLIBC	61,293	9.2					
ZMS	200	360.1	1,000	55.5	283.9	176.2	114.4
			4,000	21.8	94.5	49.6	38.2
			16,000	15.5	29.5	32.0	23.0

and communicate via TCP/IP. The machines were not dedicated but their load was reasonably low during the course of our experiments. To make sure, we ran each experiment three times and report the best achieved time. These machines did not have any optimized BLAS library installed, so we also ran the same experiments on a “cluster” of one node, a dual-core 2.53GHz MacBook Pro with 4GB RAM and *vecLib*, a fast BLAS/LAPACK library provided by the vendor. This HW setup is marked as “serial” in the result tables, to differentiate it from the otherwise equivalent 1-node setup coming from our BLAS-less four-node cluster.

Table 3 summarizes performance results for the small corpus. For ZMS, the update time is proportional to *number of updates* · *cost of update*  $\approx \lceil \frac{n}{c} \rceil \cdot m(k + c)^2$ , so that the minimum (fastest execution) is attained by setting job size  $c = k$ . Overall, we can see that the direct in-core SVDLIBC decomposition is the fastest. The HEBB implementation was forcefully terminated after one hour, with some estimated eight hours left to complete. The speed of DLSA<sub>2</sub> approaches the speed of SVDLIBC as the job size increases; in fact, once the job size reaches the size of the whole corpus, it becomes equivalent to SVDLIBC. However, unlike SVDLIBC, DLSA can proceed in document chunks smaller than the whole corpus, so that corpora that do not fit in RAM can be processed, and so that different document chunks can be processed on different nodes in parallel.

For experiments on the medium corpus, we only included algorithms that ran reasonably fast during our accuracy assessment on the small corpus, that is, only ZMS, DLSA in its fastest variant *DLSA<sub>2</sub>* and SVDLIBC.

Performance of running the computation in distributed mode is summarized in Table 4(b). As expected, performance scales nearly linearly, the only overhead being sending and receiving jobs over the network and the final merges at the very end of the distributed algorithm. This overhead is only significant when dealing with a slow network and/or with extremely few updates. The faster the connecting network and the greater the number of updates,  $\lceil \frac{n}{c} \rceil \gg p$ , the more linear this algorithm becomes.

Another interesting observation comes from comparing  $DLSA_2$  speed in the “serial setup” (fast BLAS) vs. “cluster setup” (no fast BLAS) in Table 4(b). The serial setup is about five times faster than the corresponding cluster version with  $p = 1$ , so that even spreading the computation over four BLAS-less nodes results in *slower* execution than on the single “serial” node. As installing a fast, threaded BLAS library<sup>7</sup> is certainly cheaper than buying five times as many computers to get comparable performance, we strongly recommend doing the former (or both).

**English Wikipedia results.** Since the Wikipedia corpus is too large to fit in RAM, we only ran the streamed  $ZMS$  and  $DLSA_2$  algorithms, asking for 400 factors in each case. On the “serial setup” described above,  $ZMS$  took 109 hours,  $DLSA_2$  8.5 hours. We’d like to stress that these figures are achieved using a single commodity laptop, with a one-pass online algorithm on a corpus of 3.2 million documents, without any subsampling. In distributed mode with six nodes, the time of  $DLSA_2$  drops to 2 hours 23 minutes<sup>8</sup>.

## 4 Conclusion

In this paper we developed and presented a novel distributed single-pass eigen decomposition method, which runs in constant memory w.r.t. the number of observations. This method is suited for processing extremely large (possibly infinite) sparse matrices that arrive as a stream of observations, where each observation must be immediately processed and then discarded. The method is embarrassingly parallel, so we also evaluated its distributed version.

We applied this algorithm to the application of Latent Semantic Analysis, where observations correspond to documents and the number of observed variables (word types) is in the hundreds of thousands. The implementation achieves excellent runtime performance in serial mode (i.e., running on a single computer) and scales linearly with the size of the computer cluster [16]. The implementation is released as open source, under the OSI-approved LGPL licence, and can be downloaded from <http://nlp.fi.muni.cz/projekty/gensim/>.

## Acknowledgements

This study was partially supported by the LC536 grant of MŠMT ČR.

<sup>7</sup> Options include vendor specific libraries (e.g. Intel’s MKL, Apple’s vecLib, Sun’s Sunperf), [ATLAS](#), [GotoBLAS](#), . . .

<sup>8</sup> For details and reproducibility instructions, see the [gensim package documentation](#).

## References

1. Berry, M.: Large-scale sparse singular value computations. *The International Journal of Supercomputer Applications* 6(1), 13–49 (1992)
2. Brand, M.: Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications* 415(1), 20–30 (2006)
3. Comon, P., Golub, G.: Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE* 78(8), 1327–1343 (1990)
4. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science* 41(6), 391–407 (1990)
5. Golub, G., Van Loan, C.: *Matrix computations*. Johns Hopkins University Press, Baltimore (1996)
6. Gorrell, G., Webb, B.: Generalized hebbian algorithm for incremental Latent Semantic Analysis. In: *Ninth European Conference on Speech Communication and Technology* (2005)
7. Griffiths, T., Steyvers, M., Tenenbaum, J.: Topics in semantic representation. *Psychological Review* 114(2), 211–244 (2007)
8. Halko, N., Martinsson, P., Tropp, J.: Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. arXiv 909 (2009)
9. Levy, A., Lindenbaum, M.: Sequential Karhunen–Loeve basis extraction and its application to images. *IEEE Transactions on Image Processing* 9(8), 1371 (2000)
10. Salton, G.: *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley Longman Publishing Co., Inc., Boston (1989)
11. Vigna, S.: Distributed, large-scale latent semantic analysis by index interpolation. In: *Proceedings of the 3rd International Conference on Scalable Information Systems*, pp. 1–10. ICST (2008)
12. William, H., Teukolsky, S., Vetterling, W.: *Numerical Recipes in C: The art of scientific computing*. Cambridge University Press, New York (1988)
13. Zha, H., Marques, O., Simon, H.: Large-scale SVD and subspace-based methods for Information Retrieval. *Solving Irregularly Structured Problems in Parallel*, 29–42 (1998)
14. Zha, H., Simon, H.: On updating problems in Latent Semantic Indexing. *SIAM Journal on Scientific Computing* 21, 782 (1999)
15. Zha, H., Zhang, Z.: Matrices with low-rank-plus-shift structure: Partial SVD and Latent Semantic Indexing. *SIAM Journal on Matrix Analysis and Applications* 21, 522 (2000)
16. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: *Proceedings of LREC 2010 Workshop on New Challenges for NLP Frameworks*, pp. 45–50 (2010)

# Text Retrieval Methods for Item Ranking in Collaborative Filtering

Alejandro Bellogín<sup>1</sup>, Jun Wang<sup>2</sup>, and Pablo Castells<sup>1</sup>

<sup>1</sup> Universidad Autónoma de Madrid,  
Escuela Politécnica Superior,  
Francisco Tomás y Valiente, 11, 28049 Madrid, Spain  
{alejandro.bellogin,pablo.castells}@uam.es

<sup>2</sup> Department of Computer Science,  
University College London,  
Male Place, London, WC1E 6BT, UK  
jun.wang@cs.ucl.ac.uk

**Abstract.** Collaborative Filtering (CF) aims at predicting unknown ratings of a user from other similar users. The uniqueness of the problem has made its formulation distinctive to other information retrieval problems. While the formulation has proved to be effective in rating prediction tasks, it has limited the potential connections between these algorithms and Information Retrieval (IR) models. In this paper we propose a common notational framework for IR and rating-based CF, as well as a technique to provide CF data with a particular structure, in order to be able to use any IR weighting function with it. We argue that the flexibility of our approach may yield to much better performing algorithms. In fact, in this work we have found that IR models perform well in item ranking tasks, along with different normalization strategies.

**Keywords:** Collaborative Filtering, Text Retrieval, Unified Models.

## 1 Introduction

Recommender Systems (RS) suggest *interesting* items to users by taking into account users' profiles. Interestingly, although from the beginning IR techniques have been used in RS and their underlying goals are essentially equivalent [2], no exact equivalences have been established between the models and structures used in IR and those in RS. In particular, we are interested in Collaborative Filtering (CF), one of the most extended types of RS. Specifically, in this work we aim at answering the following research questions: (RQ1) is it possible to use IR models in the rating-based CF framework? and, in that case (RQ2) are IR formulations of CF better or worse than classic CF algorithms? We believe these questions are important because they would allow a better understanding of CF strategies. Furthermore, IR researchers could design better and sound recommender systems using their knowledge on IR and a proper mapping between CF data and IR structures.

**Table 1.** Query and document weighting components for different retrieval methods

Method	$w_t^q$	$w_t^d$
Binary	1 if $t \in q$	1 if $t \in d$
TF dot	qf( $t$ )	tf( $t, d$ )
TF-IDF	qf( $t$ )	tf( $t, d$ ) $\log\left(\frac{N}{df(t)}\right)$
BM25	$\frac{(k_3+1) \text{qf}(t)}{k_3 + \text{qf}(t)}$	$\log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5}\right) \frac{(k_1 + 1) \text{tf}(t, d)}{k_1 (1 - b) + b \cdot dl(d)/dl} + \text{tf}(t, d)$

Our main contribution is a common notational framework for IR and rating-based CF, which provides a general retrieval function adopting any text retrieval weighting function with CF preference data. We also evaluate how well IR methods perform against standard techniques when applied to item ranking, that is, returning a ranking for each user. We have found that IR methods perform particularly well in this task (which is actually equivalent to the classic ad-hoc IR task), whereas different normalization strategies also provide good results.

## 2 A New Collaborative Filtering Framework

Recommender Systems have usually been seen as an IR technique applied when no explicit query has been provided, but a user profile is known instead. However, these two areas have been developed independently. Recently, some works have started to seek explicit links between one area and the other [3, 4, 9]. Rating-based CF is not yet fully integrated with IR, mainly because the input data and the final goal are different: in IR we have query and documents represented by terms, while in rating-based CF we have a set of ratings, which are used to infer the preferences of users towards items, where how to represent the users or the items is unclear. In the next sections, we define a general framework for these two areas, presenting a novel formulation for CF. Based on this framework, we propose a unification of the rating-based CF framework with a text-oriented IR framework, thus enabling the application of classic IR models to a CF system. With this formulation, we can also study different normalisation techniques, beyond those historically used in classic strategies for CF.

### 2.1 A General Text Retrieval Framework

Representing documents and queries as vectors in a common space is known as the vector space model (VSM) [8] and is fundamental to different IR operations [6]. Documents are represented in the vocabulary space using the bag of words approach, in which each document can be described based on the words occurrence frequency (captured by tf or term frequency), so  $\mathbf{d}^i = [\text{tf}(t_1, d^i), \dots, \text{tf}(t_T, d^i)]$  is the vector representation for document  $d^i$ , while  $\mathbf{q} = [\text{qf}(t_1), \dots, \text{qf}(t_T)]$  is the representation for a query  $q$ . Since many plausible weighting functions can be used to represent the importance of a term in a document (query), we keep our formulation general by representing this as follows:  $w(\mathbf{d}^i) = [w_1^i, \dots, w_T^i]$ , and



$w(\mathbf{q}) = [w_1^q, \dots, w_T^q]$ , where  $T$  is the number of terms in the collection and  $w_j^i$  is the weight of term  $t_j$  in document  $d^i$  (or in the query, for  $w_j^q$ ).

On top of the representation model, the core function in any retrieval system is the scoring of a document for a particular query. In the VSM, since queries and documents live in the same space we may use the dot product to obtain a score between a query and a document. We can generalise and formalise this as follows:

$$s(q, d) = \sum_{t \in g(q)} s(q, d, t) \quad (1)$$

where the function  $g(\cdot)$  returns the term components of a query, and  $s(q, d, t)$  is a function of a term, the query and the document. If  $s(q, d, t) = w_t^q \cdot w_t^d$  we get the dot product, which is a simpler but fairly generic formulation (it receives the name of *factored form* in [7]). In Table 1 we can see several examples of different weighting functions; further functions could also be represented this way, such as the ones in language models (as proposed in [7]).

## 2.2 A General Collaborative Filtering Framework

Rating-based CF algorithms deal directly with the set of ratings given by the user community to a set of items. Let us represent as  $r_i^u$  the rating assigned to item  $i$  by user  $u$ , where  $r_i^u \in \{1, \dots, R, \perp\}$ , assuming a rating scale from 1 to  $R$ , the symbol  $\perp$  representing an unknown rating value. The main goal in a rating-based CF system is to predict the user rating for an unknown item, that is, to find the most accurate prediction  $\hat{r}_i^u$  [1]. This can be formulated in a fairly general way by the following equation:

$$\hat{r}_i^u = \sum_{e \in h(u)} f(u, i, e) \quad (2)$$

where the functions  $f$  and  $h$  depend on the CF strategy (user- or item-based). More specifically, in the item-based approach [1] we have  $h(u) = I_u$ , the subset of items rated by user  $u$ , and

$$f(u, i, e) = \frac{\text{sim}(i, e)}{\sum_{e \in h(u)} |\text{sim}(i, e)|} r_e^u \quad (3)$$

A popular similarity function  $\text{sim}(i, e)$  is the Pearson correlation [5]. We can find an analogy between Eq. 2 and 1 simply by equating users to queries and items to documents. In fact, similarly to what we did in the previous section, we may also split up function  $f$  in two parts: one depending exclusively on the user ( $f_e^u$ ) and another on the target item ( $f_e^i$ ), in such a way that  $f(u, i, e) = f_e^u \cdot f_e^i$ , obtaining a dot product too. In Table 2 we show the different possible values for these components according to the CF strategy (user- or item-based). For the user-based approach, the similarity function  $\text{sim}(i, j)$  between two items can also be calculated, for instance, using Pearson correlation, and function  $h(u) = N[u]$  is the set of neighbours for user  $u$ .

**Table 2.** User and item components for function  $f$  in user- and item-based CF.  $E$  represents the space where  $e$  belongs, that is,  $e \in E$ .

Approach	$f_e^u$	$f_e^i$	$E$	$w_e^u$	$w_e^i$
User-based	$\frac{\text{sim}(u,e)}{\sum_{e \in N[u]}  \text{sim}(u,e) }$	$r_i^e$	users	$\text{sim}(u,e)$	$r_i^e$
Item-based	$r_e^u$	$\frac{\text{sim}(i,e)}{\sum_{e \in I_u}  \text{sim}(i,e) }$	items	$r_e^u$	$\text{sim}(i,e)$

### 2.3 Unifying Text Retrieval and Collaborative Filtering

In the last two sections we have presented two general scoring frameworks for text retrieval and rating-based CF under a unified formulation. Now we will explicitly derive the relation between these two frameworks, that is, we define how the IR models listed in Table 1 can be used in the framework defined in Section 2.2. The simplest way to do this is to define what  $\text{tf}$  and  $\text{qf}$  mean in the CF space and then apply the formulas for  $w(\mathbf{q})$  and  $w(\mathbf{d}^i)$  shown in Section 2.1.

It can be shown that taking  $\text{qf}(t) = w_e^u$  and  $\text{tf}(t, d) = w_e^i$  as defined in Table 2, we can obtain equivalent scoring functions to those defined by standard CF algorithms (such as Eq. 3), specifically, when applying the TF model. This implies a positive answer for **RQ1**, since it is possible to make an equivalence between a rating-based item-based CF system and an IR system by means of identifying the terms with the items in the collection, the frequency of a term in the query with the rating assigned by a user to that item, and the frequency of a term in a document with the similarity between the two items involved. Equivalence with respect to user-based CF can be found analogously.

Now, we may rephrase the second research question **RQ2** we address in this paper as: can other IR models, different from basic TF, obtain better results than standard CF techniques (which are equivalent to the TF model)? In the next section, we answer this question.

## 3 Empirical Evaluation

Rating-based CF is generally used for predicting an unknown rating, and the methods are consequently evaluated using error metrics. In contrast, since our approach applies to item ranking task, it needs to be evaluated based on precision metrics. This is more similar to how IR algorithms are usually evaluated.

Our experiments have been carried out using two different datasets from Movielens<sup>1</sup>: *Movielens 100K* and *Movielens 1M*. In our evaluation, we performed a 5-fold cross validation where each split takes 80% of the data for training, and the rest for testing. The item ranking task was performed by predicting a rating score for all the items contained in the test set for the current user.

We evaluated our approach in the user- and item-based versions, although due to space constraints we only show here results for the item-based algorithms. We used four different normalisation techniques and two norms:  $L_1$  and  $L_2$ . These

<sup>1</sup> [www.grouplens.org/node/73](http://www.grouplens.org/node/73)

**Table 3.** Results for different normalisation functions in the item ranking task for item-based (*Movielens 1M*). Standard CF algorithm is underlined, † represents the best method for each normalisation method, ‡ represents the best performing method. Typical values are used for constants ( $k_1 = 1.2, k_3 = 8$ ).

(a) Normalization $s_{00}$				(b) Normalization $s_{10}$			
Method	nDCG	MAP	P@10	Method	nDCG	MAP	P@10
BM25	0.10	0.00	0.00	BM25 $L_1$	0.23†	0.02†	0.00
TF-IDF	0.15	0.01	0.01	TF-IDF $L_1$	0.10	0.01	0.00
TF	0.19†	0.01	0.01	TF $L_1$	0.07	0.00	0.00
				BM25 $L_2$	0.16	0.01	0.00
				TF-IDF $L_2$	0.13	0.01	0.01†
				TF $L_2$	0.16	0.01	0.00

(c) Normalization $s_{01}$				(d) Normalization $s_{11}$			
Method	nDCG	MAP	P@10	Method	nDCG	MAP	P@10
BM25 $L_1$	0.09	0.01	0.00	BM25 $L_1$	0.21	0.02	0.02†
TF-IDF $L_1$	0.05	0.00	0.00	TF-IDF $L_1$	0.01	0.00	0.00
<u>TF <math>L_1</math></u>	0.05	0.00	0.00	TF $L_1$	0.00	0.00	0.00
BM25 $L_2$	0.24	0.03	0.03	BM25 $L_2$	0.06	0.00	0.00
TF-IDF $L_2$	0.29	0.05	0.07	TF-IDF $L_2$	0.05	0.00	0.00
TF $L_2$	0.34‡	0.07‡	0.10‡	TF $L_2$	0.27†	0.03†	0.00

techniques are denoted as  $s_{QD}$ , where  $Q$  and  $D$  are 0 or 1, depending on which vectors (query or document) are used for normalization. For example, Eq. 3 is the same as  $s_{01}$  when the  $L_1$  norm is used.

In Table 3 we present results from the item ranking task in the *Movielens 1M* dataset. For the experiments, we assume vectors  $w(\mathbf{q})$  and  $w(\mathbf{d})$  are defined as in Section 2.1, where values  $w_j^q$  and  $w_j^d$  are given by any retrieval method we want to use (see Table II). Besides that, the interpretation of qf and tf is taken as in Section 2.3. We can see that for each normalization strategy there is at least one method outperforming the baseline. Moreover, BM25 obtains very good results in some of these situations, while the TF method performs better with the  $L_2$  norm. Note that neither of these two methods matches the standard one used in the CF literature. We have obtained very similar results with the *Movielens 100K* dataset, including the best performing algorithms in each situation.

From these experiments, we can see that there can be better IR-based formulations than classic ones used in CF, providing a positive answer to RQ2 in those cases. Besides, since other norms, as well as different weighting functions, appear naturally in our framework, they have also been tested with good results.

## 4 Conclusion and Future Work

We have proposed a generalised model for ranking items in rating-based CF which can fit many different algorithms, including different normalisation

techniques and weighting functions commonly used in IR models. An analogy between IR and rating-based CF has been found: in item-based CF, terms are seen as the items, while the term frequencies are the user ratings (in the query representation) or the item similarity (in the document representation). As a result, it is possible to directly apply IR models in our framework with comparable or better results than classic CF formulations. Besides that, different normalisation techniques can fit into our framework and also lead to good results; furthermore, as in IR, different frequency normalisation techniques can also be used, which in CF can be translated into z-scores [5] instead of ratings, for example. These techniques have not been studied in this work, but are envisioned as future work. We also plan to extend our study to further IR models such as language models or probabilistic models.

Finally, although our results are consistent across two different datasets, we plan to test our framework on further publicly available datasets, such as Netflix or *Movielens 10M*. Besides, comparison with other state-of-the-art techniques should be performed, such as SVD, for a more detailed answer to **RQ2**.

**Acknowledgments.** This work was supported by the Spanish Ministry of Science and Innovation (TIN2008-06566-C04-02), and the Regional Government of Madrid (S2009TIC-1542).

## References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE* 17(6), 734–749 (2005)
2. Belkin, N.J., Croft, W.B.: Information filtering and information retrieval: two sides of the same coin? *Commun. ACM* 35(12), 29–38 (1992)
3. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: 14th Conference on UAI, pp. 43–52 (1998)
4. Cöster, R., Svensson, M.: Inverted file search algorithms for collaborative filtering. In: 25th ACM SIGIR Conference, pp. 246–252. ACM, New York (2002)
5. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: 22nd ACM SIGIR Conference, pp. 230–237. ACM, New York (1999)
6. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*, 1st edn. Cambridge University Press, Cambridge (2008)
7. Metzler, D., Zaragoza, H.: Semi-parametric and non-parametric term weighting for information retrieval. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) *ICTIR 2009*. LNCS, vol. 5766, pp. 42–53. Springer, Heidelberg (2009)
8. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. *Commun. ACM* 18(11), 613–620 (1975)
9. Wang, J., Robertson, S., de Vries, A., Reinders, M.: Probabilistic relevance ranking for collaborative filtering. *Information Retrieval* 11(6), 477–497 (2008)

# Classifying with Co-stems

## A New Representation for Information Filtering

Nedim Lipka and Benno Stein

Bauhaus-Universität Weimar, 99421 Weimar, Germany  
{nedim.lipka,benno.stein}@uni-weimar.de

**Abstract.** Besides the content the writing style is an important discriminator in information filtering tasks. Ideally, the solution of a filtering task employs a text representation that models both kinds of characteristics. In this respect word stems are clearly content capturing, whereas word suffixes qualify as writing style indicators. Though the latter feature type is used for part of speech tagging, it has not yet been employed for information filtering in general. We propose a text representation that combines both the output of a stemming algorithm (stems) and the stem-reduced words (co-stems). A co-stem can be a prefix, an infix, a suffix, or a concatenation of prefixes, infixes, or suffixes. Using accepted standard corpora, we analyze the discriminative power of this representation for a broad range of information filtering tasks to provide new insights into the adequacy and task-specificity of text representation models. Altogether we observe that co-stem-based representations outperform the classical bag of words model for several filtering tasks.

## 1 Introduction

Identifying relevant, interesting, high quality, or humorous texts in wikis, emails, and blogs is the tedious job of every searcher. Algorithmic information filtering [4] simplifies this process by finding those texts in a stream or a collection that fulfill a given information interest. Current information filtering technology mostly relies on text classification where the classes describe the information interests. Usually the text representations are content-based, although various filtering tasks are characterized by their intricate combination of content and style.

In this paper we evaluate whether the untapped potential of a style representation in fact is substantial. We propose a model that encodes both (1) text content and (2) text style in the form of word stems and word co-stems respectively. To draw a clear and comprehensive picture of the underlying effects and their importance we resort to a straightforward vector representation. We consider the computational simplicity of this representation as a useful contribution, and to the best of our knowledge the co-stem representation has not been proposed or investigated in this respect. Also the number and heterogeneity of information filtering tasks that are compared in this paper goes beyond existing evaluations. In particular, we analyze the tasks in Table 1 that are marked with ticks (✓) and refer to the relevant literature. In this table,  $d$  denotes a plain text document, extracted from an email, a wiki page, a blog entry, or a web page, depending on the task in question.

**Table 1.** Overview of information filtering tasks. Those tasks which are analyzed in this paper are tagged with the ✓-symbol.

Task	Description	Reference
Age Group Detection	Determine the age of the author who wrote <i>d</i> .	[17]
Authorship Attribution	Determine the author of <i>d</i> , given a set of authors.	[18] ✓
Authorship Verification	Determine if <i>d</i> is written by more than one author.	[5]
Gender Detection	Determine the gender of the author who wrote <i>d</i> .	[17] ✓
Genre Analysis	Determine the genre of <i>d</i> , given a set of genres.	[19] ✓
Information Quality Assessment	Determine whether <i>d</i> is of high quality.	[8] ✓
Language Identification	Determine the language of <i>d</i> .	[3]
Sarcasm Detection	Determine whether <i>d</i> is sarcastic.	[20]
Sentiment Analysis	Determine the sentiment expressed in <i>d</i> .	[13] ✓
Spam Detection (email, web page)	Determine whether <i>d</i> is spam or non-spam.	[11][10] ✓
Topic Detection	Determine the topic of <i>d</i> .	[7] ✓
Vandalism Detection	Determine whether <i>d</i> is vandalized.	[15]

## 2 Co-stems

This section introduces the construction of co-stems as the following operation: given a word its stem is computed first, and then the residuals of the word without its stem are taken as co-stems. For example, consider the words “timeless” and “timelessly” along with the application of different stemming algorithms, shown in Table 2.

Stems are the output of a stemming algorithm, which is “[. . .] a computational procedure which reduces all words with the same root (or, if prefixes are left untouched, the same stem) to a common form, usually by stripping each word of its derivational and inflectional suffixes” [9]. A root is the base form of a word and cannot be reduced without losing its identity. An inflectional suffix changes the grammatical role of a word in a sentence, it indicates gender, number, tense, etc. A derivational suffix is used for word-formation. For example, the word “timelessly” has the inflectional suffix “ly” and the derivational suffix “less”.

A word can have at most three co-stems, namely the part before, after, and inside the stem. Depending on the used stemming algorithm, a co-stem can be a single affix or a combination of affixes. Note that most stemming algorithms are language-dependent, and that some stemming algorithms regard a stem as one or more root morphemes plus a derivational suffix (the Lovins stemmer does not).

**Table 2.** Different co-stems for the words “timelessly” and “timeless” resulting from different stemming algorithms

Stemming Algorithm	Co-stem	Stem	Co-stem	Co-stem	Stem	Co-stem	Reference
Porter, Lancaster, Krovetz	-	timeless	ly	-	timeless	-	[14][11][6]
Lovins	-	time	lessly	-	tim	eless	[9]
Truncation(3)	-	tim	elessly	-	tim	eless	
rev. Truncation(3)	-	timeles	sly	-	timel	ess	

## 3 Evaluation

The general setting in our evaluation is as follows: Given a task, the Lovins stemming algorithm [9] computes the stems of an extracted plain text. The algorithm uses a list of

297 suffixes and strips the longest suffix from a word; hence the resulting co-stems in this study are suffixes. Since the goal is to capture the writing style of a text, we enhance the set of co-stems with stopwords and punctuation. A plain text  $d$  is represented by a vector  $\mathbf{x}$ , where each dimension specifies the frequency of its associated token. A token can be a word, a stem, or a co-stem. We apply as classification technologies a generative approach as well as a discriminative approach, namely Naïve Bayes and linear support vector machines (SVM).

**Performance Comparison.** Table 3 compares the classification performances of words, stems, co-stems, and stems combined with co-stems. The symbols  $\circ$  and  $\bullet$  indicate statistically significant improvement and degradation respectively, compared to the bag of words model in a paired T-test with 0.05 significance. For each precision value, recall value,  $F$ -Measure value, and area under ROC curve value (**P**, **R**, **F**, **Auc**) the average is given, weighted by the class distribution. The best solution of a task in terms of the  $F$ -Measure is shown bold. The performance scores are averaged over ten repetitions of a 10-fold cross validation. The table also shows details of the used corpora, whereby each corpus is specific for its respective field and accepted as a comparable standard. Since we consider only binary classification tasks, we randomly select two categories for those corpora that cover more than two categories.

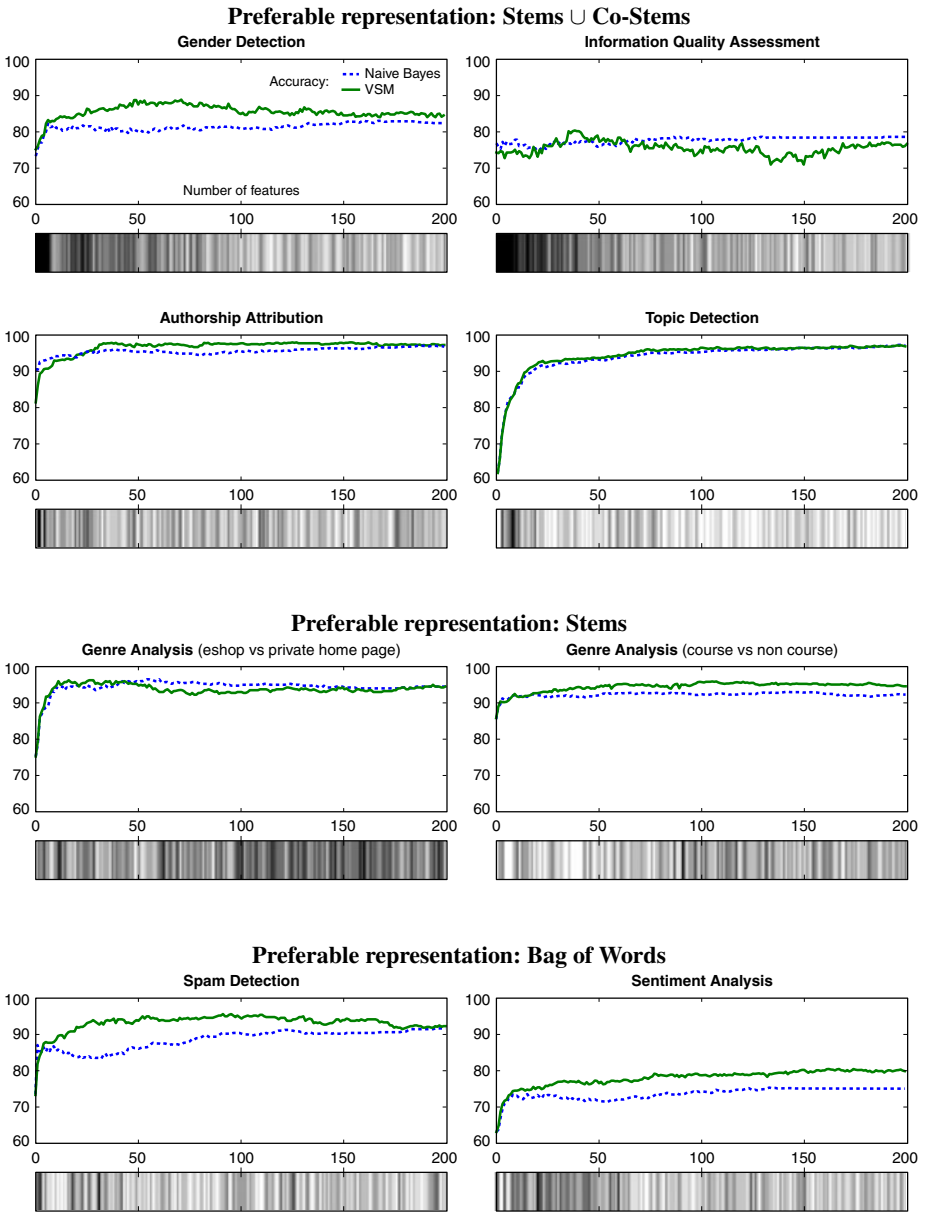
Co-stems are effective in Gender Detection, while the combination of co-stems and stems leads to the best classification result. The combination leads also to the best results in Information Quality Assessment and Authorship Attribution, and it is able to compete in Topic Detection. For Genre Analysis (e-shop vs private home page) and Genre Analysis (course vs non course) the performances of the representation based on stems is comparable with the best solution. Finally, the standard bag of words model performs best in Spam Detection and Sentiment Analysis.

**Influence of co-stems.** To understand of the influence of stems and co-stems in information filtering, Figure 4 illustrates the feature importance characteristics for each task. They show the 10-fold cross validation accuracy scores of the SVM and Naïve Bayes classifiers when the top  $k$  features (stems and co-stems) are used. The top  $k$  features are computed by the information gain criteria on the training split in each fold. The striped bars below the figures illustrate the preference between stems (white) and co-stems (black) according to the information gain criterion among the top 200 features. The frequent occurrence of co-stems in all tasks among the top 200 features emphasize the discriminative power of co-stems. Each task has its own characteristics that are shown by the classification accuracy and the color distribution. A dark left area indicates a superior impact of co-stems with respect to the classification performance in the specific task, which can be observed in particular for Gender Detection and Information Quality Assessment. Co-stems are valuable within tasks where the texts to be filtered typically originate from a specific writer or group of uniform writers. Examples are Authorship Attribution and Information Quality Assessment, where a high quality Wikipedia article is edited by a group of writers who are likely to share style elements.

**Table 3.** Classification performance

Representation	Naïve Bayes				SVM			
	P	R	F	Auc	P	R	F	Auc
<i>Task: Gender Detection</i>								
<i>Corpus:</i> 400/400 blog entries written by different male/female bloggers.								
<i>Source:</i> "The Blog Authorship Corpus" [17] with 681,288 blog entries from 19,320 bloggers on blogger.com.								
Bag of Words	0.72	0.71	0.71	0.78	0.70	0.70	0.70	0.74
Stems $\cup$ Co-Stems	0.82 $\circ$	0.82 $\circ$	0.82 $\circ$	0.90 $\circ$	0.87 $\circ$	0.86 $\circ$	<b>0.86</b> $\circ$	0.91 $\circ$
Stems	0.77 $\circ$	0.77 $\circ$	0.77 $\circ$	0.84 $\circ$	0.80 $\circ$	0.80 $\circ$	0.80 $\circ$	0.85 $\circ$
Co-Stems	0.83 $\circ$	0.83 $\circ$	<b>0.83</b> $\circ$	0.90 $\circ$	0.86 $\circ$	0.85 $\circ$	0.85 $\circ$	0.91 $\circ$
<i>Task: Information Quality Assessment.</i>								
<i>Corpus:</i> 255/255 "featured" (high quality) and "non-featured" articles.								
<i>Source:</i> The english version of Wikipedia.								
Bag of Words	0.79	0.78	0.78	0.87	0.84	0.83	0.83	0.87
Stems $\cup$ Co-Stems	0.80 $\circ$	0.80 $\circ$	<b>0.80</b> $\circ$	0.88 $\circ$	0.87 $\circ$	0.87 $\circ$	<b>0.87</b> $\circ$	0.91 $\circ$
Stems	0.80	0.80	<b>0.80</b>	0.86	0.81 $\bullet$	0.81 $\bullet$	0.81 $\bullet$	0.84 $\bullet$
Co-Stems	0.78	0.78	0.78	0.87	0.86 $\circ$	0.85 $\circ$	0.85 $\circ$	0.91 $\circ$
<i>Task: Authorship Attribution</i>								
<i>Corpus:</i> 357/481 blog entries from one author/from all other authors.								
<i>Source:</i> The engineering category from "The Blog Authorship Corpus" [17].								
Bag of Words	0.98	0.97	<b>0.97</b>	1.00	0.98	0.98	0.98	1.00
Stems $\cup$ Co-Stems	0.97	0.97	<b>0.97</b>	1.00	0.99 $\circ$	0.99 $\circ$	<b>0.99</b> $\circ$	1.00
Stems	0.96 $\bullet$	0.96 $\bullet$	0.96 $\bullet$	1.00	0.98 $\circ$	0.98 $\circ$	0.98 $\circ$	1.00
Co-Stems	0.96 $\bullet$	0.95 $\bullet$	0.95 $\bullet$	1.00	0.95 $\bullet$	0.94 $\bullet$	0.94 $\bullet$	0.99 $\bullet$
<i>Task: Topic Detection.</i>								
<i>Corpus:</i> 1,000/800 messages from the (top-level) newsgroups computer-related discussions/recreation and entertainment.								
<i>Source:</i> The well-known "20 Newsgroups" with 20,000 Usenet articles.								
Bag of Words	0.98	0.98	<b>0.98</b>	1.00	0.97	0.97	0.97	0.99
Stems $\cup$ Co-Stems	0.98	0.98	<b>0.98</b>	1.00	0.98 $\circ$	0.98 $\circ$	<b>0.98</b> $\circ$	0.99
Stems	0.98	0.98	<b>0.98</b>	1.00	0.98 $\circ$	0.98 $\circ$	<b>0.98</b> $\circ$	1.00
Co-Stems	0.83 $\bullet$	0.82 $\bullet$	0.82 $\bullet$	0.90 $\bullet$	0.88 $\bullet$	0.88 $\bullet$	0.88 $\bullet$	0.93 $\bullet$
<i>Task: Genre Analysis (e-shop vs private home page).</i>								
<i>Corpus:</i> 200/200 web pages from eshops/personal home pages.								
<i>Source:</i> The "7-web genre collection" [16] with 1,400 web pages.								
Bag of Words	0.96	0.96	0.96	0.99	0.94	0.94	<b>0.94</b>	0.98
Stems $\cup$ Co-Stems	0.95 $\bullet$	0.94 $\bullet$	0.94 $\bullet$	0.98 $\bullet$	0.94 $\circ$	0.93	0.93	0.98
Stems	0.97 $\circ$	0.97 $\circ$	<b>0.97</b> $\circ$	0.99	0.94	0.93	0.93	0.98
Co-Stems	0.87 $\bullet$	0.85 $\bullet$	0.85 $\bullet$	0.95 $\bullet$	0.88 $\bullet$	0.86 $\bullet$	0.86 $\bullet$	0.94 $\bullet$
<i>Task: Genre Analysis (course vs non course)</i>								
<i>Corpus:</i> 230/821 web pages about courses/non-courses.								
<i>Source:</i> The subset of "The 4 Universities Data Set" used in the Co-training Experiments [2].								
Bag of Words	0.94	0.94	<b>0.94</b>	0.98	0.93	0.91	0.91	0.98
Stems $\cup$ Co-Stems	0.92 $\bullet$	0.92 $\bullet$	0.92 $\bullet$	0.96 $\bullet$	0.91 $\bullet$	0.88 $\bullet$	0.89 $\bullet$	0.98
Stems	0.93 $\bullet$	0.93 $\bullet$	0.93 $\bullet$	0.97 $\bullet$	0.93 $\circ$	<b>0.92</b> $\circ$	0.92 $\circ$	0.98
Co-Stems	0.88 $\bullet$	0.89 $\bullet$	0.88 $\bullet$	0.91 $\bullet$	0.91 $\bullet$	0.90	0.90 $\bullet$	0.95 $\bullet$
<i>Task: Spam Detection</i>								
<i>Corpus:</i> 160/320 spam/non-spam emails.								
<i>Source:</i> The "SpamAssassin public email corpus" with 1,397 spam and 2,500 non-spam emails. <a href="http://spamassassin.apache.org">http://spamassassin.apache.org</a>								
Bag of Words	0.92	0.92	<b>0.92</b>	0.97	0.95	0.94	<b>0.94</b>	0.98
Stems $\cup$ Co-Stems	0.92	0.91 $\bullet$	0.91 $\bullet$	0.96 $\bullet$	0.93 $\bullet$	0.91 $\bullet$	0.91	0.98 $\bullet$
Stems	0.92	0.91 $\bullet$	0.91 $\bullet$	0.96 $\bullet$	0.94 $\bullet$	0.92 $\bullet$	0.93	0.98 $\bullet$
Co-Stems	0.89 $\bullet$	0.89 $\bullet$	0.89 $\bullet$	0.95 $\bullet$	0.93 $\bullet$	0.93 $\bullet$	0.93 $\bullet$	0.96 $\bullet$
<i>Task: Sentiment Analysis</i>								
<i>Corpus:</i> 1,000/1,000 positive/negative movie reviews.								
<i>Source:</i> The "Cornell Movie Review Dataset" [12] with 1,000 positive and 1,000 negative reviews.								
Bag of Words	0.80	0.80	<b>0.80</b>	0.88	0.85	0.85	<b>0.85</b>	0.91
Stems $\cup$ Co-Stems	0.76 $\bullet$	0.76 $\bullet$	0.75 $\bullet$	0.84 $\bullet$	0.84 $\bullet$	0.83 $\bullet$	0.83 $\bullet$	0.91
Stems	0.81	0.80	<b>0.80</b>	0.89	0.82 $\bullet$	0.82 $\bullet$	0.82 $\bullet$	0.89 $\bullet$
Co-Stems	0.63 $\bullet$	0.62 $\bullet$	0.62 $\bullet$	0.68 $\bullet$	0.72 $\bullet$	0.72 $\bullet$	0.71 $\bullet$	0.79 $\bullet$





**Fig. 1.** Task-specific discrimination analysis of stems and co-stems. Each plot shows the classification accuracy ( $y$ -axis) over the number  $m$  of employed features ( $x$ -axis),  $m \in [1, 200]$ , for a given task. The two curves correspond to the Naïve Bayes classifier (dotted blue) and SVM (solid green) respectively. The striped bars below the plots illustrate whether a stem (white) or a co-stem (black) is chosen by the information gain criterion as the  $m$ -th feature: the results are obtained from a 10-fold cross validation, and the exact value of the average calculation is reflected by a gray-scale value. A dark left area indicates the superiority of co-stems over stems.

## 4 Conclusion

Each information filtering task has its own characteristics in terms of the importance of co-stems. For the tasks Gender Detection, Information Quality Assessment, and Authorship Attribution the combination of stems and co-stems leads to a statistically significant improvement compared to the bag of words model. We provide evidence for the discriminative power of co-stems by setting up experiments with accepted corpora, and by analyzing and illustrating the distribution of the top discriminating features.

## References

1. Blanzieri, E., Bryl, A.: A survey of learning-based techniques of email spam filtering. *Artificial Intelligence Review* 29(1), 63–92 (2008)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: *Proc. of the Workshop on Computational Learning Theory*, pp. 92–100 (1998)
3. Gottron, T., Lipka, N.: A comparison of language identification approaches on short, query-style texts. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Ruger, S., van Rijsbergen, K. (eds.) *ECIR 2010. LNCS*, vol. 5993, pp. 611–614. Springer, Heidelberg (2010)
4. Hanani, U., Shapira, B., Shoval, P.: Information filtering: Overview of issues, research and systems. *User Modeling and User-Adapted Interaction* 11(3), 203–259 (2001)
5. Koppel, M., Schler, J.: Authorship verification as a one-class classification problem. In: *Proc. of ICML*, p. 62 (2004)
6. Krovetz, R.: Viewing morphology as an inference process. In: *Proc. of SIGIR*, pp. 191–202 (1993)
7. Lang, K.: Newsweeder: learning to filter netnews. In: *Proc. of ICML*, pp. 331–339 (1995)
8. Lipka, N., Stein, B.: Identifying Featured Articles in Wikipedia: Writing Style Matters. In: *Proc. of WWW*, pp. 1147–1148 (2010)
9. Lovins, J.B.: Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics* 11, 22–31 (1968)
10. Ntoulas, A., Najork, M., Manasse, M., Fetterly, D.: Detecting spam web pages through content analysis. In: *Proc. of WWW*, pp. 83–92 (2006)
11. Paice, C.D.: Another Stemmer. *SIGIR Forum* 24(3), 56–61 (1990)
12. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: *Proc. of ACL*, pp. 271–278 (2004)
13. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pp. 79–86 (2002)
14. Porter, M.F.: An algorithm for suffix stripping. *Program: Electronic Library & Information Systems* 40(3), 211–218 (1980)
15. Friedhorsky, R., Chen, J., Lam, S.T.K., Panciera, K., Terveen, L., Riedl, J.: Creating, destroying, and restoring value in wikipedia. In: *GROUP 2007: Proc. of the International ACM Conference on Supporting Group Work*, pp. 259–268 (2007)
16. Santini, M.: Common criteria for genre classification: Annotation and granularity. In: *Third International Workshop on Text-Based Information Retrieval* (2006)

17. Schler, J., Koppel, M., Argamon, S., Pennebaker, J.: Effects of age and gender on blogging. In: Proc. of AAAI - Symposium on Computational Approaches for Analyzing Weblogs, pp. 191–197 (2006)
18. Stamatatos, E.: A survey of modern authorship attribution methods. *JASIST* 60, 538–556 (2009)
19. Stein, B., Eissen, S.M.Z., Lipka, N.: Web genre analysis: Use cases, retrieval models, and implementation issues. *Genres on the Web* 42, 167–189 (2011)
20. Tsur, O., Davidov, D., Rappoport, A.: A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Product Reviews. In: Proc. of AAAI - ICWSM (2010)

# Interactive Trademark Image Retrieval by Fusing Semantic and Visual Content

Marçal Rusiñol, David Aldavert, Dimosthenis Karatzas,  
Ricardo Toledo, and Josep Lladós

Computer Vision Center, Dept. Ciències de la Computació  
Edifici O, Univ. Autònoma de Barcelona  
08193 Bellaterra (Barcelona), Spain  
{marcal,aldavert,dimos,ricard,josep}@cvc.uab.cat

**Abstract.** In this paper we propose an efficient queried-by-example retrieval system which is able to retrieve trademark images by similarity from patent and trademark offices' digital libraries. Logo images are described by both their semantic content, by means of the Vienna codes, and their visual contents, by using shape and color as visual cues. The trademark descriptors are then indexed by a locality-sensitive hashing data structure aiming to perform approximate  $k$ -NN search in high dimensional spaces in sub-linear time. The resulting ranked lists are combined by using a weighted Condorcet method and a relevance feedback step helps to iteratively revise the query and refine the obtained results. The experiments demonstrate the effectiveness and efficiency of this system on a realistic and large dataset.

**Keywords:** Multimedia Information Retrieval, Trademark Image Retrieval, Graphics Recognition, Feature Indexing.

## 1 Introduction

The digital libraries of patent and trademark offices contain millions of trademark images registered over the years. When registering a new trademark it is important to browse these databases in order to be sure that there is no other company having a similar logo design so as to avoid infringing someone else's intellectual property rights. However, nowadays the means we have to browse and retrieve information from these databases are not really useful enough to assess if a logo to be registered might provoke trademark infringement. Usually, the way that Intellectual Property Offices offer to navigate through the trademark image collections is by the use of subject terms that aim to catalogue the image contents. Each logo is labelled with a set of predefined metadata which enable to search in the collection by the selection of a set of labels. The most widely used metadata classification codes in the Intellectual Property Offices are the ones from the Vienna classification system, developed by the World Intellectual Property Organization [16]. This manually annotation of the logos' contents has the advantage of clustering the logos by semantic information. This categorization system is hierarchical so the category definition can be specified from

broad concepts to more specific contents. As an example in the broad categories we can find *Category 1: Celestial bodies, natural phenomena, geographical maps*, and under this category we find for instance *class 1.7.6: Crescent moon, half-moon* or *class 1.15.3: Lightning*. Another example for broad categories would be *Category 17: Horological instruments, jewelry, weights and measures*, where on the more specific classes we find, *class 17.2.13: Necklaces, bracelets, jewelry chains*, or *class 17.1.9: Clocks, alarm clocks*. However this classification approach presents some limitations, specially for abstract or artistic images, where the use of manual classification codes is not always distinctive. The user that browses a trademark image database is usually looking for images that *look* similar to the one that is willing to register, but images in the same semantic category are usually far from being similar one to each other, so a tedious manual inspection of all the logos in a given category is still needed to retrieve near-duplicate images. In that specific scenario it would be very interesting if we could enhance this existing retrieval mechanism with techniques from the Content-based Image Retrieval (CBIR) domain which help us to describe trademark images in terms of their visual similarity to the given query.

Since the early years of CBIR, a lot of effort has been devoted to the particular application of trademark image retrieval. Systems like STAR [17], ARTISAN [3] or TAST [11] were proposed to be used in different trademark offices to index logo images by visual similarity. These systems just focused on binary device mark images and use shape and texture as the only indicators of similarity. In the latest years some attempts to merge shape and color for the retrieval of trademark images have been proposed. For instance, the method presented by Hsieh and Fan in [8] use the RGB color values as input for a region growing algorithm that further characterizes the trademark designs with a shape description of these color regions. In [6], Hitam et al. propose to use the spatial distribution of RGB values to describe trademarks by means of which colors compose them and where they appear. More recently, Hesson and Androutsos proposed in [5] to describe trademark images with the use of a color naming algorithm instead of the raw RGB values in order to provide a response that agrees more with the human perception of colors. From the perceptual point of view, works like [14], [7] or [9] introduce the use of the Gestalt theory to the description of trademarks. All these systems perform well at retrieving relevant images by visual similarity, but they all discard semantic information that in some cases might also be very relevant. To our best knowledge very few attempts have been made to combine semantic and visual information in a trademark retrieval system. Ravela et al. presented in [13] a system that retrieves binary trademark images by restricting the search to a set of predefined semantic categories. In our paper we extend this idea by proposing a method that takes into account semantic information without restricting the search corpus to a specific subset of images. In addition, relevance feedback is used to refine the semantic and visual cues.

The main contribution of this paper is to present an efficient queried-by-example retrieval system which is able to retrieve logos by visual similarity and semantic content from large databases of isolated trademark images. Logos are

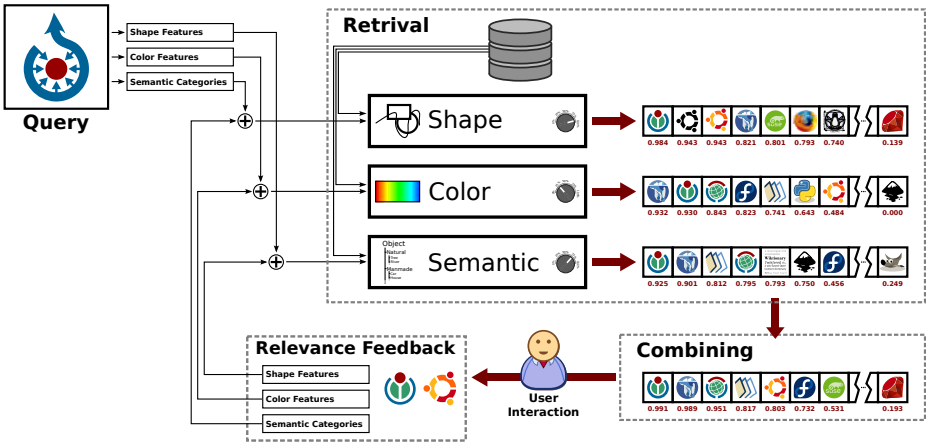


Fig. 1. Overview of the proposed system

compactly described by descriptors of their shape and color. The semantic description of the trademark images is given by a hierarchical organization of the Vienna codes. These descriptors are then organized by a locality-sensitive hashing indexing structure aiming to perform approximate  $k$ -NN search in high dimensional spaces in sub-linear time. The use of a hashing technique allow us to quickly index and retrieve queried logos by visual similarity and semantic content. The resulting ranked lists are then fused by using the Condorcet method and a relevance feedback step helps to iteratively revise the query and refine the obtained results. To conduct the experimental results, we will focus on a large collection of real-world trademark images registered to the Spanish Intellectual Property Office.

The remainder of this paper is organized as follows: section 2 is devoted to present an overview of the system. In section 3 the logo description using visual cues is presented, while in section 4 the semantic representation provided by the Vienna codes is introduced. Subsequently, in section 5, the indexing structure for efficient retrieval, the ranking combination algorithm and the relevance feedback strategy are presented. Section 6 provides the experimental results and finally section 7 is a summary and discussion of extensions and future work.

## 2 System Overview

We can see in Figure 1 an overview of the proposed system. The user provides a query trademark image he wants to register and, optionally, a set of semantic concepts from the Vienna classification codes. Shape and color features are extracted from the query image and an indexing mechanism efficiently retrieves from the database the trademarks that have a shape or color description similar to the query one. These ranked lists are combined to form a single resulting list which is presented to the user. The user can then refine the search by selecting the logo images that are considered to be relevant. This relevance feedback

mechanism allows the user’s search query to be revised in order to include a percentage of relevant and non-relevant documents as a means of increasing the search engine’s precision and recall. The relevance feedback step is also a way to weight the importance of a visual cue. In some cases the color might be relevant to the user despite the shape (dis)similarity or viceversa. In addition, in the relevance feedback step, if the user did not provide any semantic concept in the query, these are deduced from the selection of relevant documents and used in the following iterations in order to obtain a refined result list. If the user starts selecting trademarks from a given semantic class despite being dissimilar in both shape and color, the system automatically adapts the queries and the combination of the results to the user needs, giving more importance to the semantics than to the visual information. After this brief overview, let us present each of the system modules, starting with the visual description of trademark images.

### 3 Visual Description of Trademark Images

We base our visual description of logo images on two separate descriptors, one encoding shape information and the other describing the colors of the trademark designs. For the shape information we use the shape context descriptor and for the color we use a quantization of the  $CIE L^*C^*h$  color space. Let us first briefly overview the shape context descriptor, and then focus on how we describe trademark images by color.

#### 3.1 Shape Information

The shape context descriptor was proposed by Belongie et al. in [1]. This descriptor allows to measure shape similarity by recovering point correspondences between the two shapes under analysis. In a first step, a set of interest points has to be selected from the logos. Usually, a Canny edge detector is used and the edge elements are sub-sampled in order to obtain a fixed number of  $n$  points  $p_i$  per logo  $\ell$ . For each of these  $n$  points, the shape context captures the edge point distribution within its neighborhood. A histogram using log-polar coordinates counts the number of points inside each bin. For a point  $p_i$  of the shape, a histogram  $h_i$  of the coordinates of the nearby points  $q$  is computed as:

$$h_i(k) = \# \{q \neq p_i : q \in \text{bin}_{p_i}(k)\} \quad (1)$$

In our experimental setup, we have chosen 5 bins for  $\log r$  and 12 bins for  $\theta$ . The descriptor offers a compact representation of the distribution of points relative to each selected point. Once all the  $n$  points in a shape are described by their shape context descriptor, we compute the shapeme histogram descriptor presented by Mori et al. in [12] to efficiently match two trademarks. This description technique was inspired by the shape context descriptor described above, and the bag-of-words model. The main idea is to compute the shape context descriptor for all the interest points extracted from a symbol and then use vector quantization in the space of shape contexts. Vector quantization involves a clustering stage of

the shape context feature space. Once the clustering is computed, each shape context descriptor can be identified by the index of the cluster which it belongs to. These clusters are called shapemes. Each logo is then described by a single histogram representing the frequencies of appearance of each shapeme.

In the learning stage, given a set of model logos, we compute their shape context descriptors and cluster this space by means of the  $k$ -means algorithm, identifying a set of  $k$  cluster centers and assigning to them a given integer index  $I \in [1, k]$ . Then, in the recognition stage, given a logo  $\ell$ , and its  $n$  sampled points from its edge map, we compute their shape context descriptors  $h_i, \forall i \in [0, n]$ . Each shape context descriptor of the points  $p_i$  is then projected to the clustered space and can be identified by a single index  $I_i$ . The logo  $\ell$  can thus be represented by a single histogram coding the frequency of appearance of each of the  $k$  shapeme indices. By this means, we globally describe by a unique histogram  $SH$  each logo by applying the following equation:

$$SH(x) = \# \{I_i == x : I_i \in [0, k]\} \quad (2)$$

By using the shapeme histogram descriptor, the matching of two logos is reduced to find the  $k$ -NN in the space of shapeme histograms, avoiding much more complex matching strategies. We can see an example in Table II of the performance of this descriptor to retrieve trademark images by shape.

### 3.2 Color Information

In order to represent the color information of the trademark images, we use a color quantization process in a color space that is perceptually uniform, i.e. distances in this space agree with the human perception on whether two colors are similar or not.

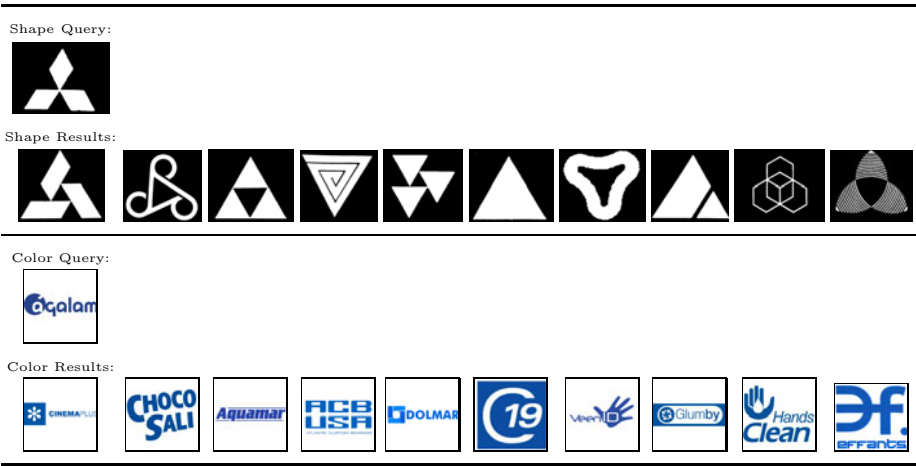
We transform the RGB trademark image to the  $CIEL^*C^*h$  color space assuming D65 as a reference illuminant. Details on the formulae can be found in [18]. The  $L^*$  component corresponds to the lightness, the  $C^*$  component to the chroma and the  $h$  component to the hue. After having converted the image to a perceptually uniform color space, we compute the color descriptor of the image as a histogram of appearances of different colors. As we did with the shape descriptor, the  $k$ -means method clusters the  $CIEL^*C^*h$  color space. Once the clustering is computed, each pixel value can be identified and quantized by the index of the cluster which it belongs to. Each logo is then described by a single histogram representing the frequencies of appearance of each color representatives.

In the learning stage, given a set of model logos, we convert the images to the  $CIEL^*C^*h$  color space and cluster this space by using the  $k$ -means algorithm, identifying a set of  $q$  color representatives with indices  $J \in [1, q]$ . Then, in the recognition stage, given a logo  $\ell$  converted to the  $CIEL^*C^*h$  color space, each pixel  $P_{xy}$  is projected to the clustered space and it is identified by an index  $J_{xy}$ . The logo  $\ell$  is represented by a single histogram coding the frequency of appearance of each of the  $q$  color indices. We globally describe by a unique histogram  $CO$  each logo by applying the following equation:

$$CO(x) = \# \{J_{xy} == x : J_{xy} \in [0, q]\} \quad (3)$$



**Table 1.** Queries and first ten results by shape and color similarity respectively



By using the color histogram descriptor, the matching of two logos is reduced to find the  $k$ -NN in the space of color histogram. We can see an example in Table 1 of the performance of this descriptor to retrieve trademark images by color.

### 4 Description of Trademark Images by Vienna Codes

The Vienna classification codes [16] offer a hierarchical categorization of the image contents. In our system, all the trademarks in the database have an associated list of Vienna codes. We can see in Figure 2 a set of trademark images all belonging to the same category. As we can appreciate, there are few visual similarities yet all contain a horse. For a given trademark image and its associated Vienna code, we will define four different sets  $S_1 \dots S_4$  that cluster the dataset depending on its semantical agreement. In  $S_1$  we will have all the trademark images under the same Vienna category, for the trademarks in the example that would be all the trademarks in category 03.03.01 (containing horses or mules). In  $S_2$  we store all the trademark images that are under category 03.03 (horses, mules, donkeys, zebras) without being in  $S_1$ , in  $S_3$  all the trademarks under category 03 (Animals). Finally,  $S_4$  is the set of trademark images without any semantic agreement with the query.



**Fig. 2.** Example of searching logos by using the category tree of Vienna codes (a); and example of all the associated Vienna codes for a given trademark image (b)

Since a trademark image can have more than one Vienna code, each trademark image is stored in the set that agrees more with the query. Note that for the semantic categorization of trademarks, the retrieval result does not produce a ranked list, but sets of images. The only notion of order here is that elements in  $S_1$  are better ranked than elements in  $S_2$  and so on.

## 5 Retrieval Stage

In this section, we are going to present in detail the retrieval stage of the proposed method. We first introduce the indexing mechanism that allows to retrieve trademarks by visual similarity. Then, we present how visual and semantic cues are combined together. Finally, we detail the relevance feedback step where the user interaction is taken into account in order to refine the obtained results.

### 5.1 Indexing Trademark Visual Descriptors with LSH

In order to avoid a brute force matching of the shape and color feature vectors, we propose to use an algorithm for approximate  $k$ -NN search. This allows to efficiently obtain a set of candidates that probably lie nearby the queried point. The method we use is the locality-sensitive hashing (LSH), first introduced by Indyk and Motwani in [10], and then revised by Gionis et al. in [4]. The LSH algorithm has been proven to perform approximate  $k$ -NN search in sub-linear time.

The basic idea of the LSH method is to index points from a database by using a number  $l$  of naïve hash functions  $g$ , in order to ensure that objects close in the feature space have a high probability of provoking collisions in the hash tables. Given a query feature vector  $q$  the algorithm iterates over the  $l$  hash functions  $g$ . For each  $g$  considered, it retrieves the data points that are hashed into the same bucket as  $q$ . In our experimental setup, we are using  $l = 50$  simple hash functions. We can see in Figure 3 a comparison of the average time taken to retrieve the 50 topmost similar vectors for different sizes of the database.

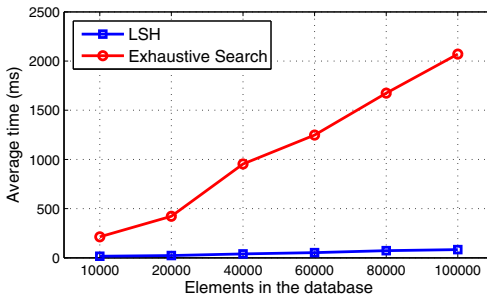


Fig. 3. Average time to retrieve the 50-NN for different database sizes

## 5.2 Ranking Combination and Relevance Feedback

The retrieval of trademark images either by shape or by color result in a ranked list. Those lists need to be combined to return to the user a single result list. However, the semantic part do not provide the results in a ranked list but in separate sets. In order to combine these results we choose to use a weighted Condorcet method [2]. The particularity of this method compared to other classic combination algorithms, is that in can handle tied results.

The Condorcet algorithm is a method which specifies that the winner of the election is the candidate that beats or ties with every other candidate in a pair-wise comparison. In our case, we consider all the trademarks in the same semantic set as tied whereas the trademarks in upper sets are beaten. Given a query trademark and the ranked lists for the shape, the color and the ordered sets for the semantic retrieval, to obtain a final rank of trademark images we use their win and lose values. If the number of wins that a logo has is higher than the other one, then that trademark wins. Otherwise if their win property is equal we consider their lose scores, the trademark image which has smaller lose score wins. By letting the user weight the contribution of color, shape and semantics, we obtain a weighted Condorcet method. The weighting factor allow to fuse the ranked lists taking into account the user's needs, i.e. selecting the information cue more relevant to its query. Albeit the computational cost of the Condorcet algorithm is high, we limit the comparisons of wins and loses to the best subset of elements returned by the LSH algorithm instead of using the full dataset corpus (in our experiments limited to best 100 logos by shape and by color). Therefore, the cost of applying the Condorcet algorithm is kept bounded.

After applying the Condorcet algorithm, the user receives a single sorted list with the trademarks that are closer to the query in terms of visual similarity (shape and color) and semantic content. Once we present this list to the user, we include a relevance feedback step so as to give the chance to the user to refine the results.

Relevance feedback should be a must for any trademark image retrieval application. In our case, in order to include the feedback from the user, we use Rocchio's algorithm [15] to revise the vector queries and weight the importance of shape, color and semantics depending on the user needs. At each iteration the Rocchio's algorithm computes a new point in the query space aiming to incorporate relevance feedback information into the vector space model. The modified query vector  $\vec{Q}_m$  is computed as

$$\vec{Q}_m = \left( \alpha * \vec{Q}_o \right) + \left( \beta * \frac{1}{|D_r|} * \sum_{\vec{D}_j \in D_r} \vec{D}_j \right) - \left( \gamma * \frac{1}{|D_{nr}|} * \sum_{\vec{D}_k \in D_{nr}} \vec{D}_k \right) \quad (4)$$

where  $\vec{Q}_o$  is the original query vector, and  $D_r$  and  $D_{nr}$  the sets of relevant and non-relevant trademark images respectively.  $\alpha$ ,  $\beta$  and  $\gamma$  are the associated weights that shape the modified query vector respect the original query, the relevant and non-relevant items. In our experimental setup we have chosen the

following values  $\alpha = 0.55$ ,  $\beta = 0.4$  and  $\gamma = 0.05$  in order to keep  $\vec{Q}_m$  normalized and within the feature space.

## 6 Experimental Results

Let us first introduce the logo dataset we used in our experiments. We will then present the experimental results.



Fig. 4. Retrieval results obtained looking for near-duplicate trademarks in the first query and semantic similar logos in the second

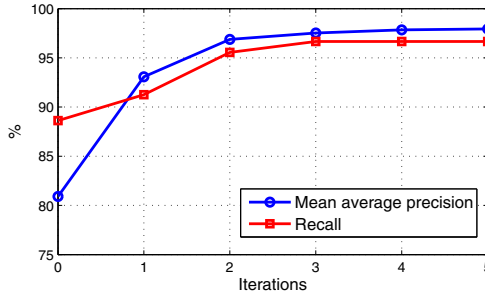


Fig. 5. Mean average precision and mean recall at different iterations

## 6.1 Trademark Dataset

To conduct the experimental results, we will focus on a collection of real trademark images downloaded from the Spanish Intellectual Property Office<sup>1</sup> with their associated Vienna codes. This dataset is composed by all the trademarks registered during the year 2009, that is near 30000 trademark images organized within 1350 different Vienna categories. In average, each trademark image has associated 2.17 Vienna codes. Another subset of 3000 trademark images has been used as training set to run the  $k$ -means clustering algorithm to build both the shape and color descriptors.

## 6.2 Method Evaluation

The first experiment presents in Figure 4 a qualitative evaluation of the proposed method. For the first query, we search for near-duplicate images in the database. In that case, the visual cues are the most important despite some trademarks from the semantic category “star” appear. Surrounded in green we present the trademarks selected as relevant by the user at each iteration. As we can see, the results are qualitatively better iteration after iteration. In the second query we show in Figure 4, we do not expect to retrieve near-duplicates but perceptually similar logo images. The user selects trademarks which contain a tree with some text somewhere in the image. In that case, initially the proposed method fails in the topmost images, but iteration after iteration the results are correctly reranked.

The second experiment provides a simple quantitative evaluation of the use of the relevance feedback step. We selected six different trademarks that have several (more than 10) near-duplicate entries in the database as our queries. These near-duplicate images are labelled as being the only relevant answers that the system might provide. With this groundtruth data, we are able to compute the precision and recall measures. We can see in Figure 5 the evolution of the mean average precision and recall measures (computed at the topmost 30 elements in return) through the successive iterations of the loop composed by retrieval and further user feedback. As we can appreciate, both measures present an important increase before reaching stability.

<sup>1</sup> <http://www.oepm.es>

## 7 Conclusions

In this paper, we have presented an interactive trademark image retrieval system which combines both visual and semantic information to obtain the most similar logos from a large realistic database. The addition of the semantic information provided by the standard Vienna codes increases the retrieval performance, by reducing the amount of possible similar logos and also enabling the method to retrieve trademarks which are dissimilar in shape or color content but have an strong semantic connection. Besides, the addition of user feedback allows to further refine the obtained results and it permits to automatically generate the semantic descriptor of the query when it is not given by the user. The qualitative evaluation of the method show a good performance of the system in retrieving trademarks by combining visual terms and semantic concepts. Besides, the use of relevance feedback steadily increases the mean average precision and recall of the system when used for searching near-duplicate trademark images in the database.

## Acknowledgments

This work has been partially supported by the Spanish Ministry of Education and Science under projects TIN2008-04998, TIN2009-14633-C03-03, TRA2010-21371-C03-01, Consolider Ingenio 2010: MIPRCV (CSD200700018) and the grant 2009-SGR-1434 of the Generalitat de Catalunya.

## References

1. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.* 24(4), 509–522 (2002)
2. de Condorcet, M.: *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix.* 1785
3. Eakins, J.P., Shields, K., Boardman, J.: ARTISAN: A shape retrieval system based on boundary family indexing. In: *Storage and Retrieval for Still Image and Video Databases IV*. Proc. SPIE, vol. 2670, pp. 17–28 (1996)
4. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: *Proceedings of the 25th International Conference on Very Large Data Bases*, pp. 518–529 (1999)
5. Hesson, A., Androustos, D.: Logo and trademark detection in images using color wavelet co-occurrence histograms. In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1233–1236 (2008)
6. Hitam, M.S., Yussof, H.J.W., Jawahir, W.N., Deris, M.M.: Hybrid Zernike moments and color-spatial technique for content-based trademark retrieval. In: *Proceedings of the International Symposium on Management Engineering* (2006)
7. Hodge, V.J., Hollier, G., Austin, J., Eakins, J.: Identifying perceptual structures in trademark images. In: *Proceedings 5th IASTED International Conference on Signal Processing, Pattern Recognition, and Applications* (2008)
8. Hsieh, I.S., Fan, K.C.: Multiple classifiers for color flag and trademark image retrieval. *IEEE Trans. Image Process.* 10(6), 938–950 (2001)

9. Iguchi, H., Abe, K., Misawa, T., Kimura, H., Daido, Y.: Recognition of grouping patterns in trademarks based on the Gestalt psychology. *Electron. Commun. Jpn.* 92(10), 49–60 (2009)
10. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp. 604–613 (1998)
11. Kwan, P.W.H., Toraichi, K., Kameyama, K., Kawazoe, F., Nakamura, K.: TAST-trademark application assistant. In: *Proceedings of the International Conference on Image Processing*, pp. 884–887 (2002)
12. Mori, G., Belongie, S., Malik, J.: Efficient shape matching using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.* 27(11), 1832–1837 (2005)
13. Ravela, S., Manmatha, R.: Multi-modal retrieval of trademark images using global similarity. Technical Report UM-CS-1999-032, University of Massachusetts (1999)
14. Ren, M., Eakins, J.P., Briggs, P.: Human perception of trademark images: Implications for retrieval system design. *J. Electron. Imaging* 9(4), 564–575 (2000)
15. Rocchio, J.: Relevance feedback in information retrieval. In: *SMART Retrieval System: Experiments in Automatic Document Processing*, pp. 313–323 (1971)
16. Schietse, J., Eakins, J.P., Veltkamp, R.C.: Practice and challenges in trademark image retrieval. In: *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, pp. 518–524 (2007)
17. Wu, J.K., Mehtre, B.M., Gao, Y.J., Lam, P.C., Narasimhalu, A.D.: STAR - a multimedia database system for trademark registration. In: Risch, T., Litwin, W. (eds.) *ADB 1994. LNCS*, vol. 819. Springer, Heidelberg (1994)
18. Wyszecki, G., Stiles, W.S.: *Color science: Concepts and methods, quantitative data and formulae*. John Wiley and Sons, New York (1982)

# Dynamic Two-Stage Image Retrieval from Large Multimodal Databases

Avi Arampatzis, Konstantinos Zagoris, and Savvas A. Chatzichristofis

Department of Electrical and Computer Engineering,  
Democritus University of Thrace, Xanthi 67100, Greece  
{avi,kzagoris,schatzic}@ee.duth.gr

**Abstract.** Content-based image retrieval (CBIR) with global features is notoriously noisy, especially for image queries with low percentages of relevant images in a collection. Moreover, CBIR typically ranks the whole collection, which is inefficient for large databases. We experiment with a method for image retrieval from multimodal databases, which improves both the effectiveness and efficiency of traditional CBIR by exploring secondary modalities. We perform retrieval in a two-stage fashion: first rank by a secondary modality, and then perform CBIR only on the top- $K$  items. Thus, effectiveness is improved by performing CBIR on a ‘better’ subset. Using a relatively ‘cheap’ first stage, efficiency is also improved via the fewer CBIR operations performed. Our main novelty is that  $K$  is dynamic, i.e. estimated per query to optimize a predefined effectiveness measure. We show that such dynamic two-stage setups can be significantly more effective and robust than similar setups with static thresholds previously proposed.

## 1 Introduction

In content-based image retrieval (CBIR), images are represented by global or local features. Global features are capable of generalizing an entire image with a single vector, describing color, texture, or shape. Local features are computed at multiple points on an image and are capable of recognizing objects.

CBIR with global features is notoriously noisy for image queries of low *generality*, i.e. the fraction of relevant images in a collection. In contrast to text retrieval where documents matching no query keyword are not retrieved, CBIR methods typically rank the whole collection via some distance measure. For example, a query image of a red tomato on white background would retrieve a red pie-chart on white paper. If the query image happens to have a low generality, early rank positions may be dominated by spurious results such as the pie-chart, which may even be ranked before tomato images on non-white backgrounds. Figures 2a-b demonstrate this particular problem.

Local-feature approaches provide a slightly better retrieval effectiveness than global features [1]. They represent images with multiple points in a feature space in contrast to single-point global feature representations. While local approaches provide more robust information, they are more expensive computationally due to the high dimensionality of their feature spaces and usually need nearest neighbors approximation to perform points-matching [18]. High-dimensional indexing still remains a challenging problem in the database field. Thus, global features are more popular in CBIR systems as they



are easier to handle and still provide basic retrieval mechanisms. In any case, CBIR with either local or global features does not scale up well to large databases efficiency-wise. In small databases, a simple sequential scan may be acceptable, however, scaling up to millions or billion images efficient indexing algorithms are imperative [15].

Nowadays, information collections are not only large, but they may also be *multi-modal*. Take as an example Wikipedia, where a single topic may be covered in several languages and include non-textual media such as image, sound, and video. Moreover, non-textual media may be annotated in several languages in a variety of metadata fields such as object caption, description, comment, and filename. In an image retrieval system where users are assumed to target visual similarity, all modalities beyond image can be considered as secondary; nevertheless, they can still provide useful information for improving image retrieval.

In this paper, we experiment with a method for image retrieval from large multimodal databases, which targets to improve both the effectiveness and efficiency of traditional CBIR by exploring information from secondary modalities. In the setup considered, an information need is expressed by a query in the primary modality (i.e. an image example) accompanied by a query in a secondary modality (e.g. text). The core idea for improving effectiveness is to raise query generality before performing CBIR, by reducing collection size via filtering methods. In this respect, we perform retrieval in a two-stage fashion: first use the secondary modality to rank the collection and then perform CBIR only on the top- $K$  items. Using a ‘cheaper’ secondary modality, this improves also efficiency by cutting down on costly CBIR operations.

Best results re-ranking by visual content has been seen before, but mostly in different setups than the one we consider or for different purposes, e.g. result clustering [4] or diversity [12]. Others used external information, e.g. an external set of diversified images [18] (also, they did not use image queries), web images to depict a topic [17], or training data [5]. All these approaches, as well as [16], employed a static predefined  $K$  for all queries, except [18] who re-ranked the top-30% of retrieved items. They all used global features for images. Effectiveness results have been mixed; it worked for some, it did not for others, while some did not provide a comparative evaluation or system-study. Later, we will review the aforementioned literature in more detail.

In view of the related literature, our main contributions are the following. Firstly, our threshold is calculated *dynamically* per query to optimize a predefined effectiveness measure, without using external information or training data; this is also our biggest novelty. We show that the choice between static or dynamic thresholding can make the difference between failure and success of two-stage setups. Secondly, we provide an extensive evaluation in relation to thresholding types and levels, showing that dynamic thresholding is not only more effective but also more robust than static. Thirdly, we investigate the influence of different effectiveness levels of the second visual stage on the whole two-stage procedure. Fourthly, we provide a comprehensive review of related literature and discuss the conditions under which such setups can be applied effectively. In summary, with a simpler two-stage setup than most previously proposed in the literature, we achieve significant improvements over retrieval with text-only, several image-only, and two-stage with static thresholding setups.

The rest of the paper is organized as follows. In Section 2 we discuss the assumptions, hypotheses, and requirements behind two-stage image retrieval from multimodal databases. In Section 3 we perform an experiment on a standardized multimodal snapshot of Wikipedia. In Section 4 we review related work. Conclusions and directions for further research are summarized in Section 5.

## 2 Two-Stage Image Retrieval from Multimodal Databases

Multimodal databases consist of multiple descriptions or media for each retrievable item; in the setup we consider these are image and annotations. On the one hand, textual descriptions are key to retrieve relevant results for a query but at the same time provide little information about the image content [12]. On the other hand, the visual content of images contains large amounts of information which can hardly be described by words.

Traditionally, the method that has been followed in order to deal effectively with multimodal databases is to search the modalities separately and fuse their results, e.g. with a linear combination of the retrieval scores of all modalities per item. While fusion has been proved robust, it has a few issues: a) appropriate weighing of modalities is not a trivial problem and may require training data, b) total search time is the sum of the times taken for searching the participating modalities, and most importantly, c) it is not a theoretically sound method if results are assessed by visual similarity only; the influence of textual scores may worsen the visual quality of end-results. The latter issue points to that there is a *primary modality*, i.e. the one targeted and assessed by users.

An approach that may tackle the issues of fusion would be to search in a two-stage fashion: first rank with a secondary modality, draw a rank-threshold, and then re-rank only the top items with the primary modality. The assumption on which such a two-stage setup is based on is the existence of a primary modality, and the success would largely depend on the *relative effectiveness* of the two modalities involved. For example, if text retrieval always performs better than CBIR (irrespective of query generality), then CBIR is redundant. If it is the other way around, only CBIR will be sufficient. Thus, the hypothesis is that CBIR can do better than text retrieval in small sets or sets of high query generality.

In order to reduce collection size raising query generality, a ranking can be thresholded at an arbitrary rank or item score. This improves the efficiency by cutting down on costly CBIR operations, but it may not improve too much the result quality: a too tight threshold would produce similar results to a text-only search making CBIR redundant, while a too loose threshold would produce results haunted by the red-tomato/red-pie-chart effect mentioned in the Introduction. Three factors determine what the right threshold is: 1) the number of relevant items in the collection, 2) the quality of the ranking, and 3) the measure that the threshold targets to optimize [20]. The first two factors are query-dependent, thus thresholds should be selected *dynamically* per query, not statically as most previously proposed methods in the literature (reviewed in Section 4).

The approach of [18], who re-rank the top-30% retrieved items which can be considered dynamic, does not take into account the three aforementioned factors. While the number of retrieved results might be argued correlated to the number of relevant items (thus, seemingly taking into account the first factor), this correlation can be very weak at times, e.g. consider a high frequency query word (almost a stop-word) which

would retrieve large parts of the collection. Further, such percentage thresholding seems remotely-connected to factors (2) and (3). Consequently, we will resort to the approach of [2] which, based on the distribution of item scores, is capable of estimating (1), as well as mapping scores to probabilities of relevance. Having the latter, (2) can be determined, and any measure defined in (3) can be optimized in a straightforward way. More on the method can be found in the last-cited study.

Targeting to enhance query generality, the most appropriate measure to optimize would be precision. However, since the *smoothed* precision estimated by the method of [2] monotonically declines with rank, it makes sense to set a precision threshold. The choice of precision threshold is dependent on the effectiveness of the CBIR stage: it can be seen as guaranteeing the minimum generality required by the CBIR method at hand for achieving good effectiveness. Not knowing the relation between CBIR effectiveness and minimum required generality, we will try a series of thresholds on precision, as well as, to optimize other cost-gain measures. Thus, while it may seem that we exchange the initial problem of where to set a static threshold with where to threshold precision or which measure to optimize, it will turn out that the latter problem is less sensitive to its available options, as we will see.

A possible drawback of the two-stage setup considered is that relevant images with empty or very noise secondary modalities would be completely missed, since they will not be retrieved by the first stage. If there are any improvements compared to single-stage text-only or image-only setups, these will first show up on early precision since only the top results are re-ranked; mean average precision or other measures may improve as a side effect. In any case, there are efficiency benefits from searching the most expensive modality only on a subset of the collection.

The requirement of such a two-stage CBIR at the user-side is that information needs are expressed by visual as well as textual descriptions. The community is already experimenting with such setups, e.g. the ImageCLEF 2010 Wikipedia Retrieval task was performed on a multimodal collection with topics made of textual and image queries at the same time [19]. Furthermore, multimodal or holistic query interfaces are showing up in experimental search engines allowing concurrent multimedia queries [21]. As a last resort, automatic image annotation methods [14,7] may be employed for generating queries for secondary modalities in traditional image retrieval systems.

### 3 An Experiment on Wikipedia

In this section, we report on experiments performed on a standardized multimodal snapshot of Wikipedia. It is worth noting that the collection is one of the largest benchmark image databases for today's standards. It is also highly heterogeneous, containing color natural images, graphics, grayscale images, etc., in a variety of sizes.

#### 3.1 Datasets, Systems, and Methods

The ImageCLEF 2010 Wikipedia test collection has image as its primary medium, consisting of 237,434 items, associated with noisy and incomplete user-supplied textual annotations and the Wikipedia articles containing the images. Associated annotations exist in any combination of English, German, French, or any other unidentified

(non-marked) language. There are 70 test topics, each one consisting of a textual and a visual part: three title fields (one per language—English, German, French), and one or more example images. The topics are assessed by visual similarity to the image examples. More details on the dataset can be found in [19].

For text indexing and retrieval, we employ the Lemur Toolkit V4.11 and Indri V2.11 with the tf.idf retrieval model.<sup>1</sup> We use the default settings that come with these versions of the system except that we enable Krovetz stemming. We index only the English annotations, and use only the English query of the topics.

We index the images with two descriptors that capture global image features: the Joint Composite Descriptor (JCD) and the Spatial Color Distribution (SpCD). The JCD is developed for color natural images and combines color and texture information [8]. In several benchmarking databases, JCD has been found more effective than MPEG-7 descriptors [8]. The SpCD combines color and its spatial distribution; it is considered more suitable for colored graphics since they consist of a relatively small number of colors and less texture regions than color natural images. It is recently introduced in [9] and found to perform better than JCD in a heterogeneous image database [10].

We evaluate on the top-1000 results with mean average precision (MAP), precision at 10 and 20, and bpref [6].

### 3.2 Thresholding and Re-ranking

We investigate two types of thresholding: static and dynamic. In static thresholding, the same fixed pre-selected rank threshold  $K$  is applied to all topics. We experiment with levels of  $K$  at 25, 50, 100, 250, 500, and 1000. The results that are not re-ranked by image are retained as they are ranked by text, also in dynamic thresholding.

For dynamic thresholding, we use the Score-Distributional Threshold Optimization (SDTO) as described in [2] and with the code provided by its authors. For tf.idf scores, we used the *technically truncated* model of a normal-exponential mixture. The method normalizes retrieval scores to probabilities of relevance (prels), enabling the optimization of  $K$  for any user-defined effectiveness measure. Per query, we search for the optimal  $K$  in  $[0, 2500]$ , where 0 or 1 results to no re-ranking. Thus, for estimation with the SDTO we truncate at the score corresponding to rank 2500 but use no truncation at high scores as tf.idf has no theoretical maximum. If there are 25 text results or less, we always re-rank by image; these are too few scores to apply the SDTO reliably. In this category fall the topics 1, 10, 23, and 46, with only 18, 16, 2, and 18 text results respectively. The biggest strength of the SDTO is that it does not require training data; more details on the method can be found in the last-mentioned study.

We experiment with the SDTO by thresholding on prel as well as on precision. Thresholding on fixed prels happens to optimize *linear utility measures* [13], with corresponding rank thresholds:

- $\max K: P(\text{rel}|D_K) > \theta$ , where  $D_K$  is the  $K$ th ranked document. For the prel threshold  $\theta$ , we try six values. Two of them are:
  - $\theta = 0.5000$ : It corresponds to 1 loss per relevant non-retrieved and 1 loss per non-relevant retrieved, i.e. the Error Rate, and it is precision-recall balanced.

<sup>1</sup> <http://www.lemurproject.org>

- $\theta = 0.3333$ : It corresponds to 2 gain per relevant retrieved and 1 loss per non-relevant retrieved, i.e. the T9U measure used in the TREC 2000 Filtering Track [20], and it is recall-oriented.

These prel thresholds may optimize other measures as well; for example, 0.5000 optimizes also the utility measure of 1 gain per relevant retrieved and 1 loss per non-relevant retrieved. Thus, irrespective of which measure prel thresholds optimize, we arbitrarily enrich the experimental set of levels with four more thresholds: 0.9900, 0.9500, 0.8000, and 0.1000.

Furthermore, having normalized scores to prels, we can estimate precision in any top- $K$  set by simply adding the prels and dividing by  $K$ . The estimated precision can be seen as the generality in the sub-ranking. According to the hypothesis that the effectiveness of CBIR is positively correlated to query generality, we experiment with the following thresholding:

- max  $K$ :  $\text{Prec}@K > g$ , where for  $g$  is the minimum generality required by the CBIR at hand for good effectiveness. Having no clue on usable  $g$  values, we arbitrarily try levels of  $g$  at 0.9900, 0.9500, 0.8000, 0.5000, 0.3333, and 0.1000.

### 3.3 Setting the Baseline

In initial experiments, we investigated the effectiveness of each of the stages individually, trying to tune them for best results.

In the textual stage, we employ the tf.idf model since it has been found to work well with the SDTO [3]. The SDTO method fits a binary mixture of probability distributions on the score distribution (SD). A previous study suggested that while long queries tend to lead to smoother SDs and improved fits, threshold predictions are better for short queries of high quality keywords [3]. To be on the safe side, in initial experiments we tried to increase query length by enabling pseudo relevance feedback of the top-10 documents, but all our combinations of the parameter values for the number of feedback terms and initial query weight led to significant decreases in the effectiveness of text retrieval. We attribute this to the noisy nature of the annotations. Consequently, we do not run any two-stage experiments with pseudo relevance feedback at the first textual stage.

In the visual stage, first we tried the JCD alone, as the collection seems to contain more color natural images than graphics, and used only the first example image; this represents a simple but practically realistic setup. Then, incorporating all example images, the natural combination is to assign to each collection image the maximum similarity seen from its comparisons to all example images; this can be interpreted as looking for images similar to *any* of the example images. Last, assuming that the SpCD descriptor captures orthogonal information to JCD, we added its contribution. We did not normalize the similarity values prior to combining them, as these descriptors produce comparable similarity distributions [10]. Table 1 presents the results; the index  $i$  runs over example images.

The image-only runs perform far below the text-only run. This puts in perspective the quality of the currently effective global CBIR descriptors: their effectiveness in image retrieval is much worse than the effectiveness of the traditional tf.idf text retrieval model even on sparse and noisy annotations. Since the image-only runs would have provided

**Table 1.** Effectiveness of different CBIR setups against tf.idf text-only retrieval

item scoring by	MAP	P@10	P@20	bpref
$JCD_1$	.0058	.0486	.0479	.0352
$\max_i JCD_i$	.0072	.0614	.0614	.0387
$\max_i JCD_i + \max_i SpCD_i$	.0112	.0871	.0886	.0415
tf.idf (text-only)	.1293	.3614	.3314	.1806

very weak baselines, we choose as a much stronger baseline for statistical significance testing the text-only run. This makes sense also from an efficiency point of view: if using a secondary text modality for image retrieval is more effective than current CBIR methods, then there is no reason at all for using computationally costly CBIR methods.

Comparing the image-only runs to each other, we see that using more information—either from more example images or more descriptors—improves effectiveness. In order to investigate the impact of the effectiveness level of the second stage on the whole two-stage procedure, we will present two-stage results for both the best and the worst CBIR methods.

### 3.4 Experimental Results

Table 2 presents two-stage image retrieval results against text- and image-only retrieval. It is easy to see that the dynamic thresholding methods improve retrieval effectiveness in most of the experiments. Especially, dynamical thresholding using  $\theta$  shows improvements for all values we tried. The greatest improvement (+28%) is observed in P@10 for  $\theta = 0.8$ . The table contains lots of numbers; while there may be consistent increases or decreases in some places, in the rest of this section we focus and summarize only the statistically significant differences.

Irrespective of measure and CBIR method, the best thresholds are roughly at: 25 or 50 for  $K$ , 0.95 for  $g$ , and 0.8 for  $\theta$ . The weakest thresholding method is the static  $K$ : there are very few improvements only in P@20 at tight cutoffs, but they are accompanied by a reduced MAP and bpref. Actually, static thresholds hurt MAP and/or bpref almost anywhere. Effectiveness degrades also in early precision for  $K = 1000$ . Dynamic thresholding is much more robust. Comparing the two CBIR methods at the second stage, the stronger method helps the dynamic methods considerably while static thresholding does not seem to receive much improvement.

Concerning the dynamic thresholding methods, the probability thresholds  $\theta$  correspond to tighter *effective* rank thresholds than these of the precision thresholds  $g$ , for  $g$  and  $\theta$  taking values in the range  $[0.1000, 0.9900]$ . As a proxy for the effective  $K$  we use the median threshold  $\tilde{K}$  across all topics. This is expected since precision declines slower than *prel*. Nevertheless, the fact that a wide range of *prel* thresholds results to a tight range of  $\tilde{K}$ , reveals a sharp decline in *prel* below some score per query. This makes the end-effectiveness less sensitive to *prel* thresholds in comparison to precision thresholds, thus more robust against possibly unsuitable user-selected values. Furthermore, if we compare the dynamic methods at similar  $\tilde{K}$ , e.g.  $g = 0.9900$  to  $\theta = 0.9500$  ( $\tilde{K} \approx 50$ ) and  $g = 0.8000$  to  $\theta = 0.5000$  ( $\tilde{K} \approx 93$ ), we see that *prel* thresholds perform slightly better. Figure 1 depicts the evaluation measures against  $\tilde{K}$  for all methods and the stronger CBIR; Figure 2 presents the top image results for a query.

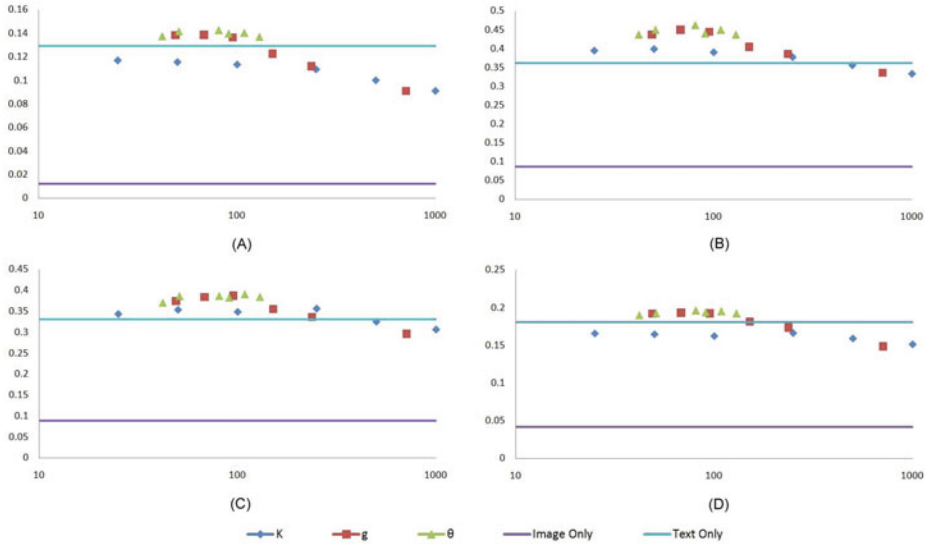
**Table 2.** Two-stage image retrieval results. The best results per measure and thresholding type are in boldface. Significance-tested with a bootstrap test, one-tailed, at significance levels 0.05 ( $^{\Delta\vee}$ ), 0.01 ( $^{\Delta\vee}$ ), and 0.001 ( $^{\Delta\vee}$ ), against the text-only baseline.

threshold	$\tilde{K}$	JCD <sub>1</sub>				max <sub>i</sub> JCD <sub>i</sub> + max <sub>i</sub> SpCD <sub>i</sub>				
		MAP	P@10	P@20	bpref	MAP	P@10	P@20	bpref	
text-only	—	.1293	.3614	.3314	.1806	.1293	.3614	.3314	.1806	
K	25	<b>.1162<sup>∇</sup></b>	<b>.3957<sup>∇</sup></b>	.3457 <sup>Δ</sup>	.1641 <sup>∇</sup>	<b>.1168<sup>∇</sup></b>	.3943 <sup>∇</sup>	.3436 <sup>Δ</sup>	.1659 <sup>∇</sup>	
	50	.1144 <sup>∇</sup>	.3829 <sup>∇</sup>	<b>.3579<sup>Δ</sup></b>	.1608 <sup>∇</sup>	.1154 <sup>∇</sup>	<b>.3986<sup>∇</sup></b>	.3557 <sup>∇</sup>	.1648 <sup>∇</sup>	
	100	.1138 <sup>∇</sup>	.3786 <sup>∇</sup>	.3471 <sup>∇</sup>	.1609 <sup>∇</sup>	.1133 <sup>∇</sup>	.3900 <sup>∇</sup>	.3486 <sup>∇</sup>	.1623 <sup>∇</sup>	
	250	.1081 <sup>∇</sup>	.3414 <sup>∇</sup>	.3164 <sup>∇</sup>	<b>.1644<sup>∇</sup></b>	.1092 <sup>∇</sup>	.3771 <sup>∇</sup>	<b>.3564<sup>∇</sup></b>	<b>.1664<sup>∇</sup></b>	
	500	.0968 <sup>∇</sup>	.3200 <sup>∇</sup>	.3007 <sup>∇</sup>	.1575 <sup>∇</sup>	.0999 <sup>∇</sup>	.3557 <sup>∇</sup>	.3250 <sup>∇</sup>	.1590 <sup>∇</sup>	
	1000	.0865 <sup>∇</sup>	.2871 <sup>∇</sup>	.2729 <sup>∇</sup>	.1493 <sup>∇</sup>	.0909 <sup>∇</sup>	.3329 <sup>∇</sup>	.3064 <sup>∇</sup>	.1511 <sup>∇</sup>	
g	.9900	49	<b>.1364<sup>∇</sup></b>	<b>.4214<sup>Δ</sup></b>	.3550 <sup>∇</sup>	.1902 <sup>Δ</sup>	.1385 <sup>Δ</sup>	.4371 <sup>Δ</sup>	.3743 <sup>Δ</sup>	.1921 <sup>Δ</sup>
	.9500	68	.1352 <sup>∇</sup>	.4171 <sup>Δ</sup>	<b>.3586<sup>∇</sup></b>	<b>.1912<sup>Δ</sup></b>	<b>.1386<sup>Δ</sup></b>	<b>.4500<sup>Δ</sup></b>	.3836 <sup>Δ</sup>	<b>.1932<sup>Δ</sup></b>
	.8000	95	.1318 <sup>∇</sup>	.4000 <sup>∇</sup>	.3536 <sup>∇</sup>	.1892 <sup>∇</sup>	.1365 <sup>∇</sup>	.4443 <sup>Δ</sup>	<b>.3871<sup>Δ</sup></b>	.1924 <sup>∇</sup>
	.5000	151	.1196 <sup>∇</sup>	.3814 <sup>∇</sup>	.3393 <sup>∇</sup>	.1808 <sup>∇</sup>	.1226 <sup>∇</sup>	.4043 <sup>∇</sup>	.3550 <sup>∇</sup>	.1813 <sup>∇</sup>
	.3333	237	.1085 <sup>∇</sup>	.3500 <sup>∇</sup>	.3000 <sup>∇</sup>	.1707 <sup>∇</sup>	.1121 <sup>∇</sup>	.3857 <sup>∇</sup>	.3364 <sup>∇</sup>	.1734 <sup>∇</sup>
	.1000	711	.0864 <sup>∇</sup>	.2871 <sup>∇</sup>	.2621 <sup>∇</sup>	.1461 <sup>∇</sup>	.0909 <sup>∇</sup>	.3357 <sup>∇</sup>	.2964 <sup>∇</sup>	.1487 <sup>∇</sup>
θ	.9900	42	.1342 <sup>∇</sup>	.4043 <sup>∇</sup>	.3414 <sup>∇</sup>	.1865 <sup>∇</sup>	.1375 <sup>Δ</sup>	.4371 <sup>Δ</sup>	.3700 <sup>Δ</sup>	.1897 <sup>Δ</sup>
	.9500	51	.1371 <sup>∇</sup>	.4214 <sup>Δ</sup>	.3586 <sup>∇</sup>	.1903 <sup>Δ</sup>	.1417 <sup>Δ</sup>	.4500 <sup>Δ</sup>	.3864 <sup>Δ</sup>	.1924 <sup>Δ</sup>
	.8000	81	<b>.1384<sup>Δ</sup></b>	<b>.4229<sup>Δ</sup></b>	.3614 <sup>∇</sup>	.1921 <sup>Δ</sup>	<b>.1427<sup>Δ</sup></b>	<b>.4629<sup>Δ</sup></b>	.3871 <sup>Δ</sup>	<b>.1961<sup>Δ</sup></b>
	.5000	91	.1367 <sup>∇</sup>	.4057 <sup>∇</sup>	.3571 <sup>∇</sup>	.1919 <sup>Δ</sup>	.1397 <sup>Δ</sup>	.4400 <sup>Δ</sup>	.3829 <sup>Δ</sup>	.1937 <sup>Δ</sup>
	.3333	109	.1375 <sup>∇</sup>	.4129 <sup>∇</sup>	<b>.3636<sup>Δ</sup></b>	<b>.1933<sup>Δ</sup></b>	.1404 <sup>Δ</sup>	.4500 <sup>Δ</sup>	<b>.3907<sup>Δ</sup></b>	.1949 <sup>Δ</sup>
	.1000	130	.1314 <sup>∇</sup>	.4100 <sup>∇</sup>	.3629 <sup>∇</sup>	.1866 <sup>∇</sup>	.1370 <sup>∇</sup>	.4371 <sup>Δ</sup>	.3843 <sup>Δ</sup>	.1922 <sup>Δ</sup>
image-only	—	.0058 <sup>∇</sup>	.0486 <sup>∇</sup>	.0479 <sup>∇</sup>	.0352 <sup>∇</sup>	.0112 <sup>∇</sup>	.0871 <sup>∇</sup>	.0886 <sup>∇</sup>	.0415 <sup>∇</sup>	

In summary, static thresholding improves initial precision at the cost of MAP and bpref, while dynamic thresholding on precision or prel does not have this drawback. The choice of a static or precision threshold influences greatly the effectiveness, and unsuitable choices (e.g. too loose) may lead to a degraded performance. Prel thresholds are much more robust in this respect. As expected, better CBIR at the second stage leads to overall improvements, nevertheless, the thresholding type seems more important: While the two CBIR methods we employ vary greatly in performance (the best has almost double the effectiveness of the other), static thresholding is not influenced much by this choice; we attribute this to its lack of respect for the number of relevant items and for the ranking quality. Dynamic methods benefit more from improved CBIR. Overall, prel thresholds perform best, for a wide range of values.

## 4 Related Work

Image re-ranking can be performed using textual, e.g. [11], or visual descriptions. Next, we will focus only on visual re-ranking. Subset re-ranking by visual content has been seen before, but mostly in different setups than the one we consider or for different purposes, e.g. result clustering or diversity. It is worth mentioning that all the previously proposed methods we review below used global image features to re-rank images.

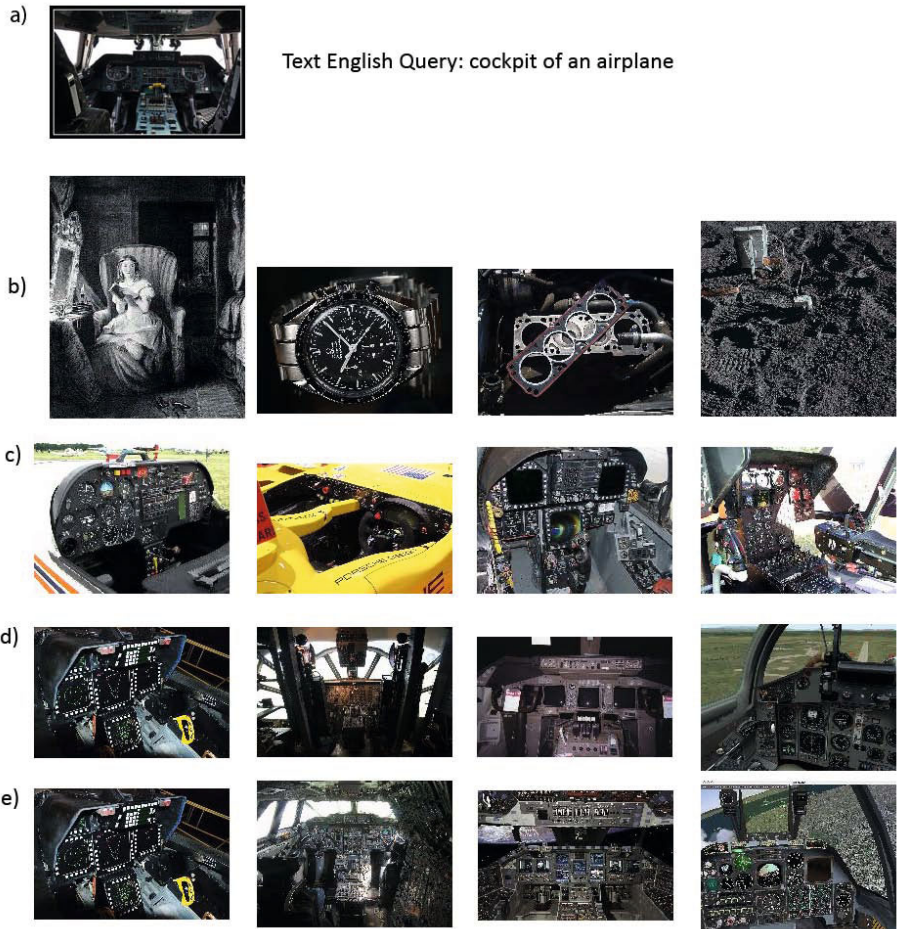


**Fig. 1.** Effectiveness, for the strongest CBIR stage: (A) MAP, (B) P@10, (C) P@20, (D) bpref

For example, [4] proposed an image retrieval system using keyword-based retrieval of images via their annotations, followed by clustering of the top-150 results returned by Google Images according to their visual similarity. Using the clusters, retrieved images were arranged in such a way that visually similar images are positioned close to each other. Although the method may have had a similar effect to ours, it was not evaluated against text-only or image-only baselines, and the impact of different values of  $K$  was not investigated. In [12], the authors retrieved the top-50 results by text and then clustered the images in order to obtain a diverse ranking based on cluster representatives. The clusters were evaluated against manually-clustered results, and it was found that the proposed clustering methods tend to reproduce manual clustering in the majority of cases. The approach we have taken does not target to increasing diversity.

Another similar approach was proposed in [18], where the authors state that Web image retrieval by text queries is often noisy and employ image processing techniques in order to re-rank retrieved images. The re-ranking technique was based on the visual similarity between image search results and on their dissimilarity to an external contrastive class of diversified images. The basic idea is that an image will be relevant to the query, if it is visually similar to other query results and dissimilar to the external class. To determine the visual coherence of a class, they took the top 30% of retrieved images and computed the average number of neighbors to the external class. The effects of the re-ranking were analyzed via a user-study with 22 participants. Visual re-ranking seemed to be preferred over the plain keyword-based approach by a large majority of the users. Note that they did not use an image query but only a text one; in this respect, the setup we have considered differs in that image queries are central, and we do not require external information.





**Fig. 2.** Retrieval results: (a) query, (b) image-only, (c) text-only, (d)  $K = 25$ , (e)  $\theta = 0.8$

In [17], the authors proposed also a two-stage image retrieval system with external information requirements: the first stage is text-based with automatic query expansion, whereas the second exploits the visual properties of the query to improve the results of the text search. In order to visually re-rank the top-1000 images, they employed a visual model (a set of images which depicts each topic) using Web images. To describe the visual content of the images, several methods using global or local features were employed. Experimental results demonstrated that visual re-ranking improves the retrieval performance significantly in MAP, P@10 and P@20. We have confirmed that visual re-ranking of top-ranked results improves early precision, though with a simpler setup without using external information.

Some other similar setups to the one we propose are these in [5] and [16]. In [5], the authors trained their system to perform automatic re-ranking on all results returned by text retrieval. The re-ranking method considered several aspects of both document and query (e.g. generality of the textual features, color amount from the visual features). Improved results were obtained only when the training set had been derived from the database which is searched. Our method re-ranks the results using only visual features; it does not require training and can be applied to any database. In [16], the authors re-rank the top- $K$  results retrieved by text using visual information. The rank thresholds of 60 and 300 were tried and both resulted to a decrease in mean average precision compared to the text-only baseline, with the 300 performing worse. Our experiments have confirmed their result: static thresholds degrade MAP. They did not report early precision figures.

## 5 Conclusions and Directions for Further Research

We have experimented with two-stage image retrieval from a large multimodal database, by first using a text modality to rank the collection and then perform content-based image retrieval only on the top- $K$  items. In view of previous literature, the biggest novelty of our method is that re-ranking is not applied to a preset number of top- $K$  results, but  $K$  is calculated dynamically per query to optimize a predefined effectiveness measure. Additionally, the proposed method does not require any external information or training data. The choice between static or dynamic nature of rank-thresholds has turned out to make the difference between failure and success of the two-stage setup.

We have found that two-stage retrieval with dynamic thresholding is more effective and robust than static thresholding, practically insensitive to a wide range of reasonable choices for the measure under optimization, and beats significantly the text-only and several image-only baselines. A two-stage approach, irrespective of thresholding type, has also an obvious efficiency benefit: it cuts down greatly on expensive image operations. Although we have not measured running times, only the 0.02–0.05% of the items (on average) had to be scored at the expensive image stage for effective retrieval from the collection at hand. While for the dynamic method there is some overhead for estimating thresholds, this offsets only a small part of the efficiency gains.

There are a couple of interesting directions to pursue in the future. First, the idea can be generalized to *multi-stage* retrieval for multimodal databases, where rankings for the modalities are successively being thresholded and re-ranked according to a modality hierarchy. Second, although in Section 2 we merely argued on the unsuitability of fusion under the assumptions of the setup we considered, a future plan is to compare the effectiveness of two-stage against fusion. Irrespective of the outcome, fusion does not have the efficiency benefits of two-stage retrieval.

## Acknowledgments

We thank Jaap Kamps for providing the code for the statistical significance testing.

## References

1. Aly, M., Welinder, P., Munich, M.E., Perona, P.: Automatic discovery of image families: global vs. local features. In: *ICIP*, pp. 777–780. IEEE, Los Alamitos (2009)
2. Arampatzis, A., Kamps, J., Robertson, S.: Where to stop reading a ranked list: threshold optimization using truncated score distributions. In: *SIGIR*, pp. 524–531. ACM, New York (2009)
3. Arampatzis, A., Robertson, S., Kamps, J.: Score distributions in information retrieval. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) *ICTIR 2009*. LNCS, vol. 5766, pp. 139–151. Springer, Heidelberg (2009)
4. Barthel, K.U.: Improved image retrieval using automatic image sorting and semi-automatic generation of image semantics. In: *International Workshop on Image Analysis for Multimedia Interactive Services*, pp. 227–230 (2008)
5. Berber, T., Alpkocak, A.: DEU at ImageCLEFMed 2009: Evaluating re-ranking and integrated retrieval systems. In: *CLEF Working Notes (2009)*
6. Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: *SIGIR*, pp. 25–32. ACM, New York (2004)
7. Chang, E., Goh, K., Sychay, G., Wu, G.: CBSA: content-based soft annotation for multimodal image retrieval using bayes point machines. *IEEE Transactions on Circuits and Systems for Video Technology* 13(1), 26–38 (2003)
8. Chatzichristofis, S.A., Boutalis, Y.S., Lux, M.: Selection of the proper compact composite descriptor for improving content-based image retrieval. In: *SPPRA*, pp. 134–140 (2009)
9. Chatzichristofis, S.A., Boutalis, Y.S., Lux, M.: SpCD—spatial color distribution descriptor. A fuzzy rule based compact composite descriptor appropriate for hand drawn color sketches retrieval. In: *ICAART*, pp. 58–63 (2010)
10. Chatzichristofis, S.A., Arampatzis, A.: Late fusion of compact composite descriptors for retrieval from heterogeneous image databases. In: *SIGIR*, pp. 825–826. ACM, New York (2010)
11. Kilinc, D., Alpkocak, A.: Deu at imageclef 2009 wikipediainm task: Experiments with expansion and reranking approaches. In: *Working Notes of CLEF (2009)*
12. van Leuken, R.H., Pueyo, L.G., Olivares, X., van Zwol, R.: Visual diversification of image search results. In: *WWW*, pp. 341–350. ACM, New York (2009)
13. Lewis, D.D.: Evaluating and optimizing autonomous text classification systems. In: *SIGIR*, pp. 246–254. ACM Press, New York (1995)
14. Li, J., Wang, J.Z.: Real-time computerized annotation of pictures. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 985–1002 (2008)
15. Li, X., Chen, L., Zhang, L., Lin, F., Ma, W.Y.: Image annotation by large-scale content-based image retrieval. In: *ACM Multimedia*, pp. 607–610. ACM, New York (2006)
16. Maillot, N., Chevillet, J.P., Lim, J.H.: Inter-media pseudo-relevance feedback application to imageclef 2006 photo retrieval. In: *CLEF Working Notes (2006)*
17. Myoupo, D., Popescu, A., Le Borgne, H., Moellic, P.: Multimodal image retrieval over a large database. In: Peters, C., Caputo, B., Gonzalo, J., Jones, G.J.F., Kalpathy-Cramer, J., Müller, H., Tsirikika, T. (eds.) *CLEF 2009*. LNCS, vol. 6242, pp. 177–184. Springer, Heidelberg (2010)
18. Popescu, A., Moellic, P.A., Kanellos, I., Landais, R.: Lightweight web image reranking. In: *ACM Multimedia*, pp. 657–660. ACM, New York (2009)
19. Popescu, A., Tsirikika, T., Kludas, J.: Overview of the wikipedia retrieval task at imageclef 2010. In: *CLEF (Notebook Papers/LABS/Workshops) (2010)*
20. Robertson, S.E., Hull, D.A.: The TREC-9 filtering track final report. In: *TREC (2000)*
21. Zagoris, K., Arampatzis, A., Chatzichristofis, S.A.: www.mmretrieval.net: a multimodal search engine. In: *SISAP*, pp. 117–118. ACM, New York (2010)

# Comparing Twitter and Traditional Media Using Topic Models

Wayne Xin Zhao<sup>1</sup>, Jing Jiang<sup>2</sup>, Jianshu Weng<sup>2</sup>, Jing He<sup>1</sup>, Ee-Peng Lim<sup>2</sup>,  
Hongfei Yan<sup>1</sup>, and Xiaoming Li<sup>1</sup>

<sup>1</sup> Peking University, China

<sup>2</sup> Singapore Management University, Singapore

**Abstract.** Twitter as a new form of social media can potentially contain much useful information, but content analysis on Twitter has not been well studied. In particular, it is not clear whether as an information source Twitter can be simply regarded as a faster news feed that covers mostly the same information as traditional news media. In This paper we empirically compare the content of Twitter with a traditional news medium, New York Times, using unsupervised topic modeling. We use a Twitter-LDA model to discover topics from a representative sample of the entire Twitter. We then use text mining techniques to compare these Twitter topics with topics from New York Times, taking into consideration topic categories and types. We also study the relation between the proportions of opinionated tweets and retweets and topic categories and types. Our comparisons show interesting and useful findings for downstream IR or DM applications.

**Keywords:** Twitter, microblogging, topic modeling.

## 1 Introduction

Over the past few years, Twitter, a microblogging service, has become an increasingly popular platform for Web users to communicate with each other. Because tweets are compact and fast, Twitter has become widely used to spread and share breaking news, personal updates and spontaneous ideas. The popularity of this new form of social media has also started to attract the attention of researchers. Several recent studies examined Twitter from different perspectives, including the topological characteristics of Twitter [1], tweets as social sensors of real-time events [2], the forecast of box-office revenues for movies [3], etc. However, the explorations are still in an early stage and our understanding of Twitter, especially its large textual content, still remains limited.

Due to the nature of microblogging, the large amount of text in Twitter may presumably contain useful information that can hardly be found in traditional information sources. To make use of Twitter's textual content for information retrieval tasks such as search and recommendation, one of the first questions one may ask is what kind of special or unique information is contained in Twitter. As Twitter is often used to spread breaking news, a particularly important question

is how the information contained in Twitter differs from what one can obtain from other more traditional media such as newspapers. Knowing this difference could enable us to better define retrieval tasks and design retrieval models on Twitter and in general microblogs.

To the best of our knowledge, very few studies have been devoted to content analysis of Twitter, and none has carried out deep content comparison of Twitter with traditional news media. In this work we perform content analysis through topic modeling on a representative sample of Twitter within a three-month time span, and we empirically compare the content of Twitter based on the discovered topics with that of news articles from a traditional news agency, namely, New York Times, within the same time span. Specifically we try to answer the following research questions:

- Does Twitter cover similar categories and types of topics as traditional news media? Do the distributions of topic categories and types differ in Twitter and in traditional news media?
- Are there specific topics covered in Twitter but rarely covered in traditional news media and vice versa? If so, are there common characteristics of these specific topics?
- Do certain categories and types of topics attract more opinions in Twitter?
- Do certain categories and types of topics trigger more information spread in Twitter?

Some of our major findings are the following: (1) Twitter and traditional news media cover a similar range of topic categories, but the distributions of different topic categories and types differ between Twitter and traditional news media. (2) As expected, Twitter users tweet more on personal life and pop culture than world events. (3) Twitter covers more celebrities and brands that may not be covered in traditional media. (4) Although Twitter users tweet less on world events, they do actively *retweet* (forward) world event topics, which helps spread important news.

These findings can potential benefit many Web information retrieval applications. For example, for Web information retrieval and recommendation, our findings suggest that Twitter is a valuable source for entertainment and lifestyle topics such as celebrities and brands to complement traditional information sources. Retweets can also be used to indicate trendy topics among Web users to help search engines refine their results.

## 2 Data Preparation

We use a sample of the Edinburgh Twitter Corpus [4] as our Twitter dataset. The original corpus was collected through Twitter’s streaming API and is thus a representative sample of the entire Twitter stream. It covers a time span from November 11, 2009 to February 1, 2010.

In order to obtain a parallel news corpus that represents the traditional news media, we chose New York Times (NYT) as our source of news articles. We

**Table 1.** Some statistics of the Twitter and the NYT data sets after preprocessing

Collection	Docs	Users	Words	Vocabulary
Twitter	1,225,851	4,916	8,152,138	21,448
NYT	11,924	–	4,274,404	26,994

crawled news articles dating from November 11, 2009 until February 1, 2010 through NYT’s search page<sup>1</sup>.

For both the Twitter and the NYT collections, we first removed all the stop words. We then removed words with a document frequency less than 10 and words that occurred in more than 70% of the tweets (news articles) in the Twitter (NYT) collection. For Twitter data, we further removed tweets with fewer than three words and all the users with fewer than 8 tweets. Some statistics of the two datasets after preprocessing are summarized in Table 1.

### 3 Topic Discovery and Classification

To compare the content of Twitter and New York Times, we first introduce three major concepts used in this paper.

**Definition 1.** A **topic** is a subject discussed in one or more documents. Examples of topics include news events such as “the Haiti earthquake,” entities such as “Michael Jackson” and long-standing subjects such as “global warming.” Each topic is assumed to be represented by a multinomial distribution of words.

**Definition 2.** A **topic category** groups topics belonging to a common subject area together. We adopt the topic categories defined in New York Times<sup>2</sup> with some modifications. See Figure 3 for the full set of topic categories we use.

**Definition 3.** A **topic type** characterizes the nature of a topic. After examining some topics from both Twitter and New York Times, we define three topic types, namely, event-oriented topics, entity-oriented topics and long-standing topics.

Note that topic categories and topic types are two orthogonal concepts. We assume that each topic can be assigned to a topic category and has a topic type. We use fully automatic methods to discover topics from each data collection first. We then use semi-automatic methods to assign the topics to the predefined topic categories as well as to remove noisy background topics. Finally we manually label the topics with topic types.

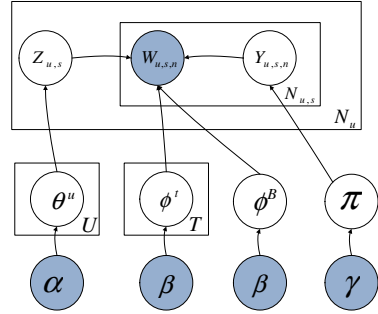
#### 3.1 Topic Discovery

**New York Times.** To discover topics from NYT, we choose to directly apply Latent Dirichlet Allocation (LDA) [5]. Our experiments show that we can obtain meaningful topics from the NYT data set using standard LDA. We set the

<sup>1</sup> <http://query.nytimes.com/search/>

<sup>2</sup> As of July 5, 2010.

1. Draw  $\phi^B \sim \text{Dir}(\beta), \pi \sim \text{Dir}(\gamma)$
2. For each topic  $t = 1, \dots, T$ ,
  - (a) draw  $\phi^t \sim \text{Dir}(\beta)$
3. For each user  $u = 1, \dots, U$ ,
  - (a) draw  $\theta^u \sim \text{Dir}(\alpha)$
  - (b) for each tweet  $s = 1, \dots, N_u$ 
    - i. draw  $z_{u,s} \sim \text{Multi}(\theta^u)$
    - ii. for each word  $n = 1, \dots, N_{u,s}$ 
      - A. draw  $y_{u,s,n} \sim \text{Multi}(\pi)$
      - B. draw  $w_{u,s,n} \sim \text{Multi}(\phi^B)$  if  $y_{u,s,n} = 0$  and  $w_{u,s,n} \sim \text{Multi}(\phi^{z_{u,s}})$  if  $y_{u,s,n} = 1$



**Fig. 1.** The generation process of tweets **Fig. 2.** Plate notation of our Twitter-LDA

number of topics to 100 and ran 1000 iterations of Gibbs sampling using the GibbsLDA++ toolkit<sup>3</sup>. We use  $\mathcal{T}_{\text{nyt}}$  to denote the set of topics we obtained from NYT.

**Twitter.** Standard LDA may not work well with Twitter because tweets are short. To overcome this difficulty, some previous studies proposed to aggregate all the tweets of a user as a single document [6,7]. In fact this treatment can be regarded as an application of the author-topic model [8] to tweets, where each document (tweet) has a single author. However, this treatment does not exploit the following important observation: A single tweet is usually about a single topic. We therefore propose a different Twitter-LDA model.

Formally, we assume that there are  $T$  topics in Twitter, each represented by a word distribution. Let  $\phi^t$  denote the word distribution for topic  $t$  and  $\phi^B$  the word distribution for background words. Let  $\theta^u$  denote the topic distribution of user  $u$ . Let  $\pi$  denote a Bernoulli distribution that governs the choice between background words and topic words. When writing a tweet, a user first chooses a topic based on her topic distribution. Then she chooses a bag of words one by one based on the chosen topic or the background model. The generation process of tweets is described in Figure 1 and illustrated in Figure 2. Each multinomial distribution is governed by some symmetric Dirichlet distribution. We use Gibbs sampling to perform model inference. Due to the space limit we leave out the derivation details and the sampling formulas.

We quantitatively evaluated the effectiveness of our Twitter-LDA model compared with standard LDA model (i.e. treating each tweet as a single document) and the author-topic model (i.e. treating all tweets of the same user as a single document). We set  $T$  to 110 (based on preliminary experiments) for these two baselines and our Twitter-LDA. We then randomly mixed the 330 topics from the three models. We asked two human judges to assign a score to each topic

<sup>3</sup> <http://gibbslda.sourceforge.net/>

**Table 2.** Comparison between Twitter-LDA, author-topic model and standard LDA

Method	Avg. Score	Agreement between Judges	Cohen’s Kappa
Twitter-LDA	<b>0.675</b>	65.5%	0.433
Author-Topic	0.539	54.5%	0.323
Standard LDA	0.509	70.9%	0.552

according to the following guidelines based on the top-10 topic words and took their average as the score for each topic: 1 (meaningful and coherent), 0.5 (containing multiple topics or noisy words), 0 (making no sense). The average scores of topics discovered by each method are shown in Table 2 together with the annotation agreement information. We can see that the Twitter-LDA model clearly outperformed the other two models, giving more meaningful top topic words, indicating that our Twitter-LDA model is a better choice than standard LDA for discovering topics from Twitter.

### 3.2 Categorizing Topics

**New York Times.** For the NYT dataset, because the articles already have category labels, intuitively, if a topic is associated with many articles in a particular category, the topic is likely to belong to that category. To capture this intuition, we categorize topics by assigning topic  $t$  to category  $q^*$  where  $q^* = \arg \max_q p(q|t) = \arg \max_q p(t|q)p(q)/p(t) = \arg \max_q p(t|q)$ , assuming that all categories are equally important. We can estimate the probability of topic  $t$  given category  $q$  as

$$p(t|q) = \frac{\sum_{d \in \mathcal{D}_{\text{NYT},q}} \tilde{p}(t|d)}{|\mathcal{D}_{\text{NYT},q}|}, \quad (1)$$

where  $\tilde{p}(t|d)$  denotes the learned probability of topic  $t$  given document  $d$  and  $\mathcal{D}_{\text{NYT},q}$  denote the subset of documents in the NYT collection that are labeled with category  $q$ .

To further remove noisy topics (e.g. topics with incoherent words.) or background topics (e.g. topics consisting mainly of common words such as “called,” “made,” “added,” etc.), we exploit the following observation: Most meaningful topics are related to a single topic category. If a topic is closely related to many categories, it is likely a noisy or background topic. We therefore define a measure called *category entropy* ( $CE$ ) as follows:

$$CE(t) = - \sum_{q \in \mathcal{Q}} p(q|t) \log p(q|t). \quad (2)$$

The larger  $CE(t)$  is, the more likely  $t$  is a noisy or background topic. We remove topics whose  $CE(t)$  is larger than a threshold (empirically set to 3.41). After removing noisy and background topics, we obtain 83 topics from  $\mathcal{T}_{\text{nyt}}$  as the final set of NYT topics we use for our empirical comparison later.



**Table 3.** Statistics of topics in different types

Collection	Event-oriented	Entity-oriented	Long-standing
Twitter (81 topics)	7	19	55
NYT (83 topics)	20	9	54

**Twitter.** Unlike NYT documents, tweets do not naturally have category labels. We use the following strategy to categorize Twitter topics. For each Twitter topic we first find the most similar NYT topic. If it is similar enough to one of the NYT topics, we use that NYT topic’s category as the Twitter topic’s category. Otherwise, we manually assign it to one of the topic categories or remove it if it is a noisy topic. Specifically, to measure the similarity between a Twitter topic  $t$  and an NYT topic  $t'$ , we use JS-divergence between the two word distributions, denoted as  $p_t$  and  $p_{t'}$ :

$$\text{JS-div}(p_t||p_{t'}) = \frac{1}{2}\text{KL-div}(p_t||p_m) + \frac{1}{2}\text{KL-div}(p_{t'}||p_m),$$

where  $p_m(w) = \frac{1}{2}p_t(w) + \frac{1}{2}p_{t'}(w)$ , and KL-div is the KL-divergence. The JS-divergence has the advantage that it is symmetric. After the semi-automatic topic categorization, we obtain a set of 81 topics from Twitter to be used in later empirical comparison. In the future we will look into automatic methods for cleaning and categorizing Twitter topics.

### 3.3 Assigning Topic Types

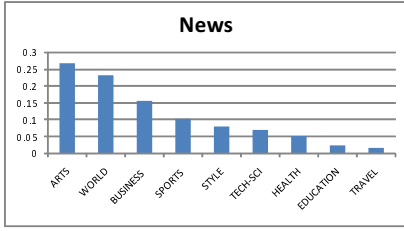
As we described earlier, we have defined three topic types, namely, *event-oriented* topics, *entity-oriented* topics and *long-standing* topics. Because these topic types are not based on semantic relatedness of topics, it is hard to automatically classify the topics into these topic types. We therefore manually classified the Twitter and the NYT topics into the three topic types. Some statistics of the topics in each type are shown in Table 3.

## 4 Empirical Comparison between Twitter and New York Times

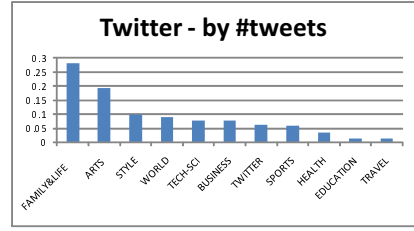
As we have stated, the focus of this study is to compare the content of Twitter with that of New York Times in order to understand the topical differences between Twitter and traditional news media and thus help make better use of Twitter as an information source. In this section we use the discovered topics from the two datasets together with their category and type information to perform an empirical comparison between Twitter and NYT.

### 4.1 Distribution of Topics

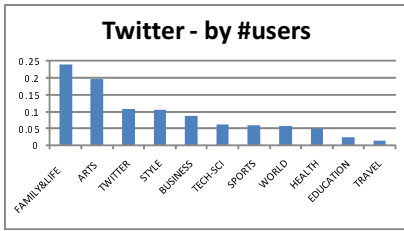
**By Topic Categories.** In traditional news media, while the categories of articles span a wide range from business to leisure, there is certainly an uneven



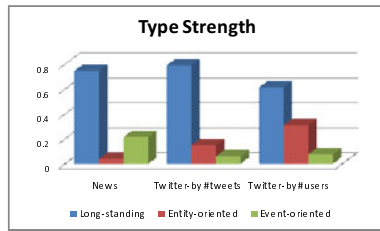
**Fig. 3.** Distribution of categories in NYT



**Fig. 4.** Distribution of categories (by #tweets) in Twitter



**Fig. 5.** Distribution of categories (by #users) in Twitter



**Fig. 6.** Distributions of topic types in the two data sets

distribution over these categories. In microblogging sites such as Twitter, where content is generated by ordinary Web users, how does the distribution of different categories of topics differ from traditional news media? To answer this question, we first compute the distributions of different topic categories in NYT and in Twitter respectively in the following way. For NYT, because we have the category labels of news articles, we measure the relative strength of a category simply by the percentage of articles belonging to that category. For Twitter, similarly, we can use the percentage of tweets belonging to each category as a measure of the strength of that category. With the help of the Twitter-LDA model, each tweet has been associated with a Twitter topic, and each Twitter topic is also assigned to a particular category as we have shown in Section 3.2. We also consider an alternative measure using the number of users interested in a topic category to gauge the strength of a category. Only users who have written at least five tweets belonging to that topic category are counted.

We plot the distributions of topic categories in the two datasets in Figure 3, Figure 4 and Figure 5. As we can see from the figures, both Twitter and NYT cover almost all categories. But the relative degrees of presence of different topic categories are quite different between Twitter and NYT. For example, in Twitter, *Family&Life* dominates while this category does not appear in NYT (because it is a new category we added for Twitter topics and therefore no NYT article is originally labeled with this category). *Arts* is commonly strong in both Twitter and NYT. However, *Style* is a strong category in Twitter but not so strong in NYT.

**Table 4.** Topics specific to NYT

Category	Topics
Arts	book,novel,story,life,writes world,century,history,culture art,museum,exhibition war,history,world,civil,time
Business	cars,car,ford,toyota,vehicle media,news,magazine,ads
Edu.	project,money,group,center percent,study,report,rate
Style	french,paris,luxury,swiss,watch
Tech-Sci	space,moon,station,spirit,earth
World	case,charges,prison,trial,court officials,announced,news,week department,agency,federal,law south,north,korea,korean, power

**Table 5.** Topics specific to Twitter

Category	Topics
Arts	rob,moon,love,twilight gaga,lady,#nowplaying adam,lambert,fans,kris chirs,brown,song,beyonce download,live,mixtape,music
Business	#ebay,auction,closing #jobs,job,#ukjobs
Family & Life	dog,room,house,cat,door good,night,hope,tonight life,#quote,success,change god,love,lord,heart,jesus smiles,laughs,hugs,kisses
Twitter	tweet, follow, account lmaoo,smh,jus,aint,lmaooo

**By Topic Types.** Similarly, we can compare the distributions of different topic types in Twitter and in NYT. We show the comparison in Figure 6. An interesting finding is that Twitter clearly has relatively more tweets and users talking about entity-oriented topics than NYT. In contrast, event-oriented topics are not so popular in Twitter although it has a much stronger presence than entity-oriented topics in NYT. We suspect that many entity-oriented topics are about celebrities and brands, and these tend to attract Web users' attention. To verify this, we inspected the entity-oriented topics in Twitter and found that indeed out of the 19 entity-oriented topics in Twitter 10 of them are on celebrities and the other 9 of them are on brands and big companies. Note that long-standing topics are always dominating. It may be surprising to see this for NYT, but it is partly because with LDA model each news article is assumed to have a mixture of topics. So even if a news article is mainly about an event, it may still have some fractions contributing to long-standing topics.

## 4.2 Breadth of Topic Coverage

Another kind of topic difference is the difference in the breadth of topic coverage. For example, for *Arts*, although both Twitter and NYT have strong presence of this category, we do not know whether they cover roughly the same set of topics. In this section, we first show topics that are covered extensively in Twitter (NYT) but not covered or covered very little in NYT (Twitter). We then try to characterize these topics by ranking topic categories and topic types by their breadth of topic coverage.

**Twitter-specific and NYT-specific Topics.** To identify topics present in one dataset but covered very little in the other dataset, we make use of the topic mapping method introduced in Section 3.2. Basically given a topic in Twitter

(NYT), we first find its most similar topic in NYT (Twitter) in the same category using the JS-divergence measure. If the divergence measure is above a certain threshold, meaning that the topic similarity is low, we decide that the topic is not covered in NYT (Twitter). Following Section 3.2 we use a threshold of 0.5 to find Twitter-specific topics and a threshold of 0.504 to find NYT-specific topics. (Both thresholds were set empirically.)

We show some sample specific topics in Table 4 and Table 5. Each topic is shown in one line and represented by a few keywords. First of all, we can see that Twitter-specific topics are concentrated in *Arts* and *Family&Life*. Because we have previously seen that the strength of *Family&Life* is much higher in Twitter than in NYT, it is not surprising to see that this category also has a broader topic coverage than NYT. However, it is interesting to see that although the *Arts* category does not show much difference in terms of relative strength or degree of presence in Twitter and in NYT, its topic coverage is quite different in Twitter and in NYT. In Twitter, there are many specific topics, especially entity-oriented topics such as “Lady Gaga” and “Chris Brown” that are not covered much in NYT. In NYT, there are also certain kinds of topics under *Arts* such as “museum” and “history” that are not covered much in Twitter. In retrospect, if we had separated out a *Pop Culture* category from *Arts*, we might have got different strengths of *Arts* in Twitter and in NYT. On the other hand, many NYT-specific topics are from the category *World*, which is similar to our findings from Section 4.1. It also indicates that news Web sites have broader reports on important events in detail, while due to the length restriction, Twitter tends to report breaking news in brief.

**Categories Ranked by Topic Coverage.** We would like to better characterize the differences of topic coverage of the two data sources in terms of topic categories and types. For topic categories, we would like to see which categories have relative smaller topic coverage in NYT compared with Twitter, and vice versa. To do so, we define the following *topic coverage divergence* (TC-div) measure, which measures the divergence of the topic coverage of one category in Twitter (NYT) with that in NYT (Twitter).

$$\begin{aligned}
 \text{TC-div}_{\text{Twitter}}(q) &= \frac{\sum_{t \in \mathcal{T}_{\text{Twitter},q}} \min_{t' \in \mathcal{T}_{\text{NYT},q}} \text{JS-div}(p_t || p_{t'})}{|\mathcal{T}_{\text{Twitter},q}|}, \\
 \text{TC-div}_{\text{NYT}}(q) &= \frac{\sum_{t \in \mathcal{T}_{\text{NYT},q}} \min_{t' \in \mathcal{T}_{\text{Twitter},q}} \text{JS-div}(p_t || p_{t'})}{|\mathcal{T}_{\text{NYT},q}|}.
 \end{aligned}$$

Here  $\mathcal{T}_{\text{Twitter},q}$  denotes the set of topics in Twitter and belonging to category  $q$ .

Based on this measure, we can rank the categories for Twitter and for NYT. Table 6 shows the ranking of categories. If a category is ranked high or has a large TC-div value in Twitter (NYT), it means there are many topics in this category that are covered well in Twitter (NYT) but not well in NYT (Twitter).

**Types Ranked by Topic Coverage.** Similarly, we can also rank the topic types by their topic coverage divergence measures. For both NYT and Twitter,

**Table 6.** Ranking of topic categories based on topic coverage divergence

Twitter	NYT
Arts	Education
Family&Life	Style
Business	Art
Travel	Travel
Tech-Sci	World
Health	Business
Education	Health
Style	Tech-Sci
World	Sports
Sports	—

**Table 7.** Opinion proportions of different categories in Twitter

Category	Opinion proportion
Family&Life	0.355
Education	0.294
Arts	0.289
Style	0.257
Twitter	0.242
Sports	0.226
Travel	0.198
Health	0.189
Business	0.186
Tech-Sci	0.151
World	0.097

**Table 8.** Retweet proportions of different categories in Twitter

Category	Retweet proportion
World	0.359264
Travel	0.22061
Tech-Sci	0.209646
Sports	0.187932
Twitter	0.182681
Style	0.170511
Arts	0.155924
Family&Life	0.141174
Health	0.155875
Business	0.11262
Education	0.082559

we have the ranking: *Entity-oriented* > *Long-standing* > *Event-oriented*. Event-oriented type has the smallest TC-div for both news and Twitter while entity-oriented type has the largest TC-div for both news and Twitter. It suggests that Twitter and NYT have more overlap of event-oriented topics but less overlap of entity-oriented topics. Also, event-oriented type has a smaller TC-div in Twitter than in NYT, suggesting that NYT covers event-related content of Twitter well but Twitter does not cover that of NYT quite well.

### 4.3 Opinions in Twitter

One characteristic of Twitter content compared with traditional news media is arguably the amount and coverage of user opinions expressed in tweets. We further study what categories and types of topics can generate a large number of opinionated tweets. We use a sentiment lexicon of 50 opinionated words<sup>4</sup> to identify opinionated tweets. We roughly estimate the proportions of tweets in each category that are opinionated by the number of tweets in each topic category or topic type that contain at least one of the opinion words. We show the results in Table 7. Interestingly, we can see that while the category *Education* is not a popular topic category in terms its total number of tweets, its proportion of opinionated tweets is ranked high, right after *Family&Life*. Categories such as *Tech-Sci*, *Business* and *World*, whose popularity in Twitter is in the mid-range, have been pushed down to the bottom in terms of their proportions of opinionated tweets. This change of ranking suggests that Twitter users tend to use Twitter to spread news in these categories rather than discuss their own opinions on news in these categories. On the other hand, more life and leisure-related topic categories such as *Style*, *Travel* and *Sports* tend to trigger more personal opinions.

<sup>4</sup> We manually went through our Twitter data and selected the top 50 opinion words based on our own judgment.

Similarly, we can do this with topic types. For opinion proportions, *Long-standing* > *Entity-oriented* > *Event-oriented*. As we can see, long-standing topics attract more opinionated tweets. It is interesting to see that entity-oriented topics attract relatively more opinions than event-oriented topics. This may be because many event-oriented topics actually also belong to the *World* and *Business* categories, while many entity-oriented topics are related to celebrities and brands, which are more closely related to life and leisure.

#### 4.4 Topic Spread Through Retweet

Another special property of Twitter is that it allows people to spread news through *retweet* messages. We further compute the proportions of retweet messages in each topic category and topic type by identifying the pattern `RT: @username`. From Table 8, we can see that the category *World* has the most retweet proportion among all categories. For Retweet proportion of topic types, we get *Entity-oriented* > *Long-standing* > *Event-oriented*. Event-oriented type has the most retweet proportion among all types. This makes sense because many topics in the *World* category also belong to event-oriented topic type, e.g., topics on breaking-news such as “Haiti earthquake.” This observation is interesting because although our previous analysis has shown that the strength and breadth of topic coverage of *World* topics in Twitter is low, we do see that Twitter users most actively spread *World* topics than other topics. It shows that retweeting is an important way for dissemination of significant events.

### 5 Related Work

Recently Twitter has attracted much attention in the research community, e.g. [6,11]. Our work is quite different from many pioneering studies on Twitter because we try to compare the content differences between Twitter and traditional news media. In terms of topic modeling, our model is based on [8] but samples a single topic for a whole sentence. Recently, [9] applied labeled-LDA to Twitter, but the model relies on hashtags in Twitter, which may not include all topics. [7] conducted an empirical study of different strategies to aggregate tweets based on existing models. Our proposed Twitter-LDA differs from the models studied in [7] in that we model one tweet with one topic label, which is similar to [10,11] but for different applications. Another related area is comparison of text corpora [12,13,14]. The nature of Twitter makes our work more difficult than previous studies because tweets are short messages and different from traditional documents. In addition, no previous work has compared topics in different views, i.e. topics of different categories and topics of different types. A most recent piece of work [15] tries to explore search behavior on the popular microblogging site Twitter, which also has a different focus than ours.

### 6 Conclusions

In this paper we empirically compared the content of Twitter with a typical traditional news medium, New York Times, focusing on the differences between

these two. We developed a new Twitter-LDA model that is designed for short tweets and showed its effectiveness compared with existing models. We introduced the concepts of topic categories and topic types to facilitate our analysis of the topical differences between Twitter and traditional news media. Our empirical comparison confirmed some previous observations and also revealed some new findings. In particular, we find that Twitter can be a good source of entity-oriented topics that have low coverage in traditional news media. And although Twitter users show relatively low interests in world news, they actively help spread news of important world events.

In the future, we will study how to summarize and visualize Twitter content in a systematic way. Our method of associating tweets with different categories and types may also help visualization of Twitter content.

**Acknowledgement.** This work was done during Xin Zhao's visit to the Singapore Management University. Xin Zhao, Hongfei Yan and Xiaoming Li are partially supported by NSFC under the grant No. 70903008, 60933004, 61073082 and 61050009, CNGI grant No. 2008-122 and Grant No. SKLSDE-2010KF-03, Beihang University.

## References

1. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: Proceedings of the 19th WWW (2010)
2. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the 19th WWW (2010)
3. Asur, S., Huberman, B.A.: Predicting the future with social media. WI-IAT (2010)
4. Petrović, S., Osborne, M., Lavrenko, V.: The Edinburgh Twitter corpus. In: Proceedings of the NAACL HLT 2010 Workshop (2010)
5. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. JMLR (2003)
6. Weng, J., Lim, E.P., Jiang, J., He, Q.: TwitterRank: finding topic-sensitive influential twitterers. In: Proceedings of the Third ACM WSDM (2010)
7. Hong, L., Davison, B.D.: Empirical study of topic modeling in Twitter. In: Proceedings of the SIGKDD Workshop on SMA (2010)
8. Steyvers, M., Smyth, P., Rosen-Zvi, M., Griffiths, T.: Probabilistic author-topic models for information discovery. In: SIGKDD (2004)
9. Ramage, D., Dumais, S., Liebling, D.: Characterizing micrblogs with topic models. In: Proceedings of AAAI on Weblogs and Social Media (2010)
10. Titov, I., McDonald, R.: Modeling online reviews with multi-grain topic models. In: Proceedings of the 17th WWW (2008)
11. Li, P., Jiang, J., Wang, Y.: Generating templates of entity summaries with an entity-aspect model and pattern mining. In: Proceedings of the 48th ACL (2010)
12. Zhai, C., Velivelli, A., Yu, B.: A cross-collection mixture model for comparative text mining. In: Proceedings of the Tenth ACM SIGKDD (2004)
13. Paul, M., Girju, R.: Cross-cultural analysis of blogs and forums with mixed-collection topic models. In: Proceedings of the 2009 EMNLP (2009)
14. Leskovec, J., Backstrom, L., Kleinberg, J.: Meme-tracking and the dynamics of the news cycle. In: Proceedings of the 15th ACM SIGKDD (2009)
15. Teevan, J., Ramage, D., Morris, M.: #Twittersearch: A comparison of microblog search and web search. In: Proceedings of the Fourth ACM WSDM (2011)

# Exploiting Thread Structures to Improve Smoothing of Language Models for Forum Post Retrieval

Huizhong Duan and Chengxiang Zhai

University of Illinois at Urbana-Champaign,  
201 N Goodwin Ave, Urbana, IL 61801, USA  
duan9@illinois.edu, czhai@cs.uiuc.edu

**Abstract.** Due to many unique characteristics of forum data, forum post retrieval is different from traditional document retrieval and web search, raising interesting research questions about how to optimize the accuracy of forum post retrieval. In this paper, we study how to exploit the naturally available raw thread structures of forums to improve retrieval accuracy in the language modeling framework. Specifically, we propose and study two different schemes for smoothing the language model of a forum post based on the thread containing the post. We explore several different variants of the two schemes to exploit thread structures in different ways. We also create a human annotated test data set for forum post retrieval and evaluate the proposed smoothing methods using this data set. The experiment results show that the proposed methods for leveraging forum threads to improve estimation of document language models are effective, and they outperform the existing smoothing methods for the forum post retrieval task.

**Keywords:** Forum post retrieval, language modeling, smoothing.

## 1 Introduction

There are nowadays more and more ways for publishing information on the Web. Among them, online forums and discussion boards are of great importance and widely used. The reason lies in several aspects. For one thing, it is much easier for users to post contents on forums, compared with composing web pages. The infrastructure of forums allows users to focus on the content of the post instead of putting much effort on the designing of the presentation. For another, users are able to interact with each other in forums while they publish their opinions. This makes the web contents live and people are therefore more inclined to looking into forum posts for information. As more and more forums are available online, forum post retrieval becomes an important task. According to one of the most popular forum search engines, BoardTracker, it has more than 32,000 forums indexed [1]. A number of forum search engines have been built in recent years [2, 3]. Despite the growth of forums, little research has been done on models for forum post retrieval.

<sup>1</sup> <http://www.boardreader.com>

<sup>2</sup> <http://www.boardtracker.com>

<sup>3</sup> <http://www.omgili.com>



As a new retrieval problem, forum post retrieval both raises new challenges and offers new opportunities. The challenges are raised from the unique characters of forum posts. Posts are usually short in length. Background information is often omitted in posts as it is assumed that readers share the same background knowledge. For example, in Figure 1, post 2 and post 6 are suggesting the software “VNC” without mentioning its usage. This is because the authors of the posts assume their readers have already read the previous posts and are hence aware of the topic they are talking about. This raises big challenges for traditional retrieval techniques as the relation among posts cannot be overlooked.

On the other hand, there are new opportunities to optimize the performance of forum post retrieval as forums contain richer context for each post. A post usually has strong connection with its previous discussions. Therefore, the thread structure can be leveraged to overcome the aforementioned problems and improve the accuracy of retrieval. Indeed, recent work [13] has shown that high-quality thread structures learned based on manually created training data can improve retrieval performance. However, it is still unclear how we can improve retrieval accuracy by using only the *raw* thread structures naturally available in all the forums. In this paper, we study the use of raw structure information of forum contents to improve the quality of retrieval. Particularly, we study how to use the thread structure to improve smoothing of language models for forum post retrieval.

We propose two smoothing schemes for exploiting the thread structure in forums. The first is model expansion, which is in essence a variant of the two stage smoothing scheme [20]. In the first stage, it makes use of the language models of related posts to smooth the language model of the target post, in order to expand the language model to incorporate more contextual information and give better estimation of the topical words. In the second stage, it uses the collection language model to explain away the common words in queries. The second scheme is count expansion. In this smoothing scheme, we directly propagate the counts of words from relevant posts within the same thread to the target post. Then the



Fig. 1. A Fragment of a Thread

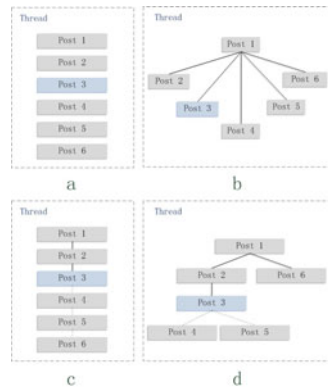


Fig. 2. Representation of a Thread

language model is estimated based on the propagated word counts using maximum likelihood estimation. The model is further smoothed with a collection language model in the end. Experiments show that both smoothing schemes can achieve good performances through appropriate use of thread structures.

Within the two proposed schemes, we further study different ways of adopting the thread context for a given post to improve the estimation of language models. Particularly, we study four different representations of the thread structure, namely the flat plate representation, one level tree representation, timeline representation and reply tree representation. Moreover, we also study different weighting functions for combining the contents of relevant posts, including structural distance, content similarity as well as their combination. Our experiments show that smoothing with the reply tree representation and combined weighting function tend to consistently achieve the best performance.

To test the performance of our proposed methods as well as existing retrieval models, we created a test set consisting of a full crawl of an online forum as well as a set of automatically generated queries. The queries are generated from the online community of question answering services to reflect the real world information needs. Manual judgements are obtained through the use of Amazon Mechanical Turk<sup>4</sup>. Voting is performed to ensure the quality of judgements. The test set has been made publicly available for future research on this topic<sup>5</sup>.

## 2 Related Work

Our study is based on the state-of-the-art language modeling approach in information retrieval [6, 19]. Language model was first applied to information retrieval more than a decade ago [11, 9, 4]. During its development, smoothing was shown to be a crucial part for achieving good retrieval performance [19, 20]. Furthermore, Liu and Croft [8] and Tao et al. [14] used richer context for smoothing to further improve the performance. Our work adds to this line of work a new way of smoothing based on thread structures.

One similar topic to forum post retrieval is XML retrieval. XML retrieval is similar to forum post retrieval but different in the following aspects. First, an XML document is usually composed by the same author instead of multiple authors. Therefore, such contents are usually better formatted and more grammatical. Second, XML retrieval does not require long queries for expressing information need, as XML documents are typically records of facts and events. Queries for such information are usually short in length. For XML retrieval, language modeling was also shown to be effective [3, 10]. Ogilvie and Callan introduced shrinkage models which takes each node's parent node's model in estimation of language model [10]. Our method of modeling with reply structure is similar to the shrinkage method, but does not require as many parameters. Therefore, it is more suitable for practical use. Another related body of literature is Email discussion retrieval. Weerkamp et al. used thread context to improve

<sup>4</sup> <https://www.mturk.com>

<sup>5</sup> <http://timan.cs.uiuc.edu/downloads.html>

language modeling for Email archive retrieval [15]. However, they did not research into the detail structure of threads. In this paper we study the internal structure of forum threads for improving the performance of forum post retrieval.

There are also some initial work on forum mining and retrieval. Lin et al. [7] used a sparse coding approach to model the semantic topics of forum posts, and applied the model to reconstruct the reply relationship between posts. Xu and Ma [16] built implicit links among posts to simulate the link based algorithm on web pages. Cong et al. [2] and Hong and Davison [5] extract question answer pairs from discussions. None of these work studied the problem of forum post retrieval directly. One of the most recent work [13] revealed that with the *accurately annotated* structure of thread, retrieval performance can be substantially increased; in contrast, we study the potential of using *raw* thread structures in the threads to improve retrieval performance.

### 3 State-of-the-Art Language Models for Information Retrieval

Language model has been extensively studied for information retrieval in recent years. In this work, we use the KL-Divergence model [18] as our base model. In this model, queries and documents are considered as samples generated from different language models, and the ranking of documents is based on the negative KL-Divergence of the query language model and the document language model. Formally, the ranking score is computed as:

$$-D_{KL}(\theta_q||\theta_d) = -\sum_w p(w|\theta_q)\log\frac{p(w|\theta_q)}{p(w|\theta_d)} \quad (1)$$

where  $\theta_q$  and  $\theta_d$  are query language model and document language model, respectively. They are usually estimated through maximum likelihood estimation. Note that KL-Divergence is a general form of language model as it naturally subsumes other models such as query likelihood model.

To avoid overfitting and zero probability problem, smoothing is needed. We make use of two commonly used smoothing methods: Jelinek Mercer (JM) smoothing and Dirichlet prior smoothing [19]. In JM smoothing, each document language model is linearly interpolated with a collection background model:

$$p(w|\hat{\theta}_d) = (1 - \lambda)p_{ml}(w|\theta_d) + \lambda p_{ml}(w|C) \quad (2)$$

where  $p_{ml}(w|\theta_d)$  is the maximum likelihood estimation of the probability of word  $w$  in document  $d$ 's language model,  $p_{ml}(w|C)$  is the probability of  $w$  in the background model.  $\lambda$  is a parameter controlling the amount of smoothing.

In Dirichlet smoothing, the document model is considered to have a conjugate prior with Dirichlet distribution. The derived smoothing formula is:

$$p(w|\hat{\theta}_d) = \frac{|d|}{|d| + \mu} p_{ml}(w|\theta_d) + \frac{\mu}{|d| + \mu} p_{ml}(w|C) \quad (3)$$

where  $\mu$  is a parameter controlling the amount of smoothing and can also be interpreted as the total number of pseudo counts of words introduced through the prior.

A number of techniques were proposed to further improve these basic smoothing methods [17]. Closely related to our work are two studies [8][14] that use similar documents to smooth a document language model. Liu and Croft [8] proposed a clustering based smoothing method (CBDM), where each document is first smoothed with clustering analysis before being smoothed with the collection language model. Tao et al. [14] improved CBDM by introducing the neighborhood based smoothing method (DELM). In their method, each document is enriched using the contents of the documents that have the highest content similarity with it before apply maximum likelihood estimation. Therefore documents are treated more fairly in terms of incorporating contexts. Both methods are expensive to compute. Although these methods can be applied to forum post retrieval, they do not take advantage of the thread structures in a forum. As shown in Figure 1, a thread provides very useful context for a post. In the next section, we propose new smoothing methods to exploit thread structures for smoothing.

## 4 Improving Smoothing by Exploiting Thread Structure

In this section, we propose two smoothing schemes for exploiting thread context to improve smoothing of a forum post language model, namely the model expansion scheme and the count expansion scheme. We then study different ways of exploiting thread context, as well as ways to weight and combine them into our smoothing scheme.

### 4.1 The Smoothing Schemes

**Model Expansion.** The essence of our model expansion scheme is a two stage smoothing process. In the first stage we make use of language model of the thread context of a post to smooth its language model obtained through maximum likelihood estimation. This is meant to improve the coverage of contextual words as well as to improve the estimation of topical words. In the second stage we further smooth the language model with a reference model, in order to assign non-zero probability to unseen words and explain away the common words [20]. In the first stage of smoothing, Dirichlet prior smoothing is used because the amount of contextual information to be incorporated is dependent on the length of the target post. A short post would need more contextual contents while a long post may already have enough contexts in itself. In the second stage Jelinek Mercer smoothing is used. In fact, different settings of smoothing methods are explored in practice and it is concluded that such a setting consistently achieves a slightly better performance. Therefore, we base our discussion on this setting. With the KL-Divergence language model, the ranking formula of model expansion scheme can be derived as:

$$-D_{KL}(\theta_q || \theta_d) \propto \sum_{w \in d} p_{ml}(w | \theta_q) \log \left( 1 + \frac{(1-\lambda) \left( \frac{|d|}{|d|+\mu} p_{ml}(w | \theta_d) + \frac{\mu}{|d|+\mu} p(w | \theta_{T(d)}) \right)}{\lambda p_{ml}(w | C)} \right) \quad (4)$$

where

$$p(w|\theta_{T(d)}) = \sum_{d' \in T(d)} \omega(d', d) p_{ml}(w|\theta_{d'}) \quad (5)$$

where  $T(d)$  is the thread context/relevant content of  $d$ , and  $\omega(d', d)$  is a weighting function for the interpolation of the thread contexts,  $\sum_{d' \in T(d)} \omega(d', d) = 1$ . Given this smoothing scheme, our major challenge is to figure out the optimal setting of  $T(d)$  and  $\omega(d', d)$ , which we will discuss later.

**Count Expansion.** In count expansion scheme, we adopt the similar idea to neighborhood based smoothing. We first propagate the contextual words to the target post to obtain pseudo counts, then estimate the language model based on the pseudo counts with maximum likelihood estimation. Instead of linearly interpolating the probabilities from different models, we interpolate the counts of words directly in this scheme. [14] suggests that count expansion tends to achieve better performance than model expansion. Formally, the ranking function can be derived as:

$$-D_{KL}(\theta_q|\theta_d) \propto \sum_{w \in d} p_{ml}(w|\theta_q) \log\left(1 + \frac{(1-\lambda)p_{ml}(w|\theta_{d_{exp}})}{\lambda p_{ml}(w|C)}\right) \quad (6)$$

where

$$p_{ml}(w|\theta_{d_{exp}}) = \frac{(1-\beta)c(w; d) + \beta \sum_{d' \in T(d)} \omega(d', d)c(w; d')}{(1-\beta)|d| + \beta \sum_{d' \in T(d)} \omega(d', d)|d'|} \quad (7)$$

where  $c(w; d)$  is the count of word  $w$  in post  $d$ . The count expansion scheme is also subject to the implementation of  $T(d)$  and  $\omega(d', d)$ . In the subsequent discussions, we will describe in detail how  $T(d)$  and  $\omega(d', d)$  are developed.

## 4.2 Utilizing Thread Contexts

As aforementioned, for both model expansion and count expansion, we need to define  $T(d)$  and  $\omega(d', d)$  in order to incorporate thread context. Here we discuss different possibilities for defining the two functions, each leading to a different variation of the model expansion smoothing scheme and the count expansion scheme.

**Smoothing with Flat Plate Representation.** The simplest way to define  $\omega(d', d)$  is to give equal weight to all the posts within the same thread. This corresponds to our notion of flat plate representation (Figure 2a). Formally,  $T(d)$  consists of all other posts in the thread beside  $d$ , and,

$$\omega(d', d) = \frac{1}{|T(d)|} \quad (8)$$

To further study the weighting function, we introduce two other weighting factors. They are both defined over a pair of posts within the same thread.

*Inverse Structural Distance.* This corresponds to the number of posts between the two posts in a certain representation of threads plus one. For example, in Figure 2a (assuming the time information is preserved in the representation),

the inverse structural distance between post 1 and post 3 is  $1/2$ . While in [2b](#) the inverse structural distance between post 1 and 3 is 1. A large inverse structural distance usually indicates a tight relationship between two posts, if the structure is correctly represented. This is denoted by  $IDist(d', d)$ .

*Contextual Similarity.* This corresponds to the similarity between the contents of two posts. Intuitively, posts with similar word distributions would share more contexts. In practice we use cosine similarity to measure the contextual similarity. This is denoted as  $Sim(d', d)$

The weighting function  $\omega(d', d)$  is defined based on these two factors. We explore three different settings, corresponding to using normalized  $IDist(d', d)$  and  $Sim(d', d)$  each separately and using them as a combined function:

$$\omega(d', d) = \frac{IDist(d', d)Sim(d', d)}{\sum_{d'' \in T(d)} IDist(d'', d)Sim(d'', d)} \quad (9)$$

**Smoothing with One-Level Tree Representation.** One major problem with the flat plate representation is that it uses all the posts within a thread without really considering the internal structure of a thread. Usually, many posts within the same thread are actually about very different topics. One important observation in the thread structure is that the first post tends to be more important than all the others, as it initiates the discussion and provides the background knowledge for all the following posts. Therefore, we can use the one-level tree representation for exploring contextual information, which assigns  $T(d) = \{Root(d)\}$  for any  $d$ . Figure [2b](#) illustrates this representation. This extreme method assigns 1 to the first post and 0 to all the others in the weighting function  $\omega(d', d)$ . As there is only one relevant post involved, we do not need to further explore the use of structural distance or contextual similarity.

**Smoothing with Timeline Representation.** Smoothing with the entire thread and smoothing with only the first post are both extreme approaches. Intuitively, the context of a post is mostly captured by the contents posted before the post. Posts published after the target post may introduce off-topic information or even noise. Based on this idea, we make use of the timeline representation of the thread and design the weighting function. An example is given in [2c](#). We restrict  $T(d) = \{d' | d' \in Thread(d) \& d' \prec d\}$ , where  $d' \prec d$  means  $d'$  is posted before  $d$ . Structural distance and content similarity can then be used as weighting functions.

**Smoothing with Reply Tree Representation.** In forums, it is possible that physically adjacent posts do not share the same topic at all. This is usually due to the problem known as topic drift. Therefore, instead of considering all contents posted before a post as the context, it makes more sense to explore the reply structure in order to get contexts. A thread starts with a single topic, and gradually grows into a multi-topic discussion. Future users tend to participate in one of the specified topics more than in the basic general topic. Finally the thread grows into a tree structure based on the reply-to relation. In this reply tree representation, the context of a post is given by the posts it replies to. These

contents naturally indicate how the topic of the given post is developed. Based on this observation, we can focus on exploring the reply relation when designing the weighting functions. Particularly we restrict  $T(d) = \{d' | d' \in Thread(d) \& d' \leftarrow d\}$ , where  $d' \leftarrow d$  means  $d'$  is on the reply path from  $d$  to the first post in the thread. Figure 2d demonstrates this representation. For  $\omega(d', d)$ , we apply the same combinations aforementioned. Here the structural distance corresponds to the reply distance between two posts.

## 5 Experiments

### 5.1 Data Set

A main challenge in studying forum post retrieval is the lack of a test set. We solve this problem by constructing a test set with publicly available resources.

We first obtained a full crawl of the “Computer Help” forum of CNET forum<sup>6</sup>. The crawl includes 29,413 threads, 25,830 users and 135,752 posts. On average, each thread has 4.6 posts and each user writes 1.13 posts. The forum is parsed so that all the metadata including the time stamp and the reply-to link are preserved. The posts are indexed so that consecutive posts within the same thread have consecutive IDs.

To generate a query set that reflects the scenario of forum search, we crawled the “Computers & Internet” category of Yahoo! Answers<sup>7</sup>. We then randomly select 30 question titles from the category whose words all appeared in our forum. Stopwords are filtered from the question titles, and the remaining keywords are used as queries. The average length of the test queries is 6.4 words.

We use Amazon Mechanical Turk to perform manual judgements. Particularly, we first use BM25 model [12] to retrieve the top 30 documents for each query, and ask the labelers to judge the relevance of each document. In each assignment, a labeler is presented with a question and a post, and asked to judge the post as 2, 1 or 0, representing “answers the question”, “contains relevant information but not directly answers the question” and “is irrelevant” correspondingly. To guarantee the quality of the human labelers, we require the participants to have a task approval rate of more than 95%. During the judging process, we also monitor the behaviors of labelers and reject abnormal submissions, e.g. time usage less than 5 seconds. In total, 73% submitted assignments are accepted and the average time usage for each assignment is 17 seconds.

Five labelers are employed to judge each document. The final judgement is given by majority voting. In the case of tie, the document is judged as “contains relevant information but not directly answers the question”. The average human agreement rate is 0.63. There are on average 7.8 documents judged as 2, 15.3 as 1 and 6.9 as 0. As we can see, the labelers tend to judge posts as 1 – “contains relevant information but not directly answers the question”. In order to guarantee the quality of retrieval, we take level 2 as relevant, level 1 and 0 are both irrelevant.

<sup>6</sup> <http://forums.cnet.com/computer-help-forum/>

<sup>7</sup> <http://answers.yahoo.com/dir/index?sid=396545660>

**Table 1.** Performance of Existing Retrieval Models

	MAP	P@1	P@5	p@10
BM25	0.445	0.467	0.373	0.343
LM+DIR	0.415	0.367	0.387	0.357
LM+JEL	0.457	0.467	0.373	0.357
CBDM	0.434	0.433	0.347	0.317
CBDM2	0.487	0.433	0.407	0.343
DELM	<b>0.489</b>	0.467	0.367	0.353

We evaluate the performance of all the methods by doing re-ranking on the test set. This is because we are experimenting on different retrieval models and incremental pooling is too expensive. The metrics we use for evaluation are Mean Average Precision (MAP), Precision at 1, 5 and 10 (P@1, P@5 and P@10). All the reported results are based on 5-fold cross validation w.r.t. the MAP measure.

## 5.2 Experiment Results

**Existing Retrieval Models.** Since forum post retrieval is a new retrieval task, we first compare the performance of different existing retrieval models for this task. We see that BM25 model and language model achieve comparable performance. Jelinek Mercer smoothing gives better performance than Dirichlet smoothing. This is consistent with [19]’s finding that Jelinek Mercer smoothing outperforms Dirichlet smoothing on long queries due to better modeling of common words in the query.

We also see that CBDM does not actually outperform language model, while DELM improves the performance significantly. A slightly modified version of CBDM, namely CBDM2, rewrites the CBDM formula to achieve the IDF effect. CBDM2 achieves a slightly lower performance than DELM. This is also in accordance with [14]’s findings. These methods serve as the baselines for studying the effectiveness of our proposed new smoothing methods.

**Smoothing with Model Expansion.** Table 2 shows the performance of the model expansion scheme w.r.t. all the combination of ways to utilize the thread context. For simplicity, we denote model expansion scheme as “ME” and count expansion scheme as “CE” in the following. We also denote flat plate representation, one-level tree representation, timeline representation and reply tree representation of threads as “FL”, “ON”, “TI” and “RE” correspondingly. The four weighting formula – equal weight, inverse structural distance, contextual similarity and the combination of inverse structural distance and contextual similarity, are denoted as “EQ”, “DS”, “SI” and “DSSI” respectively.

In the results we see that almost all of the combinations in ME outperform the existing retrieval methods. This verifies our hypothesis that threads are natural clusters of posts. We also see that the best performance is achieved by ME+RE+DSSI. A deeper analysis suggests that using RE representation gives us better performance than any other representations. This shows the importance of utilizing the reply structure of forum threads, as they provide “cleaner”



**Table 2.** Model Expansion

	MAP	P@1	P@5	p@10
ME+FL+EQ	0.489	0.467	0.38	0.35
ME+FL+DS	0.494	0.4	0.393	0.373
ME+FL+SI	0.492	0.433	0.400	0.367
ME+FL+DSSI	0.499	0.433	0.413	0.363
ME+ON+EQ	0.504	0.467	0.393	0.36
ME+TI+EQ	0.492	0.467	0.407	0.35
ME+TI+DS	0.496	0.467	0.4	0.36
ME+TI+SI	0.506	0.467	0.427	0.363
ME+TI+DSSI	0.508	0.567	0.4	0.367
ME+RE+EQ	0.503	0.467	0.393	0.367
ME+RE+DS	0.508	0.5	0.42	0.363
<b>ME+RE+SI</b>	0.513	0.5	0.413	0.373
<b>ME+RE+DSSI</b>	<b>0.515</b>	0.533	0.420	0.363

**Table 3.** Count Expansion

	MAP	P@1	P@5	p@10
CE+FL+EQ	0.493	0.5	0.373	0.363
CE+FL+DS	0.494	0.5	0.387	0.367
CE+FL+SI	0.493	0.5	0.38	0.367
CE+FL+DSSI	0.503	0.433	0.393	0.37
CE+ON+EQ	0.511	0.433	0.393	0.363
CE+TI+EQ	0.507	0.5	0.387	0.357
CE+TI+DS	0.516	0.467	0.407	0.367
CE+TI+SI	0.498	0.467	0.387	0.37
CE+TI+DSSI	0.507	0.5	0.387	0.37
CE+RE+EQ	0.509	0.467	0.36	0.377
<b>CE+RE+DS</b>	0.515	0.5	0.36	0.38
<b>CE+RE+SI</b>	0.517	0.5	0.373	0.38
<b>CE+RE+DSSI</b>	<b>0.523</b>	0.533	0.38	0.38

context for posts. Meanwhile, in the experiments, DSSI consistently performs better than other weighting techniques, and it is also demonstrated that both structural distance and contextual similarity contribute to the improvement of accuracy. Another interesting finding is that smoothing with only the first post achieves fairly good performance. Unlike smoothing with other representations of threads, the first post is always guaranteed to be relevant to the target post. Therefore, we see in the results that ME+ON+EQ outperforms almost all the other representations with equal weighting function. However, after structural distance and contextual similarity are used for weighting functions, the other representations can outperform or at least catch up with the performance of ME+ON+EQ.

**Smoothing with Count Expansion.** Table 3 shows the performance of the count expansion scheme w.r.t. all the combination of ways to utilize thread context. Again, we see that all the combinations in CE outperform the existing retrieval methods, further confirming the effectiveness of the proposed new smoothing methods. Compared with ME, we can see that the optimal performance of CE, which is achieved by CE+RE+DSSI, is slightly better. This is in accordance with the previous findings in [14] that count expansion tends to be superior compared with model expansion. Besides, we see almost all of the findings in the previous subsection are also verified in the count expansion scheme. Therefore, we are able to conclude that thread structure in forums provides highly useful contextual information for posts. By appropriately exploiting the thread structure, we are able to significantly improve the state-of-the-art retrieval methods in forum post retrieval.

**Statistical Significance.** We perform t-test over all the runs in our experiment. The runs that outperform all the existing retrieval methods significantly ( $p$ -value<0.05) are shown in bold font in Table 2 and Table 3. This indicates that (1) modeling threads with reply tree representation is the most important factor in improving the performance of forum post retrieval, and (2) the use of context

similarity and reply distance for expansion is critical in achieving a significant improvement.

## 6 Conclusions

This paper studies the problem of forum post retrieval. The contributions of this paper are as follows. First, to the best of our knowledge, this is the first systematic study of the problem of forum post retrieval by exploiting the *raw* thread structures of forums. We constructed a test set for forum post retrieval; while our data set is small, it is possible to differentiate different retrieval methods with statistically significant results. We propose and explore two new smoothing schemes to exploit the thread structure in order to improve smoothing of language models. We also propose and study different ways of utilizing the contextual information within the smoothing schemes. Finally, extensive experiments are carried out on the test data we constructed and it is demonstrated that our proposed smoothing schemes can improve the accuracy of forum post retrieval significantly.

As for future work, we plan to explore several directions to improve forum post retrieval. First, we plan to further explore the problem of forum post retrieval on a larger scale of data. The current evaluation set is relatively small due to the limitation of resources. We plan to study how to scale up the evaluation without incurring too much cost. Meanwhile, we also plan to take into consideration the static ranking of forum posts to further optimize retrieval performance.

## Acknowledgments

This paper is based upon work supported in part by MIAS, the Multimodal Information Access and Synthesis center at UIUC, part of CCICADA, a DHS Center of Excellence, and by the National Science Foundation under grants IIS-0713581 and CNS-0834709.

## References

1. <http://www.boardtracker.com/cgi-bin/about.pl?page=1>
2. Cong, G., Wang, L., Lin, C.-Y., Song, Y.-I., Sun, Y.: Finding question-answer pairs from online forums. In: SIGIR 2008, pp. 467–474. ACM, New York (2008)
3. Hiemstra, D.: Statistical language models for intelligent XML retrieval. In: Intelligent Search on XML Data, pp. 107–118 (2003)
4. Hiemstra, D., Kraaij, W.: Twenty-one at trec-7: Ad-hoc and cross-language track. In: TREC 1999, pp. 227–238 (1999)
5. Hong, L., Davison, B.D.: A classification-based approach to question answering in discussion boards. In: SIGIR 2009 (2009)
6. Lafferty, J., Zhai, C.: Document language models, query models, and risk minimization for information retrieval. In: SIGIR 2001, September 2001, pp. 111–119 (2001)

7. Lin, C., Yang, J.-M., Cai, R., Wang, X.-J., Wang, W.: Simultaneously modeling semantics and structure of threaded discussions: a sparse coding approach and its applications. In: SIGIR 2009, pp. 131–138. ACM, New York (2009)
8. Liu, X., Croft, W.B.: Cluster-based retrieval using language models. In: SIGIR 2004, pp. 186–193. ACM Press, New York (2004)
9. Miller, D.R.H., Leek, T., Schwartz, R.M.: BBN at trec7: Using hidden markov models for information retrieval. In: Proceedings of the Seventh Text REtrieval Conference (TREC-7), pp. 80–89 (1998)
10. Ogilvie, P., Callan, J.: Hierarchical language models for xml component retrieval. In: Proceedings of INEX Workshop
11. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: SIGIR 1998, pp. 275–281. ACM Press, New York (1998)
12. Robertson, S.E., Walker, S.: Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In: SIGIR, pp. 232–241 (1994)
13. Seo, J., Croft, W.B., Smith, D.A.: Online community search using thread structure. In: CIKM 2009, pp. 1907–1910. ACM, New York (2009)
14. Tao, T., Wang, X., Mei, Q., Zhai, C.: Language model information retrieval with document expansion. In: HLT-NAACL 2006, pp. 407–414. Association for Computational Linguistics, Morristown (2006)
15. Weerkamp, W., Balog, K., de Rijke, M.: Using contextual information to improve search in email archives. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 400–411. Springer, Heidelberg (2009)
16. Xu, G., Ma, W.-Y.: Building implicit links from content for forum search. In: SIGIR 2006, pp. 300–307. ACM, New York (2006)
17. Zhai, C.: Statistical Language Models for Information Retrieval. In: Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers (2008)
18. Zhai, C., Lafferty, J.: Model-based feedback in the language modeling approach to information retrieval. In: CIKM 2001, pp. 403–410. ACM Press, New York (2001)
19. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: SIGIR 2001, pp. 334–342. ACM Press, New York (2001)
20. Zhai, C., Lafferty, J.: Two-stage language models for information retrieval. In: SIGIR 2002 (2002)

# Incorporating Query Expansion and Quality Indicators in Searching Microblog Posts

Kamran Massoudi, Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp

ISLA, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands  
kamran.massoudi@gmail.com, {e.tsagkias, derijke, w.weerkamp}@uva.nl

**Abstract.** We propose a retrieval model for searching microblog posts for a given topic of interest. We develop a language modeling approach tailored to microblogging characteristics, where redundancy-based IR methods cannot be used in a straightforward manner. We enhance this model with two groups of quality indicators: textual and microblog specific. Additionally, we propose a dynamic query expansion model for microblog post retrieval. Experimental results on Twitter data reveal the usefulness of boolean search, and demonstrate the utility of quality indicators and query expansion in microblog search.

## 1 Introduction

Microblogging platforms such as Twitter have become important real-time information resources [4], with a broad range of uses and applications, including event detection [13, 15], and mining consumer and political opinions [6, 14]. In this paper we focus on the task of finding microblog posts for a given query; this task can be thought of as building block of other (mining) tasks that require posts on a specific topic for further downstream processing. The task of searching microblog posts has been studied before: [13] investigate the real-time nature of Twitter for event detection and use query expansion to improve recall. [5] analyze so-called hashtag patterns over time and find that hashtags reflect a user's intent for joining discussions on a topic. TweetMotif [10], an exploratory search application for Twitter, groups messages by significant terms in the query subcorpus.

Items posted on microblogging platforms (like *tweets* and *status updates*) are a special type of user generated content. Their limited size has some interesting effects: (i) people use abbreviations or change spelling to fit their message in the allotted space, giving rise to a rather idiomatic language; (ii) redundancy-based IR methods may not be usable in a straightforward manner to provide effective access to very short documents.

To address the first effect, we introduce a set of *quality indicators* for microblog posts and incorporate these into our retrieval model. In a similar fashion, [16] build on a credibility framework [12] and consider credibility indicators for blog post search. They specifically look at indicators that do not make use of the blogger's identity, that are textual in nature, and can be reliably estimated using NLP-based methods. Microblogs also have their own characteristics that can be exploited as quality indicators. [7] study the topological characteristics of microblogs and try to identify influential users in microblogs. [9] extend this by considering the temporal order of information

adoption. Recency is considered an important aspect of microblogs and can be used to improve web ranking for recency sensitive queries [3]. Finally, [2] study the types and degrees of influence within the Twitter network and find that the number of followers represents a user’s popularity, but it is not related to retweets and mentions; the latter two indicate the content value of a microblog post and the value of a user’s name.

To overcome the second effect of redundancy-based IR methods, we re-examine the potential of local *query expansion* for searching microblog posts, using a time-dependent expansion flavor that accounts for the dynamic nature of a topic.

## 2 Retrieval Model and Extensions

As a baseline retrieval model, we use a generative language modeling approach (for more details, see for example [1]). An important part of this approach is the estimation of the probability of a term for a given document model (e.g., the microblog post model),  $P(t|\theta_d)$ . We use Jelinek-Mercer smoothing, and estimate  $P(t|\theta_d) = (1 - \lambda) \cdot P(t|d) + \lambda \cdot P(t)$ . Due to the short length of microblog posts we expect low language reuse within documents. For this reason, we assume that terms occurring more than once do not add supporting evidence for the relevance of a post  $d$ . This directly influences the way we estimate  $P(t|\theta_d)$ , as detailed in Eq. 1:

$$P(t|d) = \frac{\hat{n}(t,d)}{\sum_{t' \in \epsilon_d} \hat{n}(t',d)} \quad P(t) = \frac{\sum_d \hat{n}(t,d)}{N} \quad \hat{n}(t, d) = \begin{cases} 0 & \text{if } n(t, d) = 0 \\ 1 & \text{if } n(t, d) > 0, \end{cases} \quad (1)$$

where  $n(t, d)$  is the term frequency of  $t$  in  $d$ , and  $N$  is the total number of microblog posts (documents) in the collection.

We expect results to contain a fair amount of near-duplicates (so-called reposts in microblogging terminology). Reposts echo another post, but can contain additional information. This extra information can render them more relevant to the query. To deal with this, we proceed as follows. Given a list  $R$  of reposts  $r$ , originating from post source  $d$ , we keep the post  $r_{best}$  which has the highest a priori probability according to *recency* and *followers* indicators (see Sec. 2.1).

### 2.1 Quality Indicators

We borrow the following quality indicators for microblog posts from [16]: emoticons, post length, shouting, capitalization, and the existence of hyperlinks. We extend this set of indicators based on the following microblog post characteristics: *reposts*, *followers*, and *recency*, as these represent a certain value of the post.<sup>1</sup> We estimate the repost quality indicator as  $P_{repost}(d) = \log(1 + n_{reposts}(d))$ . Followers, the number of people “subscribed” to a microblog, are incorporated by putting  $P_{followers}(d) = \log(1 + n_{followers}(d))$ . Finally, we take recency into account by setting  $P_{recency}(d) = e^{-\gamma \cdot (c - c_d)}$ , where  $c$  is the query time,  $c_d$  the post time, and  $\gamma$  the recency parameter. These three microblog specific indicators are combined into a single value,  $P_{microblog}$ , by taking their average value.

<sup>1</sup> We write  $n_{reposts}(d)$  for the number of reposts of a microblog post  $d$  and  $n_{followers}(d)$  for the number of followers of the microblog to which post  $d$  belongs.

We combine all the indicators to estimate a global credibility prior probability  $P(d)$  for a microblog post  $d$ , using parameter  $\mu$  to weigh both indicator groups.

## 2.2 Query Expansion

We capture the dynamics of topics in a microblogging platform using query expansion: After expanding the query, the conditional probability of a term to occur in a query,  $P(t|\theta_Q)$ , is given by the weighted mixture of the original query  $Q$  and the expanded query  $\hat{Q}$ , controlled by parameter  $\alpha$ . To construct the expanded query, we rank terms according to Eq. 2, and select the top  $k$  terms. This model tries to take into account the dynamic nature of microblogging platforms: while a topic evolves, the language usage around it is expected to evolve as well. Consequently, selecting terms temporally closer to query time could result in terms that are more relevant for that point in time. Term scoring becomes a function of time:

$$\text{score}(t, Q) = \log \left( \frac{|N_c|}{|\{d : t \in d, d \in N_c\}|} \right) \cdot \sum_{\{d \in N_c : q \in Q \text{ and } t, q \in d\}} e^{-\beta(c-c_d)}, \quad (2)$$

where  $c$  is query submission time,  $c_d$  is post  $d$ 's publication time,  $N_c$  is the set of posts that are posted before time  $c$  and  $q$ . The parameter  $\beta$  controls the contribution of each post to the score of term  $t$  based on their posted time. We select only those terms as candidate expansion terms, that occur in more than  $\varphi$  posts.

## 3 Experimental Setup

For the purpose of evaluating microblog post search, we choose to focus on a single target collection in our experimental evaluation, namely Twitter. From the period Nov '09–Apr '10, we drew from Twitter's trending topics, to construct a topic set of 30 queries, with an average length of 1.4 words. We then collected all tweets posted between the last day the topic was "trending" and three days before that day. This resulted in a collection of 110,038,694 tweets. Queries were turned into "topic statements" (in TREC parlance) following the practice of the TREC Blog track [11]. That is, for assessment purposes, each query was equipped with a description of an information need and with a (news) event that corresponds to it. Ground truth was established by pooling the top-20 results of all the retrieval runs that we constructed and assessing the resulting set of posts. Judgments were binary: either relevant or non-relevant.

We do not preprocess the documents (e.g., stopword removal, stemming). Only for the special case of query expansion models, we use a list of 165 English and Spanish stopwords to curate candidate expansion terms. We report on standard IR measures: mean reciprocal rank (MRR), mean average precision (MAP), and precision at position 5 and 10 (P5, P10). Statistical significance is tested using a two-tailed paired t-test and is marked as  $\blacktriangle$  (or  $\blacktriangledown$ ) for significant differences for  $\alpha = .01$  and  $\triangle$  (and  $\triangledown$ ) for  $\alpha = .05$ .

In our language modeling approach, we set the smoothing parameter to  $\lambda = 0.15$ , reportedly a good value for short queries [17]. For recency, we set  $\gamma = 10^{-5}$  assuming  $c$  and  $c_d$  are measured in seconds. When combining indicators, we set  $\mu = 0.375$ .

For the query expansion model we use the top 10 candidate terms<sup>2</sup> and set  $\alpha = 0.5$ ,  $\beta = 1.5 \cdot 10^{-5}$ , and  $\varphi = 20$ .

## 4 Results and Analysis

The baseline for our retrieval experiments is set to boolean search plus recency ( $BS+R$ ). In this model, relevant posts have to include all query terms and are ranked in reverse chronological order (newer posts rank higher). Our proposed language model is reported as  $LM$ , the textual quality factor model as  $LM-T$ , the microblog quality factor model as  $LM-M$ , and their combination as  $LM-T+M$ . Results from our retrieval experiments are listed in Table 1. Runs labeled “bl” do not use query expansion and “qe” denotes the use of dynamic query expansion. As a sanity check, we have included a run (“rm2”) that uses an existing standard query expansion model, RM2 [8].

**Table 1.** Performance for boolean search plus recency ( $BS+R$ ), language modeling ( $LM$ ), textual ( $-T$ ) and microblog ( $-M$ ) quality factors with dynamic query expansion model enabled (qe) or disabled (bl), with query expansion based on RM2 added for comparison. Significance tested against  $BS+R$ . Boldface indicates best score in the respective metric.

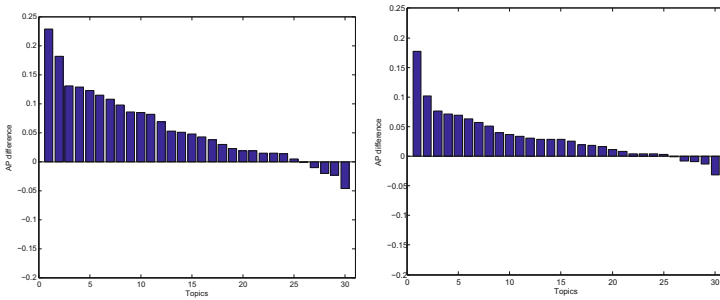
	$BS+R$		$LM$		$LM-T$		$LM-M$		$LM-T+M$	
	bl	rm2	qe	bl	qe	bl	qe	bl	qe	
MAP	0.1640	0.0700 <sup>▼</sup>	0.2390 <sup>△</sup>	0.4720 <sup>▲</sup>	0.1930	0.4640 <sup>▲</sup>	0.2190 <sup>▲</sup>	0.4510 <sup>▲</sup>	0.2500 <sup>▲</sup>	<b>0.4820<sup>▲</sup></b>
MRR	0.7170	0.1790 <sup>▼</sup>	0.5670 <sup>▽</sup>	<b>0.9670<sup>▲</sup></b>	0.7770	<b>0.9670<sup>▲</sup></b>	0.8220	0.9460 <sup>▲</sup>	0.8230	0.9510 <sup>▲</sup>
P5	0.5870	0.1470 <sup>▼</sup>	0.5530	0.9530 <sup>▲</sup>	0.5800	<b>0.9600<sup>▲</sup></b>	0.6930	0.9330 <sup>▲</sup>	0.7530 <sup>△</sup>	0.9530 <sup>▲</sup>
P10	0.5800	0.1570 <sup>▼</sup>	0.5670	0.9600 <sup>▲</sup>	0.6130	<b>0.9630<sup>▲</sup></b>	0.6870 <sup>△</sup>	0.9470 <sup>▲</sup>	0.7270 <sup>▲</sup>	0.9570 <sup>▲</sup>

Our first experiment compares the effectiveness of  $BS+R$  and  $LM$ .  $BS+R$  significantly outperforms  $LM$ , confirming the usefulness of simple boolean search and recency ranking. Next, we compare our query expansion model to RM. In general, both models show significant improvements in retrieval effectiveness compared to  $BS+R$  or  $LM$ . Our dynamic query expansion model outperforms RM2 significantly, and results in a 190% improvement in MAP over  $BS+R$  and a 500% improvement over  $LM$ .

Both  $LM-T$  and  $LM-M$  improve significantly over  $BS+R$ , and lead to 180%–260% improvements over  $LM$ . Their combination,  $LM-T+M$ , outperforms models based on individual quality indicator sets. Finally, we combine quality indicators and the query expansion model. Expanded  $LM-T$  and  $LM-M$ , individually, are unable to outperform query expansion on the baseline  $LM$ . After combining the indicator sets, however, query expansion achieves the best performance in terms of MAP.

To gain more insight in the results, we look at per topic differences in average precision (AP). Fig. 1 shows these differences, ordered by improvement in AP, between the combined quality indicators ( $LM-T+M$ ) and the individual sets ( $LM-T$  and  $LM-M$ ) without query expansion. The combination outperforms the individual sets on almost all topics. This result strongly suggests that both textual and non-textual indicators should be used in searching microblog posts.

<sup>2</sup> Based on a small set of preliminary experiments where  $k$  varied from 2 to 15 terms, we choose  $k = 10$  as a good balance between effectiveness improvement and efficiency.



**Fig. 1.** AP difference between  $LM-T+M$  and  $LM-T$  (Left) and  $LM-M$  (Right)

Next, zooming in on query expansion, we see that the query expansion component can help to retrieve real-time tweets. Looking at some examples, we observe that our model brings in top terms like 79 for the query *Woodward*<sup>3</sup> 4 and 4-1 for the query *Messi*<sup>4</sup>. Top expansion terms are not just single words, but also hyperlinks, hashtags or usernames. For the query *Google Docs*, one of the top expansion terms is <http://bit.ly/4r3sis>, which is the link to Google’s official blog. Examples of hashtags that are returned as top terms are #e09 (elections of 2009) for the query *Chile*, #sotu (State of the Union) for the query *Obama* or #hcr (health care reform) for the query *Health Care*. Finally, @wyclef<sup>5</sup> is one of the top terms for the query *Earthquake*. The @ appears in all the retweets of user *wyclef*, and as a result tweets from this user will rank higher. The examples above reveal that tokens with numeric or non-alphabetic characters (which are often discarded in traditional retrieval settings) can prove beneficial for query expansion in microblog post search.

## 5 Conclusions and Outlook

We have presented a model for retrieving microblog posts that is enhanced with textual and microblog specific quality indicators and with a dynamic query expansion model. The enhancements were specifically designed to help address the challenges of microblog posts search: rapid language evolution and limited within-document language re-use. We have compared the contribution of our models both individually and combined on Twitter data, and found that when all models are combined they have a significant positive impact on microblog post retrieval effectiveness. Query expansion on microblog data can be done in a dynamic fashion (taking time into account) and should include specific terms like usernames, hashtags, and links.

In future work, we envisage incorporating quality indicators for reweighing candidate terms for query expansion, exploring diversification techniques for further enhancement of our results and incorporating geolocation features as a prior to users or posts from different locations.

<sup>3</sup> Actor Edward Woodward passed away at age 79 on the day the query was issued.

<sup>4</sup> Lionel Messi scored four goals in a match, including his fourth hat-trick of 2010. Barcelona won the match against Arsenal 4-1.

<sup>5</sup> Musician and Haitian immigrant Wyclef Jean took to his Twitter account to ask fans to help with relief efforts after a 7.0-magnitude quake struck his homeland.



**Acknowledgments.** This research was supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430 (GALATEAS), by the PROMISE Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement no. 258191, by the DuOMAn project carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project nr STE-09-12, by the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.066.512, 612.061.814, 612.061.815, 640.004.802, 380-70-011 and by the Center for Creation, Content and Technology (CCCT).

## References

- [1] Balog, K., Weerkamp, W., de Rijke, M.: A few examples go a long way: Constructing query models from elaborate query formulations. In: SIGIR 2008 (2008)
- [2] Cha, M., Haddadi, H., Benevenuto, F., Gummadi, K.P.: Measuring user influence in Twitter: The million follower fallacy. In: ICWSM 2010 (2010)
- [3] Dong, A., Zhang, R., Kolari, P., Bai, J., Diaz, F., Chang, Y., Zheng, Z., Zha, H.: Time is of the essence: improving recency ranking using Twitter data. In: WWW 2010 (2010)
- [4] Golovchinsky, G., Efron, M.: Making sense of twitter search. In: CHI 2010 (2010)
- [5] Huang, J., Thornton, K.M., Efthimiadis, E.N.: Conversational tagging in twitter. In: HT 2010 (2010)
- [6] Jansen, B.J., Zhang, M., Sobel, K., Chowdury, A.: Twitter power: Tweets as electronic word of mouth. *J. Am. Soc. Inf. Sci. Technol.* 60(11), 2169–2188 (2009)
- [7] Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: WWW 2010 (2010)
- [8] Lavrenko, V., Croft, W.B.: Relevance based language models. In: SIGIR 2001 (2001)
- [9] Lee, C., Kwak, H., Park, H., Moon, S.: Finding influentials based on the temporal order of information adoption in twitter. In: WWW 2010 (2010)
- [10] O'Connor, B., Krieger, M., Ahn, D.: TweetMotif: Exploratory search and topic summarization for Twitter (2010)
- [11] Ounis, I., Macdonald, C., de Rijke, M., Mishne, G., Soboroff, I.: Overview of the TREC 2006 blog track. In: The Fifteenth Text REtrieval Conference (TREC 2006). NIST (2007)
- [12] Rubin, V.L., Liddy, E.D.: Assessing credibility of weblogs. In: AAAI-CAAW 2006 (2006)
- [13] Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: WWW 2010 (2010)
- [14] Tumasjan, A., Sprenger, T., Sandner, P., Welp, I.: Predicting elections with twitter: What 140 characters reveal about political sentiment (2010)
- [15] Vieweg, S., Hughes, A.L., Starbird, K., Palen, L.: Microblogging during two natural hazards events: what twitter contribute to situational awareness. In: CHI 2010 (2010)
- [16] Weerkamp, W., de Rijke, M.: Credibility improves topical blog post retrieval. In: ACL 2008:HLT (June 2008)
- [17] Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.* 22(2), 179–214 (2004)

# Discovering Fine-Grained Sentiment with Latent Variable Structured Prediction Models

Oscar Täckström<sup>1,2,\*</sup> and Ryan McDonald<sup>3</sup>

<sup>1</sup> Swedish Institute of Computer Science

<sup>2</sup> Dept. of Linguistics and Philology, Uppsala University

oscar@sics.se

<sup>3</sup> Google, Inc.

ryanmcd@google.com

**Abstract.** In this paper we investigate the use of latent variable structured prediction models for fine-grained sentiment analysis in the common situation where only coarse-grained supervision is available. Specifically, we show how sentence-level sentiment labels can be effectively learned from document-level supervision using hidden conditional random fields (HCRFs) [10]. Experiments show that this technique reduces sentence classification errors by 22% relative to using a lexicon and 13% relative to machine-learning baselines [1].

## 1 Introduction

Determining the sentiment of a fragment of text is a central task in the field of opinion classification and retrieval [8]. Most research in this area can be categorized into one of two categories: lexicon or machine-learning centric. In the former, large lists of phrases are constructed manually or automatically indicating the polarity of each phrase in the list. This is typically done by exploiting common patterns in language [2], lexical resources such as WordNet or thesauri [4], or via distributional similarity [11]. The latter approach – machine-learning centric – builds statistical text classification models based on labeled data, often obtained via consumer reviews that have been tagged with an associated star-rating, e.g., [9]. Both approaches have their strengths and weaknesses. Systems that rely on lexicons can analyze text at all levels, including fine-grained sentence, clause and phrase levels, which is fundamental to building user-facing technologies such as faceted opinion search and summarization [3]. However, lexicons are typically deployed independent of the context in which mentions occur, often making them brittle, especially in the face of domain shift and complex syntactic constructions [12]. The machine-learning approach, on the other hand, can be trained on the millions of labeled consumer reviews that exist on review aggregation websites, but the supervised learning signal is often at too coarse a level.

We propose a model that learns to analyze fine-grained sentiment strictly from coarse annotations. Such a model can leverage the plethora of labeled documents from multiple domains available on the web. In particular, we focus on sentence level sentiment

\* Part of this work was performed while the author was an intern at Google, Inc. and part was funded by the Swedish National Graduate School of Language Technology (GSLT).

<sup>1</sup> A longer version of this paper containing comprehensive descriptions of the models and experiments is available at: <http://soda.swedish-ict.se/4058>

(or polarity) analysis. As input, the system expects a sentence segmented document and outputs the corresponding sentence labels. The model we present is based on hidden conditional random fields (HCRFs) [10], a well-studied latent variable structured learning model that has been used previously in speech and vision. We show that this model naturally fits the task and can reduce fine-grained classification errors by over 20%.

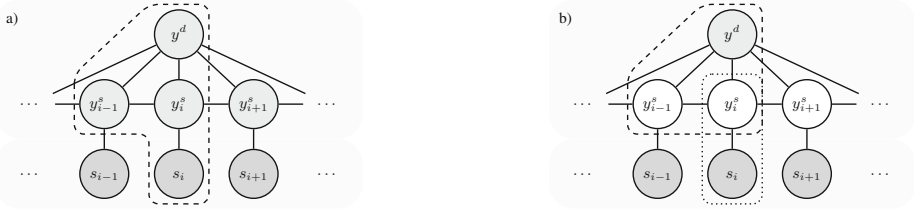
Latent-variable structured learning models have been investigated recently in the context of sentiment analysis. Nakagawa et al. [7] presented a sentence level model with latent variables placed at the nodes from the syntactic dependency tree of the sentence. Yessenalina et al. [13], presented a document level model with binary latent variables over sentences. In both these models, the primary goal was to improve the performance on the supervised annotated signal. This study inverts the evaluation and attempts to assess the accuracy of the latent structure induced from the observed coarse signal. In fact, one could argue that learning fine-grained sentiment from document level labels is the more relevant question for multiple reasons as 1) document level annotations are the most common naturally observed sentiment signal, e.g., star-rated consumer reviews, and 2) document level sentiment analysis is too coarse for most applications, especially those that rely on aggregation and summarization across fine-grained topics [3].

## 2 A Latent Variable Model of Fine-Grained Sentiment

The distribution of sentences in documents from our data (Table 2) suggests that documents contain at least one dominant class, even though they do not have uniform sentiment. Specifically, positive (negative) documents primarily consist of positive (negative) sentences as well as a significant number of neutral sentences and a small amount of negative (positive) sentences. This observation suggests that we would like a model where sentence level classifications are 1) correlated with the observed document label, but 2) have the flexibility to disagree when contextual evidence suggests otherwise.

To build such a model, we start with the supervised fine-to-coarse sentiment model described in [6]. Let  $d$  be a document consisting of  $n$  sentences,  $s = (s_i)_{i=1}^n$ . We denote by  $\mathbf{y}^d = (y^d, \mathbf{y}^s)$  random variables that include the document level sentiment,  $y^d$ , and the sequence of sentence level sentiment,  $\mathbf{y}^s = (y_i^s)_{i=1}^n$ . All random variables take values in  $\{\text{POS}, \text{NEG}, \text{NEU}\}$  for positive, negative and neutral sentiment respectively. We hypothesize that there is a sequential relationship between sentence sentiment and that the document sentiment is influenced by all sentences (and vice versa). Figure 1a shows an undirected graphical model reflecting this idea. A first order Markov property is assumed, according to which each sentence variable  $y_i^s$  is independent of all other variables, conditioned on the document variable  $y^d$  and its adjacent sentences,  $y_{i-1}^s/y_{i+1}^s$ . It was shown in [6] that when trained with fully supervised data, this model can increase both sentence and document level prediction accuracies.

We are interested in the common case where document labels are available during training, e.g., from star-rated reviews, but sentence labels are not. A modification to Figure 1a is to treat all sentence labels as unobserved as shown in Figure 1b. When the underlying model from Figure 1a is a conditional random field (CRF) [5], the model in Figure 1b is often referred to as a *hidden* conditional random field (HCRF) [10]. HCRFs



**Fig. 1.** a) Outline of graphical model from [6]. b) Identical model with latent sentence level states. Dark grey nodes are observed variables and white nodes are unobserved. Light grey nodes are observed at training time. Dashed and dotted regions indicate the maximal cliques at position  $i$ .

are appropriate when there is a strong correlation between the observed coarse label and the unobserved fine-grained variables, which Table 2 indicates is true for our task.

In the CRF model just outlined, the distribution of the random variables  $\mathbf{y}^d = (y^d, \mathbf{y}^s)$ , conditioned on input sentences  $\mathbf{s}$ , belongs to the exponential family and is written  $p_\theta(y^d, \mathbf{y}^s | \mathbf{s}) = \exp \{ \langle \phi(y^d, \mathbf{y}^d, \mathbf{s}), \theta \rangle - A_\theta(\mathbf{s}) \}$ , where  $\theta$  is a vector of model parameters and  $\phi(\cdot)$  is a vector valued feature function, which by the independence assumptions of the graphical models outlined in Figure 1 factorizes as  $\phi(y^d, \mathbf{y}^s, \mathbf{s}) = \oplus_{i=1}^n \phi(y^d, y_i^s, y_{i-1}^s, \mathbf{s})$ , where  $\oplus$  indicates vector summation. The log-partition function  $A_\theta(\mathbf{s})$  is a normalization constant, which ensures that  $p_\theta(y^d, \mathbf{y}^s | \mathbf{s})$  is a proper probability distribution. In an HCRF, the conditional probability of the observed variables is obtained by marginalizing over the posited hidden variables:  $p_\theta(y^d | \mathbf{s}) = \sum_{\mathbf{y}^s} p_\theta(y^d, \mathbf{y}^s | \mathbf{s})$ . As indicated in Figure 1b, there are two maximal cliques at each position  $i$ : one involving only the sentence  $s_i$  and its corresponding latent variable  $y_i^s$  and one involving the consecutive latent variables  $y_i^s, y_{i-1}^s$  and the document variable  $y^d$ . The assignment of the document variable  $y^d$  is thus independent of the input  $\mathbf{s}$ , conditioned on the sequence of latent sentence variables  $\mathbf{y}^s$ . This is in contrast to the original fine-to-coarse model, in which the document variable depends directly on the sentence variables as well as the input [6]. This distinction is important for learning predictive latent variables as it creates a bottleneck between the input sentences and the document label. When we allow the document label to be directly dependent on the input, we observe a substantial drop in sentence level prediction performance.

The feature function at position  $i$  is the sum of the feature functions for each clique at that position, that is  $\phi(y^d, y_i^s, y_{i-1}^s, \mathbf{s}) = \phi(y^d, y_i^s, y_{i-1}^s) \oplus \phi(y_i^s, s_i)$ . The features of the clique  $(y_i^s, s_i)$  include the identities of the tokens in  $s_i$ , the identities and polarities of tokens in  $s_i$  that match the polarity lexicon described in [12], and the corresponding number of positive and negative tokens in  $s_i$ . Features for  $(y^d, y_i^s, y_{i-1}^s)$ -cliques only involve various combinations of the document and sentence sentiment variables.

Typically, when training HCRFs, we find the MAP estimate of the parameters with respect to the marginal conditional log-likelihood of observed variables, assuming a Normal prior,  $p(\theta) \sim \mathcal{N}(0, \sigma^2)$ . However, early experiments indicated that using *hard* estimation gave slightly better performance. Let  $D = \{(\mathbf{s}_j, y_j^d)\}_{j=1}^m$  be a training set of document / document-label pairs. In hard HCRF estimation we seek parameters  $\theta$  such that:

$$\operatorname{argmax}_{\theta} \sum_{j=1}^{|D|} \log p_{\theta}(y_j^d, \hat{\mathbf{y}}_j^s | \mathbf{s}_j) - \frac{\|\theta^2\|}{2\sigma^2}, \quad (1) \quad \text{with: } \hat{\mathbf{y}}_j^s = \operatorname{argmax}_{\mathbf{y}^s} p_{\theta}(y_j^d, \mathbf{y}^s | \mathbf{s}_j). \quad (2)$$

We find the parameters  $\theta$  that maximizes (1) by stochastic gradient ascent for 75 iterations, with a fixed step size,  $\eta$ . Contrary to a fully observed CRF, equation (1) is not concave. Still, for the studied data set, results were stable simply by initializing  $\theta$  to the zero vector. Equation (2) is also used in predicting the optimal assignment of  $(y^d, \mathbf{y}^s)$ . An efficient solution to (2) is provided by the Viterbi algorithm as described in [6].

### 3 Experiments

For our experiments we constructed a large balanced corpus of consumer reviews from a range of domains. A training set was created by sampling a total of 143,580 positive, negative and neutral reviews from five different domains: *books*, *dvds*, *electronics*, *music* and *videogames*. Document sentiment labels were obtained by labeling one and two star reviews as negative (NEG), three star reviews as neutral (NEU), and four and five star reviews as positive (POS). The total number of sentences is roughly 1.5 million. To study the impact of the training set size, additional training sets, denoted  $S$  and  $M$ , were created by sampling 1,500 and 15,000 documents from the full training set, denoted  $L$ . A smaller separate test set of 294 reviews was constructed by the same procedure. This set consists of 97 positive, 98 neutral and 99 negative reviews. Two annotators marked each test set review sentence as having positive (POS), negative (NEG) or neutral (NEU) sentiment. NEU was used for sentences that are either truly neutral in opinion, have mixed positive/negative sentiment or contain no sentiment. Annotation statistics can be found in Table 1, while Table 2 shows the distribution of sentence level sentiment for each document sentiment category. Overall raw inter-annotator agreement was 86% with a Cohen’s  $\kappa$  value of 0.79<sup>2</sup>.

We compare the HCRF model to two baseline models: 1) VoteFlip, which determines the polarity of a sentence with the vote-flip algorithm [11], using the polarity lexicon from [12], and 2) Document as Sentence (DaS), which trains a document classifier on the coarse-labeled training data, but applies it to sentences independently at test time. In order to make the underlying statistical models the same, DaS was parameterized as a log-linear model with a Normal prior distribution and stochastic gradient ascent was used to find the maximum a posteriori point estimate of the parameters. We also measured the benefit of observing the document label at test time. This is a common scenario in, e.g., consumer-review aggregation [3]. The baseline of predicting all sentences with the observed document label is denoted DocOracle. Ten runs were performed with different random seeds and results were gathered by hierarchically bootstrapping medians and confidence intervals, which were also used to test statistical significance.

Table 3 shows results for each model in terms of sentence and document accuracy as well as  $F_1$ -scores for each sentence sentiment category. When using enough training data, the HCRF significantly outperforms all baselines in terms of sentence-level accuracy with quite a wide margin. Errors are reduced by 22% relative to the

<sup>2</sup> The test set is available at <http://www.sics.se/people/oscar/datasets>

**Table 1.** Number of documents per category (left) and number of sentences per category (right) in the test set for each domain

	POS	NEG	NEU	Total	POS	NEG	NEU	Total
Books	19	20	20	59	160	195	384	739
Dvds	19	20	20	59	164	264	371	799
Electronics	19	19	19	57	161	240	227	628
Music	20	20	19	59	183	179	276	638
Videogames	20	20	20	60	255	442	335	1,032
Total	97	99	98	294	923	1,320	1,593	3,836

**Table 2.** Distribution of sentence labels (columns) in documents by their labels (rows) in the test data

	POS	NEG	NEU
POS	0.53	0.08	0.39
NEG	0.05	0.62	0.33
NEU	0.14	0.35	0.51

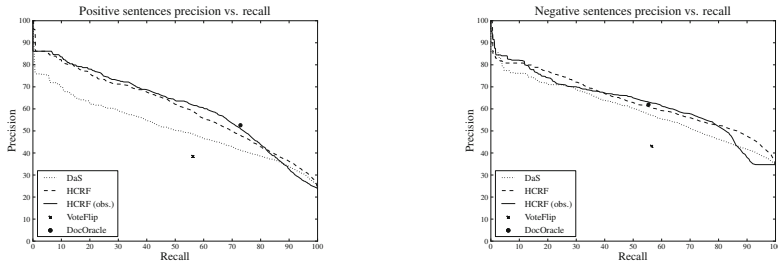
**Table 3.** Median and 95% conf. interval for  $S$ ,  $M$  and  $L$  training sets. Median  $F_1$ -scores for POS/NEG/NEU sentence labels. Bold: significantly better than comparable baselines,  $p < 0.05$ .

	Sentence acc. (S)	Sentence acc. (M)	Sentence acc. (L)	Sentence $F_1$ (L)	Document acc. (L)
VoteFlip	41.5 (-1.8, 1.8)	41.5 (-1.8, 1.8)	41.5 (-1.8, 1.8)	45.7 / 48.9 / 28.0	–
DaS	43.8 (-0.9, 0.8)	46.8 (-0.6, 0.7)	47.5 (-0.8, 0.7)	52.1 / 54.3 / 36.0	66.6 (-2.4, 2.2)
HCRF	43.0 (-1.2, 1.3)	<b>49.1 (-1.4, 1.5)</b>	<b>54.4 (-1.0, 1.0)</b>	<b>57.8 / 58.8 / 48.5</b>	64.6 (-2.0, 2.1)
DocOracle	54.8 (-3.0, 3.1)	54.8 (-3.0, 3.1)	54.8 (-3.0, 3.1)	61.1 / 58.5 / 47.0	–
HCRF (obs.)	48.6 (-1.6, 1.4)	54.3 (-1.9, 1.8)	<b>58.4 (-0.8, 0.7)</b>	62.0 / <b>62.3 / 53.2</b>	–

lexicon approach and by 13% compared to the machine learning baseline. When document labels are provided at test time, the corresponding reductions are 29% and 21%. In the latter case the reduction compared to the strong DocOracle baseline is only 8%, however, the probabilistic predictions of the HCRF are more useful than the fixed baseline as illustrated by Figure 2. In terms of document accuracy, DaS seems to slightly outperform the HCRF. This is contrary to the results reported in [13], where sentence-level latent variables improved document predictions. However, our model is restricted when it comes to document classification, since document variables are conditionally independent of the input. We observe from Table 3 that all models perform best on positive and negative sentences, with a sharp drop for neutral sentences, though the drop is substantially less for the HCRF. This is not surprising, as neutral documents are bad proxies for sentence sentiment, as can be seen from the sentence sentiment distributions per document category in Table 2.

Adding more training data improves all models and starting with the medium data set, the HCRF outperforms DaS. While the improvement from additional training data is relatively small for DaS, the improvement is substantial for the HCRF. We expect the benefit of using latent variables to increase further with more training data. Finally, Figure 2 shows sentence-level precision–recall curves for the HCRF, with and without observed document label, and DaS, together with the fixed points of VoteFlip and DocOracle. Label probabilities were calculated using the marginal probability of each label at each sentence position. The HCRF dominates at nearly all levels of precision/recall, especially for positive sentences.

Although the results for all systems may seem low, they are comparable with those in [6], which was a fully supervised model and evaluated with neutral documents excluded. In fact, the primary reason for the low scores presented here is the inclusion of neutral documents and sentences. Additional experiments with negative documents excluded showed that the HCRF achieves a document accuracy of 88.4%, which is on



**Fig. 2.** Interpolated precision-recall curves for positive and negative sentence level sentiment

par with reported state-of-the-art document accuracies for the two-class task [7,13]. Furthermore, the annotator agreement of 86% can be viewed as an upper bound on sentence accuracy.

## 4 Conclusions

In this paper we showed that latent variable structured prediction models can effectively learn fine-grained sentiment from coarse-grained supervision. Empirically, reductions in error of up to 20% were observed relative to both lexicon-based and machine-learning baselines. In the common case when document labels are available at test time as well, we observed error reductions close to 30% and over 20%, respectively, relative to the same baselines. In the latter case, our model reduces errors relative to the strongest baseline with 8%. The model we employed, a hidden conditional random field, leaves open a number of further avenues for investigating weak prior knowledge in fine-grained sentiment analysis, most notably semi-supervised learning when small samples of data annotated with fine-grained information are available.

## References

1. Choi, Y., Cardie, C.: Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In: Proc. EMNLP (2009)
2. Hatzivassiloglou, V., McKeown, K.R.: Predicting the semantic orientation of adjectives. In: Proc. EACL (1997)
3. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proc. KDD (2004)
4. Kim, S.-M., Hovy, E.: Determining the sentiment of opinions. In: Proc. COLING (2004)
5. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. ICML (2001)
6. McDonald, R., Hannan, K., Neylon, T., Wells, M., Reynar, J.: Structured models for fine-to-coarse sentiment analysis. In: Proc. ACL (2007)
7. Nakagawa, T., Inui, K., Kurohashi, S.: Dependency Tree-based Sentiment Classification using CRFs with Hidden Variables. In: Proc. NAACL (2010)
8. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Now Publishers (2008)
9. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: Proc. EMNLP (2002)
10. Quattoni, A., Wang, S., Morency, L.-P., Collins, M., Darrell, T.: Hidden conditional random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence (2007)

11. Turney, P.: Thumbs up or thumbs down? Sentiment orientation applied to unsupervised classification of reviews. In: Proc. ACL (2002)
12. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proc. EMNLP (2005)
13. Yessenalina, A., Yue, Y., Cardie, C.: Multi-level structured models for document-level sentiment classification. In: Proc. EMNLP (2010)



# Combining Global and Local Semantic Contexts for Improving Biomedical Information Retrieval

Duy Dinh and Lynda Tamine

University of Toulouse,  
118 route de Narbonne, 31062 Toulouse, France  
{dinh,lechani}@irit.fr

**Abstract.** In the context of biomedical information retrieval (IR), this paper explores the relationship between the document's global context and the query's local context in an attempt to overcome the term mismatch problem between the user query and documents in the collection. Most solutions to this problem have been focused on expanding the query by discovering its context, either *global* or *local*. In a global strategy, all documents in the collection are used to examine word occurrences and relationships in the corpus as a whole, and use this information to expand the original query. In a local strategy, the top-ranked documents retrieved for a given query are examined to determine terms for query expansion. We propose to combine the document's global context and the query's local context in an attempt to increase the term overlap between the user query and documents in the collection via document expansion (DE) and query expansion (QE). The DE technique is based on a statistical method (IR-based) to extract the most appropriate concepts (global context) from each document. The QE technique is based on a blind feedback approach using the top-ranked documents (local context) obtained in the first retrieval stage. A comparative experiment on the TREC 2004 Genomics collection demonstrates that the combination of the document's global context and the query's local context shows a significant improvement over the baseline. The MAP is significantly raised from 0.4097 to 0.4532 with a significant improvement rate of +10.62% over the baseline. The IR performance of the combined method in terms of MAP is also superior to official runs participated in TREC 2004 Genomics and is comparable to the performance of the best run (0.4075).

**Keywords:** Term Mismatch, Concept Extraction, Document Expansion, Query Expansion, Biomedical Information Retrieval.

## 1 Introduction

The effectiveness of an Information Retrieval (IR) system is influenced by the degree of term overlap between user queries and relevant documents. When a user searches an information in the collection, (s)he may formulate the query using another expression to mention the same information in the document. This causes the *term mismatch* problem yielding poor search results retrieved by IR systems [1].

In this paper, we focus on addressing the term mismatch problem in the biomedical domain. In this latter, biomedical documents contain many different expressions or term variants of the same concept such as synonyms ('cancer', 'tumor' are synonyms of the concept 'neoplasm'), abbreviations ('AMP' stands for 'Adenosine Monophosphate'), lexical variations such as differences in case, singular-plural inflection, etc. A natural solution to alleviate the term mismatch problem in biomedical IR is to use concepts in ontologies as means of normalizing the document vocabulary. Many works have been focused on both concept-based *Query expansion* (QE) [1-6] and/or *Document expansion* (DE) [3, 6-8].

The principle goal of QE is to increase the search performance by increasing the likelihood of the term overlap between a given query and documents that are likely to be relevant to the user information need. Current approaches of QE can be subdivided into two main categories: *global analysis* [1-3, 5, 6, 9] and *local analysis* [10-13]. Global techniques aim to discover word relationships in a large collection such as Web documents [9] or external knowledge sources like Wordnet [2], MeSH [5, 6] or UMLS [1, 3]. Local techniques emphasize the analysis of the top ranked documents retrieved for a query [10-12].

Similarly, DE can help to enhance the semantics of the document by expanding the document content with the most informative terms. This technique has been used recently in the context of textual document IR [7, 8] as well as in the context of biomedical IR [3, 6]. The difference between DE and QE is basically the timing of the expansion step. In DE, terms are expanded during the indexing phase for each individual document while in QE only query terms are expanded at the retrieval stage.

In the work presented in this paper, we explore the impact of using concepts from MeSH (global context) for a concept-based document and query representation enhanced with techniques of DE and QE. Our contributions are outlined through the following key points:

1. We propose a novel IR-based concept identification method for selecting the most representative concepts in each document. Concepts are then used to expand the document content to cope with the term mismatch problem.
2. We propose to combine the document's global context and the query's local context in an attempt to increase the term overlap between the user's query and documents in the collection through both DE and QE. The DE technique, which is classified as global, is based on our concept identification method using a domain terminology namely MeSH. The QE technique, which is classified as local, is based on a blind feedback approach using the top-ranked expanded documents obtained in the first retrieval stage.

The remainder of this paper is structured as follows. We introduce the related work in Section 2. Section 3 presents the techniques involved for combining query and document expansion. Experiments and results are presented in section 4. We conclude the paper in section 5 and outline research directions for future work.

## 2 Related Work

The term mismatch problem between user queries and documents has been the focus of several works in IR for almost 40 years [9, 10]. One of the earliest studies on QE was carried out by Sparck Jones [9] who clustered words based on co-occurrence in documents and used those clusters to expand the query. Since then, a number of studies have been undertaken and are divided into two main approaches: *global analysis* [2, 3, 5] and *local analysis* [10–12]. In a global strategy, all documents in the collection are used to examine word occurrences and relationships in the corpus as a whole, and use this information to expand any particular query. For example, the global context QE technique presented in [2] explored the lexical-semantic links in Wordnet in order to expand hierarchically related terms to the original query, but reported results have not been positive somehow due to term ambiguity. In a local strategy, the top-ranked documents retrieved for a given query  $q$  are examined to determine terms for QE. The local context QE presented in [10] involves generating a new query in an iterative way by taking into account the difference between relevant and non-relevant document vectors in the set of the retrieved ones: at each iteration, a new query is automatically generated in the form of a linear combination of terms from the previous query and terms automatically extracted from both relevant and irrelevant documents. Empirical studies (e.g., [11–13]) have demonstrated that such approach is usually quite effective. Moreover, in their technique, also known as pseudo-relevance feedback or blind feedback, they assume that the top-ranked documents (e.g., top 10, 20 ones) are relevant and could be used for QE.

Similar to QE methods, DE or also document smoothing, can be classified as either global [3, 6] or local [7, 8]. While the global DE made use of domain specific knowledge sources, the local DE focused on the analysis of the sub-collection. For instance, authors in [8] proposed a local DE method for expanding the document with the most informative terms from its neighbor documents, which are identified using a cosine similarity between each document and its neighbors in the collection. Another work in [7] proposed to smooth the document with additional terms collected from the top-ranked documents w.r.t the original document by using three different term selection measures: Term Selection Value [14], Kullback-Leibler Divergence [12], and BM25 [15]. Both of these works are similar in the way that they proposed to combine the local context DE with the local context analysis QE to better improve the IR effectiveness.

In the biomedical domain, several works have been done using both QE [1, 4, 5, 13] and/or DE techniques [3, 6]. The work in [13] adapted the local analysis QE approach for evaluating the IR performance on searching MEDLINE documents. Their approach relies on a blind feedback by selecting the best terms from the top-ranked documents. Candidate terms for QE are weighted using the linear combination of the within-query term frequency and the inverse document frequency according to whether the term appears in the query and/or the document. Using a global approach, the work in [5] investigated QE using MeSH to expand terms that are automatically mapped to the user query via the Pubmed's Automatic Term Mapping (ATM) service, which basically maps untagged terms

from the user query to lists of pre-indexed terms in Pubmed’s translation tables (MeSH, journal and author). Authors in [1] exploited several medical knowledge sources such MeSH, Entrez gene, SNOMED, UMLS, etc. for expanding the query with synonyms, abbreviations and hierarchically related terms identified using Pubmed. Furthermore, they also defined several rules for filtering the candidate terms according to each knowledge source. Differently from the latter, the work in [6] combined both QE and DE using the MeSH thesauri to retrieve medical records in the ImageCLEF 2008 collection. More concretely, they combined an IR-based approach of QE and DE for a conceptual indexing and retrieval purpose. For each MeSH concept, its synonyms and description are indexed as a single document in an index structure. A piece of text, the query to the retrieval system, is classified with the best ranked MeSH concepts. Finally, identified terms denoting MeSH concepts are used to expand both the document and the query. Authors in [4] proposed a knowledge-intensive conceptual retrieval by combining both the global context (i.e., concepts in ontologies) using the Pubmed ATM service and the local context (top-ranked documents) of the query. They reported an improvement rate of about 23% over the baseline.

This paper examines the utility of DE/QE for resolving the term mismatch problem in biomedical IR. Compared to previous works, the major contributions of this work are twofold. First, differently from the work in [6], we propose a novel IR-based concept extraction method by estimating concept relevance for a document by combining both document-to-concept matching degree and document-concept correlation. Second, unlike previous works [1, 3–6], which only focus on QE/DE using the global context (UMLS, MeSH, etc.) or only QE/DE using the local context (corpus-based) [7, 8] or even only QE using both the local and global context [4], we aim to point out that the combination of the document’s global context (MeSH) and the query’s local context (top-ranked documents) may be a source evidence to improve the biomedical IR effectiveness.

### 3 Combining Global and Local Contexts for Biomedical IR

Our retrieval framework is made up of two main components detailed below: (1) global context document expansion and (2) local context query expansion. We integrate them into a conceptual IR process as the combination of the global and local semantic contexts for improving the biomedical IR effectiveness.

#### 3.1 Global Context Document Expansion

In our DE approach, each document is expanded with preferred terms denoting concepts<sup>1</sup> identified using an IR-based document-to-concept mapping method. In other words, given a document, the mapping leads to the selection of the most relevant MeSH concepts using a content-based similarity measure. Furthermore,

<sup>1</sup> In MeSH, each concept is defined by one preferred term (e.g., ‘Skin Neoplasms’) and many non-preferred terms (e.g., ‘Tumors of the Skin’, ‘Skin Cancer’, etc.).

in order to take into account the importance of the word order while matching a concept entry, which can be both a preferred or non-preferred term, to a bounded multi-word term located in a window issued from a document, we propose to leverage the content-based similarity between the document and concept entries using a rank correlation based matching. Our basic assumption behind concept relevance is that a list of document words is more likely to map a concept that (1) both shares a maximum number of words either among its preferred or non-preferred terms; (2) the words tend to appear in the same order so to cover the same meaning. For example, the phrase ‘**Skin cancer**’ represents the most commonly diagnosed disease, surpassing **lung, breasts, ...**’ should be mapped to ‘**Skin cancer**’ rather than ‘**Lung cancer**’, ‘**Breast cancer**’ or ‘**Cancer**’.

Our strategy, which is based on ranking concepts extracted from documents using a combined score, involves three steps detailed below: (1) computing a content-based matching score, (2) computing a rank correlation based score, (3) selecting an appropriate number of terms denoting concepts for document expansion by ranking candidate concepts according to their combined score.

**1. Computing a content-based matching score.** According to our IR based approach, the top-ranked relevant concepts issued from MeSH are assigned to the document. Formally, we compute for each concept vector  $C$  a content-based cosine similarity w.r.t the document  $D$ , denoted  $Sim(C, D)$ , as follows:

$$Sim(C, D) = \frac{\sum_{j=1}^{N_c} c_j * d_j}{\sqrt{\sum_{j=1}^{N_c} c_j^2} * \sqrt{\sum_{j=1}^{N_c} d_j^2}} \quad (1)$$

where  $N_c$  is the total number of concepts in MeSH,  $d_j$  is the weight of word  $w_j$  in document  $D$  computed using an appropriate weighting schema,  $c_j$  is the weight of word  $w_j$  in concept  $C$  computed using the BM25 weighting schema [15]:

$$c_j = tf c_j * \frac{\log \frac{N - n_j + 0.5}{n_j + 0.5}}{k_1 * ((1 - b) + b * \frac{cl}{avcl}) + tf c_j} \quad (2)$$

where  $tf c_j$  is the number of occurrences of word  $w_j$  in concept  $C$ ,  $N$  is the total number of concepts in MeSH,  $n_j$  is the number of concepts containing at least one occurrence of word  $w_j$  in its textual fields,  $cl$  is the length of concept  $C$ , i.e. the total number of distinct words in  $C$ , and  $avcl$  is the average concept length in the MeSH thesaurus,  $k_1$ , and  $b$  are tuning parameters.

**2. Computing a rank correlation coefficient.** The candidate concepts extracted from step 1 are re-ranked according to a correlation measure that estimates how much the word order of a MeSH entry is correlated to the order of words in the document. For this aim, we propose to measure the word order correlation between the concept entry and the document both represented by word position vectors. Formally, the correlation measure is computed using the Spearman operator as follows: let document  $D = (w_{d_1}, w_{d_2}, \dots, w_{d_L})$  be the ranked word based vector according to the average position of related occurrences in document  $D$ , i.e.,  $w_{d_i}$  is the document word in  $D$  such that

$p\bar{o}s(occs(w_{d_i})) < p\bar{o}s(occs(w_{d_{i+1}})) \forall i = 1 \dots L - 1$ , where  $occs(w_{d_i})$  is the set of positions of word  $w_{d_i}$  in document  $D$ ,  $L$  is the total number of unique words in document  $D$ . Similarly, let  $E = (w_{e_1}, w_{e_2}, \dots, w_{e_{L'}})$  be the ranked word based vector according to the average position of related occurrences in concept entry  $E$ , where  $L'$  is the concept entry length. We denote the set of words in  $D$  as  $words(D) = \{w_{d_1}, w_{d_2}, \dots, w_{d_L}\}$  and in concept entry  $E$  as  $words(E) = \{w_{e_1}, w_{e_2}, \dots, w_{e_{L'}}\}$ .

First, in order to avoid false rank bias, when measuring the word order correlation, a portion of the document window bounded by the first and last word occurrences shared by the concept entry  $E$  and the document is captured and normalized as  $D_w = (w_{d_w}, w_{d_{w+1}}, \dots, w_{d_W})$ , where  $words(D_w) \subset words(D)$ ,  $w_{d_w} \in words(E)$ ,  $w_{d_{W+1}} \notin words(E)$ . Afterwards, the Spearman correlation is used to compute the word rank correlation between words in  $D$  and  $E$ :

$$\rho(E, D) = 1 - \frac{6 \sum_i^T [rank(w_i, D_w) - rank(w_i, E)]^2}{T * (T^2 - 1)} \quad (3)$$

where  $rank(w_i, D_w)$  (resp.  $rank(w_i, E)$ ) is the word order of word  $w_i$  according to  $p\bar{o}s(occs(w_{d_i}))$  in  $D_w$  (resp.  $E$ ),  $T = |words(D_w) \cap words(E)|$  is the number of shared words between document  $D$  and concept entry  $E$ . We simply assume that the rank of an absent word in  $D_w$  or  $E$  is assigned a default value  $r_0 > T$ . The correlation coefficient  $\rho(E, D)$  allows measuring the degree of agreement between two word rankings in  $E$  and  $D_w$ . The value of  $\rho(E, D)$  lies between  $-1$  and  $1$  according to the agreement between two rankings. In order to consider each significant entry separately, we practically compute:

$$\rho(C, D) = Max_{E \in Entries(C)} \rho(E, D) \quad (4)$$

where  $Entries(C)$  refers to both preferred or non-preferred terms.

**3. Selecting the candidate concepts for document expansion.** Finally the content based similarity and the correlation between concept  $C$  and document  $D$  are combined in order to compute the overall relevance score  $Rel(C, D)$ :

$$Rel(C, D) = (1 + Sim(C, D)) * (1 + \rho(C, D)) \quad (5)$$

The  $N$  top-ranked concepts with highest scores are selected as candidate concepts of document  $D$ . Preferred terms are used to expand the document content. Document terms are then weighted using the state-of-the-art BM25 model [15].

### 3.2 Local Context Query Expansion

The local context QE applies a blind-feedback technique to select the best terms from the top-ranked expanded documents in the first retrieval stage. In this expansion process, terms in the top-returned documents are weighted using a particular Divergence From Randomness (DFR) term weighting model [12]. In our work, the Bose-Einstein statistics [12] is used to weight terms in the expanded query  $q^e$  derived from the original query  $q$ . Formally:

$$weight(t \in q^e) = tfq_n + \beta * \frac{Info_{\beta, 1}}{MaxInfo} \quad (6)$$

where

- $tfq_n = \frac{tfq}{\max_{t \in q} tfq}$ : the normalized term frequency in the original query  $q$ ,
- $\text{MaxInfo} = \arg \max_{t \in q^e} \max \text{Info}_{\text{Bo1}}$ ,
- $\text{Info}_{\text{Bo1}}$  is the normalized term frequency in the expanded query induced by using the Bose-Einstein statistics, that is:

$$\begin{aligned} \text{Info}_{\text{Bo1}} &= -\log_2 \text{Prob}(\text{Freq}(t|K)|\text{Freq}(t|C)) \\ &= -\log_2\left(\frac{1}{1+\lambda}\right) - \text{Freq}(t|K) * \log_2\left(\frac{1}{1+\lambda}\right) \end{aligned} \tag{7}$$

where  $\text{Prob}$  is the probability of obtaining a given frequency of the observed term  $t$  within the topmost retrieved documents, namely  $K$ ;  $C$  is the set of documents in the collection;  $\lambda = \frac{\text{Freq}(t|C)}{N}$ , with  $N$  is the number of documents in the collection,  $\beta = 0.4$ . The number of top-ranked documents and the number of terms expanded to the original query are tuning parameters.

## 4 Experimental Evaluation

### 4.1 Experimental Data Set

We used the TREC Genomics 2004 collection [16], which is a 10-year subset (1994-2003) of the MEDLINE database, under the Terrier IR platform [17] for validating our conceptual IR approach. Some statistical characteristics of the collection are depicted in Table 1. However, human relevance judgments were merely made to a relative small pool, which were built from the top-precedence run from each of the 27 participants. Our prototype IR system only indexes and searches all human relevance judged documents, i.e. the union of 50 single pools that contains total 42,255 unique articles' titles and/or abstracts. We did not use the set of manually annotated MeSH concepts provided by human experts in our system, but which we referred to as the manual DE task.

In our experiments described later, we used the latest version of MeSH released in 2010, which consists of 25,588 main headings and also over 172,000 entry terms that assist in finding the most appropriate MeSH concepts.

For measuring the IR effectiveness, we used  $P@10$ ,  $P@20$  representing respectively the mean precision values at the top 10, 20 returned documents and  $MAP$  representing the *Mean Average Precision* calculated over all topics.

**Table 1.** TREC Genomics 2004 test collection statistics

Number of documents	4.6 millions
Average document length	202
Number of queries	50
Average query length	17
Average number of relevant docs/query	75

## 4.2 Experimental Design

The purpose of this investigation is to determine the utility of the combination of the global context of the document and the local context of the query. Hence, we carried out two series of experiments: the first one is based on the classical indexing of title and abstract articles using the state-of-the-art weighting scheme BM25 [15], as the baseline, denoted *BM25*. The second one concerns our indexing and retrieval approach and consists of five scenarios:

1. the first one concerns the document expansion using concepts manually assigned by human experts, denoted  $DE^{manual}$  or simply  $DE^m$ ,
2. the second one concerns the document expansion using concepts identified by the combination of the cosine content-based similarity and the Spearman rank correlation between word occurrences in the document and concept entries (see section 3.1, formula 5), denoted  $DE^{combination}$  or simply  $DE^c$ ,
3. the third one concerns the query expansion using the blind feedback technique applied on the original document (title and abstract) without DE (see section 3.2, formula 6), denoted *QE*.
4. the fourth one concerns the combination of both *QE* and the manual  $DE^m$  method, denoted  $QE + DE^m$ ,
5. the last one concerns our method which relies on the combination of both *QE* and our automatic  $DE^c$  strategy as described above, denoted  $QE + DE^c$ .

## 4.3 Results and Discussion

First, we aim to measure the impact of the number of expanded terms on the IR effectiveness by tuning the number of terms denoting concepts expanded to the document for DE and the number of terms from the top-ranked documents expanded to the original query for QE. Second, we will measure the IR effectiveness using the optimal number of expanded terms for QE and/or DE. Finally, we compare our IR results to the official ones in TREC 2004 Genomics Track.

**Impact of the number of expanded terms.** For automatic DE, we tuned the number of candidate concepts from 0 to 50, with a step of 5. Query terms in the expanded documents are weighted using the state-of-the-art BM25 model [15]. Table 2 shows the MAP results achieved by our document expansion method, namely  $DE^c$ . The results show that our document expansion method achieves the best MAP (0.4118) when expanding with  $N = 5$  terms denoting concepts to the document content. We observed that the query space (i.e., 50 ad hoc topics) usually contains synonyms (non-preferred terms, e.g., ‘FANCD2’, ‘ache’, ‘breast cancer’, etc.) of medical concepts while the document space has been adjusted using their preferred terms (e.g., Fanconi Anemia Complementation Group D2 Protein, Pain, Breast Neoplasms, etc.). Therefore, we believe that picking up some related terms from the expanded documents returned for each topic would better increase the term overlap between the expanded query and the expanded documents in the collection. We retain  $N = 5$  for the experiments described in the next section.



**Table 2.** Document expansion: P@10, P@20, MAP results by varying  $N$  from 0 to 50

N	P@10	P@20	MAP
0	<b>0.5920</b>	0.5380	0.4097
<b>5</b>	0.5780	<b>0.5390</b>	<b>0.4118</b>
10	<b>0.5920</b>	0.5280	0.4031
15	0.5840	0.5330	0.3999
20	0.5660	0.5290	0.4032
25	0.5660	0.5250	0.4007
30	0.5600	0.5150	0.3975
35	0.5400	0.5070	0.3918
40	0.5340	0.5020	0.3882
45	0.5300	0.4940	0.3855
50	0.5280	0.4880	0.3835

**Table 3.** Query expansion: MAP results by varying the number of expanded terms/docs

Nb. terms \ Nb. docs		5	10	15	20	25
		5	0.4369	0.4347	0.4455	0.4422
10	0.4204	0.4232	0.4286	0.4289	0.4332	
15	0.4357	0.4407	0.4431	0.4463	0.4428	
<b>20</b>	0.4373	0.4395	0.4454	<b>0.4475</b>	0.4467	
25	0.4347	0.4403	0.4429	0.4448	0.4473	

For automatic QE, the number of expanded terms and the number of selected documents are tuned from 5 to 25 with a step of 5. Query terms in the original documents are weighted using the state-of-the-art BM25 model [15]. Table 3 depicts the MAP results of 50 ad hoc topics. The best results are obtained at 20 terms and 20 top-ranked documents. Therefore, we retain 20 terms and 20 documents for the next experiments described later.

**Retrieval effectiveness.** At this level, we aim to study the impact of the combination of the global context of the document and the local context of the query. For this purpose, we measure the IR effectiveness of five retrieval scenarios described in section 4.2. We now present and discuss the experimental results.

Table 4 shows the IR effectiveness of 50 ad hoc topics. According to the results, we observe that both the manual and automatic DE methods ( $DE^m$ ,  $DE^c$ ) slightly outperform the baseline in terms of MAP, but both of these methods do not improve the IR performance in terms of P@10 and P@20. The difference between these two methods is about the number of terms expanded to the document, a dozen for the manual DE [16] and five for the automatic DE. Automatic QE using related terms, which may denote domain concepts or not, from top-ranked documents improves better the MAP. As argued in this study, we can see that the combination of the local context of the query and the global context of

<sup>2</sup> Only preferred terms are used for document expansion.

**Table 4.** IR effectiveness in terms of  $P@10$ ,  $P@20$ ,  $MAP$  (%change)

	<b>P@10</b>	<b>P@20</b>	<b>MAP</b>
<i>BM25</i>	<b>0.5920</b>	0.5380	0.4097
<i>DE<sup>m</sup></i>	0.5900 (-00.34)	0.5370 (-00.19)	0.4139 (+01.03) †††
<i>DE<sup>c</sup></i>	0.5780 (-02.36)	0.5390 (+00.19)	0.4118 (+00.51)
<i>QE</i>	0.5720 (-03.38)	0.5430 (+00.93)	0.4475 (+09.23)
<i>QE + DE<sup>m</sup></i>	0.5320 (-10.14) ††	0.5220 (-02.97)	<b>0.4567</b> (+11.47)
<i>QE + DE<sup>c</sup></i> (our method)	0.5860 (-01.01)	<b>0.5470</b> (+01.67)	0.4532 (+10.62) †††

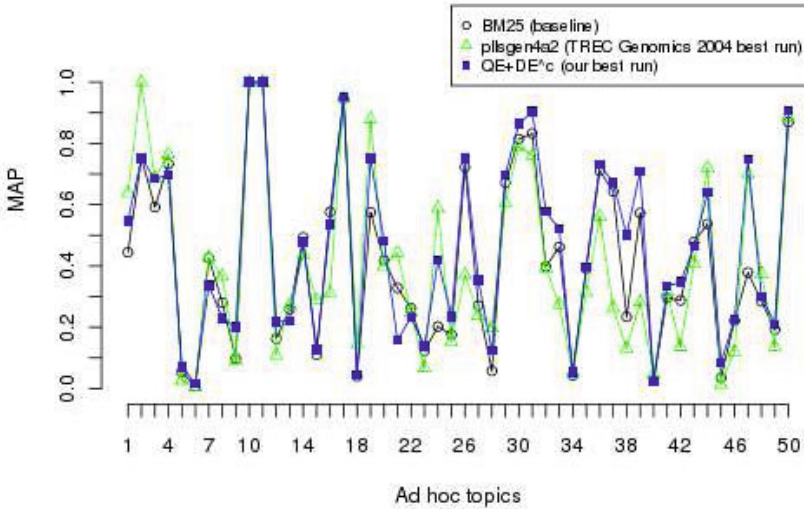
Paired sample t-test: † significant ( $p < 0.05$ ), †† very significant ( $p < 0.01$ ), and ††† extremely significant ( $p < 0.001$ ).

the document is helpful for improving much better the IR performance in terms of MAP. Indeed, as depicted in Table 4, the combination of the QE and  $DE^m$  shows a gain of +11.47% in terms of MAP over the baseline. However, the precision values of this combination are dramatically decreased. The reason could be explained as follows: in general, long queries (17 terms in average) are enough to describe the user information need, therefore expanding *related terms* to a long query may improve the recall but not the precision. Furthermore, expanded terms in the document are preferred terms while the ones in reformulated query may be terms denoting domain concepts or not. Therefore, the query space and document space are not correctly adjusted for increasing the term overlap. Our document expansion method for detecting domain concepts revealing the document subject matter(s) (the global context of the document) enhanced with the local context of the query allows to retrieve more relevant documents than the baseline as well as document expansion alone or query expansion alone. The highest MAP value of our method  $QE + DE^c$  is obtained at 0.4532 with an improvement rate of +10.62% even though the precision values are slightly different compared to the baseline. As shown in Table 4, the paired-sample T-test ( $M = 4.35\%$ ,  $t = 3.5291$ ,  $df = 49$ ,  $p = 0.0009$ ) shows that the combination of the global context of the document and the local context of the query, i.e.  $QE + DE^c$  method, is extremely statistically significant compared to the baseline.

**Comparative evaluation.** We further compare the IR performance of our best automatic retrieval method ( $QE + DE^c$ ) to official runs in TREC Genomics 2004. Table 5 depicts the comparative results of our best run with official runs of participants in the TREC 2004 Genomics Track. The results show that the precision values ( $P@10$ ,  $P@20$ ) of our best run are better than the third run but lower than the two first runs. However, the MAP of our run is much better than the first run. As shown in Figure 1, our best indexing and retrieval method ( $QE + DE^c$ ) outperforms the best run submitted to TREC Genomics 2004 ( $MAP=0.4075$ ) with a gain of +11.21%. Thus, we conclude that conceptual indexing and searching in conjunction with an efficient way of identifying appropriate concepts representing the semantics of the document as well as of the query would significantly improve the biomedical IR performance.

**Table 5.** The comparison of our best run with official runs participated in TREC 2004 Genomics Track. Runs in TREC are ranked by Mean Average Precision (MAP).

Run	P@10	P@20	MAP
pllsgen4a2 (the best)	0.6040	0.5720	0.4075
uwmtDg04tn (the second)	<b>0.6240</b>	<b>0.5810</b>	0.3867
pllsgen4a1 (the third)	0.5700	0.5430	0.3689
PDTNsmp4 (median)	0.4056	0.4560	0.2074
edinauto5 (the worst)	0.0360	0.0310	0.0012
<b>QE+DE<sup>c</sup> (our best run)</b>	0.5860	0.5470	<b>0.4532</b>

**Fig. 1.** The comparison in terms of MAP of our runs (Combination approach and the baseline) to the official best run in TREC 2004 Genomics Track on 50 ad hoc topics

## 5 Conclusion

In this paper, we have proposed a novel IR method for combining the global context DE and the local context QE. Our automatic DE relies mainly on turning the concept mapping into a concept retrieval task by means of concept relevance scoring. The QE technique relies on the selection of related terms from the top-ranked documents. The results demonstrate that our IR approach shows a significant improvement over the classical IR as well as DE or QE as alone. The performance of our approach is also significantly superior to the average of official runs in TREC 2004 Genomic Track and is comparable to the best run.

For future work, we plan to improve the precision of our concept extraction method, which will integrate the concept centrality and specificity. We believe that these two factors allow to overcome the limits of the bag-of-words based similarity by leveraging lexical and semantic contents of candidate concepts.

## References

1. Stokes, N., Li, Y., Cavedon, L., Zobel, J.: Exploring criteria for successful query expansion in the genomic domain. *Information Retrieval* 12(1), 17–50 (2009)
2. Voorhees, E.M.: Query expansion using lexical semantic relations. In: *SIGIR 1994 Conference on Research and Development in Information Retrieval*, pp. 61–69 (1994)
3. Le, D.T.H., Chevallet, J.P., Dong, T.B.T.: Thesaurus-based query and document expansion in conceptual indexing with umls. In: *RIVF 2007*, pp. 242–246 (2007)
4. Zhou, W., Yu, C.T., et al.: Knowledge-intensive conceptual retrieval and passage extraction of biomedical literature. In: *SIGIR*, pp. 655–662 (2007)
5. Lu, Z., Kim, W., Wilbur, W.J.: Evaluation of query expansion using mesh in pubmed. *Information Retrieval* 12(1), 69–80 (2009)
6. Gobeill, J., Ruch, P., Zhou, X.: Query and document expansion with medical subject headings terms at medical imageclef 2008. In: Peters, C., Deselaers, T., Ferro, N., Gonzalo, J., Jones, G.J.F., Kurimo, M., Mandl, T., Peñas, A., Petras, V. (eds.) *CLEF 2008. LNCS*, vol. 5706, pp. 736–743. Springer, Heidelberg (2009)
7. Billerbeck, B., Zobel, J.: Document expansion versus query expansion for ad-hoc retrieval. In: *The 10th Australasian Document Comput. Symp.*, pp. 34–41 (2005)
8. Tao, T., Wang, X., et al.: Language model information retrieval with document expansion. In: *Association for Computational Linguistics*, pp. 407–414 (2006)
9. Sparck Jones, K.: *Automatic Keyword Classification for Information Retrieval*. Butterworths, London (1971)
10. Rocchio, J.: Relevance Feedback in Information Retrieval, pp. 313–323 (1971)
11. Xu, J., Croft, W.B.: Query expansion using local and global document analysis. In: *Conference on Research and Development in Information Retrieval*, pp. 4–11 (1996)
12. Amati, G.: Probabilistic models for Information Retrieval based on Divergence from Randomness. PhD thesis, University of Glasgow (2003)
13. Abdou, S., Savoy, J.: Searching in medline: Query expansion and manual indexing evaluation. *Information Processing Management* 44(2), 781–789 (2008)
14. Robertson, S.E., Walker, S.: Okapi/keenbow at trec-8. *TREC* 8, 151–162 (1999)
15. Robertson, S.E., Walker, S., Hancock-Beaulieu, M.: Okapi at trec-7: Automatic ad hoc, filtering, vlc and interactive. In: *TREC-7 Proceedings*, pp. 199–210 (1998)
16. Hersh, W., Bhuptiraju, R.: Trec 2004 genomics track overview. In: *The Thirteenth Text Retrieval Conference, TREC 2004* (2004)
17. Ounis, I., Lioma, T.: Research directions in terrier. In: Baeza-Yates, R., et al. (eds.) *Novatica Special Issue on Web Information Access* (2007) (invited paper)

# Smoothing Click Counts for Aggregated Vertical Search

Jangwon Seo<sup>1</sup>, W. Bruce Croft<sup>1</sup>, Kwang Hyun Kim<sup>2</sup>, and Joon Ho Lee<sup>2</sup>

<sup>1</sup> CIIR, University of Massachusetts, Amherst, MA, 01003, USA

<sup>2</sup> NHN Corporation, 463-824, Korea

{jangwon,croft}@cs.umass.edu, {khkim,joonho}@nhn.com

**Abstract.** Clickthrough data is a critical feature for improving web search ranking. Recently, many search portals have provided *aggregated search*, which retrieves relevant information from various heterogeneous collections called *verticals*. In addition to the well-known problem of rank bias, clickthrough data recorded in the aggregated search environment suffers from severe sparseness problems due to the limited number of results presented for each vertical. This skew in clickthrough data, which we call *rank cut*, makes optimization of vertical searches more difficult. In this work, we focus on mitigating the negative effect of rank cut for aggregated vertical searches. We introduce a technique for smoothing click counts based on spectral graph analysis. Using real clickthrough data from a vertical recorded in an aggregated search environment, we show empirically that clickthrough data smoothed by this technique is effective for improving the vertical search.

## 1 Introduction

Clickthrough data is invaluable information that records user actions in response to search results. Recently, there have been a number of efforts using clickthrough data to improve ranking. For example, by analyzing clickthrough data, we can discover the users' preferences for certain search results. These preferences can be used to evaluate search systems [14] or to be incorporated for ranking functions [2]. These approaches have proved to be effective for optimizing web search ranking based on a single web repository.

Many search portals, however, now provide *aggregated* rankings based on various domain-specific collections, e.g., news, blogs, community-based question answering (CQA), images, etc. These domain-specific collections are often called *verticals*. There is a separate index for each vertical and a search engine optimized for the vertical returns its own results. Aggregation logic in the search portal selects relevant verticals and displays results returned from each vertical in a result page. This is referred to as *aggregated search*.

Clickthrough data of each vertical recorded in aggregated search has some different properties than the data recorded in a typical web search. A result page is a limited resource that the verticals share. In many cases, even if a vertical is relevant, the vertical cannot have more than a few results in an aggregated result

page. Consequently, the clickthrough data can suffer from significant distortion. For example, Figure 1 shows the click count distribution of a vertical where only the top 5 results are delivered to aggregated search. We refer to the number of results returned to aggregated search as the *cut-off rank*. Clearly, there is a steep attenuation of click counts after the cut-off rank. We refer to the attenuation caused by the cut-off rank as *rank cut*.

Previous studies on web search have reported that clickthrough data suffers from biases such as *rank bias* [14]. Generally, since rank bias is caused by users' scanning behavior, click counts affected by rank bias show a smoothly decreasing curve as the rank decreases. While we can observe rank bias in the top ranks in Figure 1, the most dominant phenomenon is rank cut, which is caused by a system parameter (cut-off rank) rather than user behavior. That is, if rank cut exists, users have limited views of results from verticals. Then, while a click on a document may be considered as a signal that the document is likely to be relevant, more clicks on a document do not imply that the document is more relevant. If we want to optimize vertical search using clickthrough data, this phenomenon can be problematic. The problem of rank cut is not limited to any specific type of aggregated search. That is, whether interleaving or grouping results for aggregated search, as long as only a few results from each of many verticals are selected for display in a result page, rank cut may occur.

Of course, there is a similar cut-off caused by displaying results in pages in general web search. In web search, however, a page delivers more results (usually, 10) and users can easily see results beyond the current page by following the link to the next page. On the other hand, in aggregated search, verticals typically return at most five results and often only one or two results are returned. Furthermore, in an aggregated result page, results from other verticals follow each other or are interleaved with each other. Accordingly, when users want to see more results, they naturally see results from other verticals rather than lower ranked documents from the same vertical. Therefore, the effect of rank cut is more noticeable in aggregated search.

Some search portals provide an option where users can issue queries directly to vertical search engines. If a large volume of queries are input through separate vertical search interfaces, we can use this clickthrough data for vertical search optimization without considering aggregated search. However, not all verticals have their own search interfaces and even if they did, sufficient clickthrough data may not be collected because aggregated search is the default search option in most cases.

Therefore, in this paper, we focus on improving vertical search using clickthrough data by mitigating a negative aspect of clickthrough data recorded in aggregated search, i.e. rank cut. To address rank cut, we introduce a technique based on spectral graph analysis (*click count regularization*). Using a real vertical collection and its click log data recorded in aggregated search, we study the effectiveness of this click count smoothing technique for vertical search optimization.

## 2 Related Work

To the best of our knowledge, there is no work that explicitly addresses skews in click distributions of verticals in aggregated search. On the other hand, there is some prior work on exploiting clickthrough data in vertical search. Li et al. [16] use click graphs to classify query intent for vertical search. Their work is similar to our work in that both use semi-supervised learning techniques based on clickthrough data. However, while they use click graphs to expand a volume of labeled queries for the purpose of training, we regularize click counts on document graphs.

There have been many efforts to optimize general Web search using clickthrough data. Joachims [12] introduces the use of clickthrough data to learn ranking functions. Joachims et al. [14] investigate various biases in clickthrough data by user behavior analysis and propose extracting implicit relevance feedback from the data. Dupret and Piwowarski [9] and Chapelle and Zhang [5] analyze browsing behaviors of users in response to search results using click models that take bias into account.

Recent work by Gao et al. [10] focus on the sparseness problem of clickthrough data. They introduce a random walk algorithm based on click graphs and a discounting technique to expand clickthrough data. Since the rank cut problem can be considered as a kind of sparseness problem, their work is the closest to our work. However, their work differs from ours in that they use non-content-based algorithms and their domain is limited to web search where the sparseness problem is often less serious than aggregated vertical search. In other related work, Radlinski et al. [19] propose an active exploration method to tackle the sparseness problem caused by rank bias.

Click count regularization can be considered as a label propagation algorithm [21] [24] [4]. Label propagation is a semi-supervised learning algorithm which leverages labeled data to classify unlabeled data. In particular, it assumes that if two instances in a high-density region are close, their corresponding values are similar. In addition, Diaz [8] leverages a graph Laplacian-based regularization technique for re-ranking tasks in Information Retrieval. Similar to our work, he builds document-content-based graphs. However, the objective to be regularized is a document score vector. Moreover, a different cost and an optimization technique are used.

## 3 Data Set

In this paper, as a data set for experiments, we use a snapshot of the community-based question answering (cQA) vertical<sup>1</sup> of Naver, which is a major commercial search portal in Korea. The snapshot which contains about 30 million documents was directly downloaded from the service databases. Since the CQA service is the most popular service in Naver and the most clicks occur on search results from the CQA vertical in aggregated search, we choose this data set. However,

<sup>1</sup> <http://kin.naver.com/>

since we want to look at the collection as a vertical in aggregated search, we do not address unique features associated with CQA collections.

Naver provides an aggregated search service. In this service, there are various verticals, e.g., news, CQA, blog, forum, image, video, book, etc. Each vertical search engine retrieves its own relevant items for a user query. The results are grouped according to the corresponding verticals and each group is ranked by a vertical ranking algorithm. Note that there are various ways to aggregate search results from verticals. The results can be interleaved with each other or grouped by each corresponding vertical. Aggregated search provided in Naver uses the latter. Currently, the CQA vertical returns 5 search results to the aggregated search interface if the CQA vertical is determined to be a relevant vertical by a vertical selection algorithm. In addition, although Naver provides a separate search interface for each vertical, aggregated search is the default and most users use aggregated search even when their queries are more suitable for a specific vertical [17].

To establish a test collection for the CQA vertical, we chose 972 of the most frequent queries input to the Naver aggregated search interface, considering the following: i) the queries should be able to address as many different topics as possible, ii) the length distribution of the queries should be close to that of real user queries. For each query, the top 50 documents retrieved by the current Naver CQA vertical search engine are considered as a document set for relevance judgments. We asked 20 editors to judge them on a four point scale: non-relevant, partially relevant, relevant and highly relevant. In total, we made 46,270 relevance judgments. Note that there are few overlaps between the judgment sets of different topics. We use 500 queries as a training set and the remaining 472 queries as a test set.

We use query logs and click logs which were collected for one week - from Aug 26, 2008 to Sep 01, 2008 - through both the aggregated search interface and the CQA vertical search interface in Naver.

An entry of the query logs contains the following information:

$$\langle q, v_1[(d_{11}, r_{11}), \dots, (d_{1n}, r_{1n})], v_2[(d_{21}, r_{21}) \dots], \dots \rangle$$

where  $q$  is a user query,  $v_i$  is a vertical ID,  $d_{ij}$  is a document ID in  $v_i$ , and  $r_{ij}$  is a rank of  $d_{ij}$ . Therefore, from the query logs, we can know which documents are returned to users.

An entry of the click logs contains the following information:

$$\langle q, v, d, r \rangle$$

where  $q$  is a user query, and  $v$  and  $r$  are a vertical and a rank of clicked document  $d$ , respectively. Therefore, from the click logs, we can know which documents are clicked for a query.

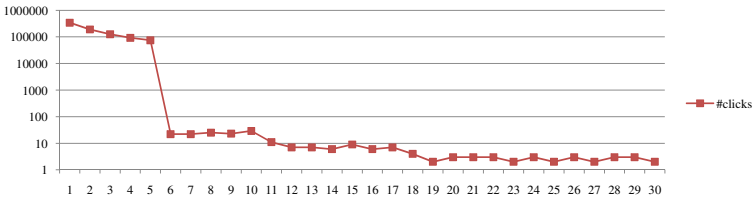
We filtered out the click log entries which do not include any result from the CQA vertical. Then we took click logs whose queries are contained in the query set of the test collection. To compare queries in the logs with queries in the test collection, we stemmed the queries using a Korean stemmer used for the



current Naver search engine. Consequently, we obtained about 3.3 million query log entries corresponding to the test collection. We applied the same process to the click logs and obtained about 1 million click log entries.

Note that only 178 test queries appear in click data collected from the separated CQA vertical search while all test queries appear in click data from the aggregated search. This is because most users input queries to the aggregated search interface. A similar situation can happen in case that a separated vertical search interface is not popular or cannot be provided to users. Then, using only the vertical click data seems inappropriate although the data is relatively free from some artifacts such as rank cut. Accordingly, in this work, we merged click data from the aggregated search interface and the separated CQA vertical search interface.

To observe skewness in click log data, for each rank, we summed all click counts over all queries in the test collection. Figure 1 presents the distribution of click counts according to ranks. As we see, the only top 5 results dominate significantly the entire click counts. This shows that the effect of rank cut can be serious.



**Fig. 1.** Distribution of click counts of a vertical recorded via aggregated search and separated vertical search according to ranks. The horizontal axis is the rank and the vertical axis is the click count. The aggregated search interface delivers the top 5 results from the vertical to users.

## 4 Click Count Regularization

One problem that can be attributed to the rank cut is that beyond a cut-off rank, there can be relevant documents which could have been clicked if they had been delivered to users. We solve this problem by predicting the click counts of such documents.

Our motivation is that if two documents are similar with respect to any measure and one of them is clicked for a query, then the other is likely to be clicked for the same query. This can be seen as an interpretation in terms of click counts of the cluster hypothesis by van Rijsbergen, that states “closely associated documents tend to be relevant to the same requests” [22].

Let us consider a click count itself as a label [2]. A document which has been returned to users can be considered as a labeled instance whereas a document

<sup>2</sup> We do not claim that a click is considered as a relevance label. A label here is a metaphor for explaining a semi-supervised learning setting.

beyond the cut-off rank can be considered as an unlabeled instance. Then, we would like to predict labels (click counts) of the unlabeled instances. This task is similar to semi-supervised learning tasks such as label propagation [21] [24] [4]. In this section, for ease of explanation, we refer to documents which have been returned to users as labeled documents and to the other documents as unlabeled documents. To determine if a document is labeled, we count query log entries which contain the document. If the document has been returned to users more than  $E$  times, we consider the document labeled. Note that some labeled documents may not receive any clicks. In the regularization process, these documents presumably contribute to non-relevant documents having small click counts. We set  $E = 100$  in this paper because most documents ranked above the cut-off of rank cut have been presented to users more than 100 times in our data set.

In order to determine relationships between unlabeled documents and labeled documents, we specify how to measure similarity between documents. For example, the heat kernel has good properties as a similarity measure [15]:

$$K_{heat}(x_1, x_2) = (4\pi t)^{-n/2} \exp(-(4t)^{-1} distance(x_1, x_2)^2) \quad (1)$$

where  $t$  is a parameter. Since we will use the graph Laplacian which is closely related to the heat equation [3] in this work, the Gaussian form of the heat kernel is a favored choice as a similarity measure. Considering the heat kernel, we define two similarity measures based on different distances: topic similarity and quality similarity.

For topic similarity, we assume that a document  $D$  can be represented by a multinomial distribution  $\theta = (tf_{w_1}/|D|, tf_{w_2}/|D|, \dots, tf_{w_n}/|D|)$ , where  $tf_{w_k}$  is a term frequency of term  $w_k$  and  $n$  is the size of vocabulary. Then, we can define the geodesic distance on the  $n$ -simplex by  $2 \arccos(\sqrt{\theta_1} \cdot \sqrt{\theta_2})$  on a manifold of multinomial models with the Fisher information metric [15]. Using this distance, the heat kernel becomes the multinomial diffusion kernel that has been proved to be effective in many tasks [15].

For quality similarity, we define a quality distance by  $1 - \min(q_1, q_2)$ , where  $q_k$  is a quality score which has a range  $[0,1]$ . The quality score can be any quality estimate such as PageRank [18]. We here use a quality estimate for a CQA vertical which is similar to what has been done by Jeon et al. [11]. The distance implies that if the qualities of both documents are good, the documents are close. That is, click diffusion is assumed to occur only between good quality documents.

We now build an affinity matrix  $W$  based on each similarity measure. One of the most straightforward approaches to build an affinity matrix is to compute the similarity between all documents in a collection. However, this is infeasible in case of a large collection. Instead, since we are interested in documents whose ranks are not high enough to be above cut-off rank  $R$ , we compute the similarity between the top  $N$  documents, where  $N \gg R$ . An  $N$  by  $N$  affinity matrix is constructed as follows:

$$W_{ij} = \begin{cases} K(x_i, x_j) & i \neq j \ \& \ K(x_i, x_j) > T \\ 0 & \text{otherwise} \end{cases}$$

where  $T$  is a threshold to make the affinity matrix sparse. Without the threshold, click counts can be diffused to unrelated documents because similarity values are not often 0, even when documents are not similar.

We may want to combine multiple similarity measures to make an affinity matrix. For example, two documents which are similar in terms of both their topics and quality may reveal a stronger connection than other two documents which are similar in terms of only quality. We combine  $M$  affinity matrices computing a geometric mean to preserve sparsity, that is, the combined entry is 0 if any entry in matrices to be combined is 0.

We now define a cost to be minimized by regularization, following the quadratic cost criterion by Joachims [13] and Bengio et al. [4].

$$C(\hat{f}) = \|I_{[l]}(\hat{f} - f)\|^2 + \pi \hat{f}^T L \hat{f} + \epsilon \|\hat{f}\|^2$$

where  $I_{[l]}$  is a diagonal matrix where  $I_{[l]}_i = 1$  only if  $i$ th column (or row) corresponds to a labeled document and all other entries are 0,  $f$  is an original click count vector,  $\hat{f}$  is a regularized click vector,  $L$  is the un-normalized graph Laplacian ( $L = D - W$ , where  $D$  is the diagonal degree matrix given by  $D_i = \sum_j W_{ij}$ ), and  $\pi$  and  $\epsilon$  are parameters which are greater than 0.

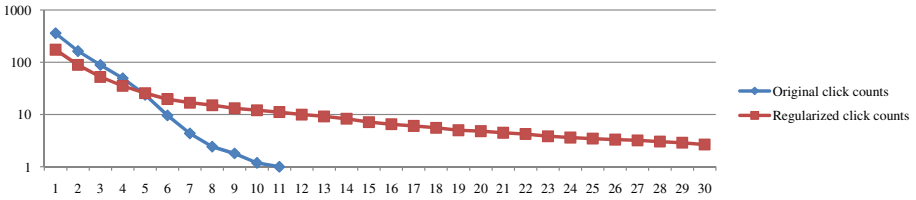
The first term of the cost is a constraint to minimize the difference between the regularized click counts and the original click counts of the labeled documents. The second term is the Dirichlet sum, which expresses smoothness on the underlying manifold [6]. The smaller the Dirichlet sum is, the smoother  $f$  is. This term is crucial because the notion of smoothness implies that two similar documents have similar click counts. The weight of this term can be controlled by the parameter  $\pi$ . The last term controls a size of  $\hat{f}$  and gives numerical stability. The weight of this term can be controlled by the parameter  $\epsilon$ .

Following Bengio et al. [4], we use the Jacobi method [11] to solve this linear system. Then, the approximated solution can be written as follows:

$$\hat{f}_i^{(t+1)} = \frac{1}{I_{[l]}_i + \pi \sum_j W_{ij} + \epsilon} \left( I_{[l]}_i f_i + \pi \sum_{j \neq i} W_{ij} f_j^{(t)} \right)$$

We can guarantee convergence of the Jacobi method as a strictly diagonally dominant matrix is used. In our experiments, most runs quickly converged with fewer than 10 steps.

We refer to this process as “click count regularization”. We call  $\hat{f}$  after convergence the regularized click count vector. Each value in  $\hat{f}$  is referred to as a regularized click count of the corresponding document. Figure 2 presents the effect of click count regularization. As we see, before regularization, the click count curve sharply decreases and only a few documents have clicks. On the other hand, after regularization, we can observe slow attenuation and clicks on more documents.



**Fig. 2.** Change of click counts after click count regularization. Clicks on documents for each query in the testing set are sorted in decreasing order and averaged over all testing queries. The horizontal axis is the sorted order and the vertical axis is the average click counts.

## 5 Experiments and Discussion

We carried out the following experiments to evaluate effectiveness of the click count regularization technique in vertical search.

### 5.1 Features and Learning

Two types of features are used for ranking experiments: text-based features and click-based features. Text-based features are derived from a bag-of words language modeling approach for Information Retrieval [7]. Specifically, we use the Dirichlet-smoothed unigram language model [23]. For implementation of the language model, we used the Indri search engine [20]. All documents are stemmed by a Korean stemmer used for the current Naver search engine. We make four language model-based features by applying different normalization factors or taking the logarithm of the query-likelihood score.

Click-based features are derived from the ratio of a click count on a document to the total click count for a given query. Similar to text-based features, we have five different features via manipulation by different normalization factors and the logarithm function.

For learning to rank with these features, we use the rank SVM algorithm [12].

### 5.2 Baselines and Evaluation

We consider a weak baseline and two strong baselines for evaluation and comparison. The weak baseline is a model which does not use any click count-based feature. That is, this is not different from the unigram language model. The first strong baseline (strong baseline 1) is a model which uses features based on the original click counts. That is, this baseline produces rankings in case that any click count smoothing technique is not employed. The second strong baseline (strong baseline 2) uses features based on click counts smoothed by a state-of-the-art smoothing technique [10] instead of the original click counts. This technique leverages a random walk algorithm on click graphs and a discounting technique inspired by the Good-Turing estimator. Gao et al. [10] have

demonstrated that this technique is effective for addressing sparseness of click-through data for Web search. Note that in contrast to our proposed technique, this technique uses click graphs extracted from click logs without considering document contents.

We evaluate the proposed click count regularization technique using the test collection described in Section 3. For each query, we initially rank 50 documents using the unigram language model on the test collection and re-rank the documents by a learned ranking model incorporating click count-based features. To build affinity matrices, we consider ‘topic + quality’ combination as well as topic similarity. We use four evaluation metrics: normalized discounted cumulative gain at 5 and at 10 (NDCG@5 and NDCG@10), and mean average precision (MAP). This scores are computed using relevance judgments on the CQA vertical described in Section 3.

In order to tune models on the training queries, we find parameter values which maximize NDCG at 5. A Dirichlet smoothing parameter  $\mu$  was set to 2000 since the value has showed the best performance for the weak baseline. For click count regularization, parameters to be tuned are a similarity threshold  $T$  ( $\in [0.1, 0.9]$ ), matrix combination parameters  $\alpha$ 's ( $\in [0.1, 0.9]$ ), and two regularization parameters,  $\pi$  and  $\epsilon$  ( $\in [0.1, 1]$ ). To test the statistical significance of an improvement, we perform a paired randomization test with  $p$ -value  $< 0.05$ .

### 5.3 Results

Table [1](#) shows the experimental results. Not surprisingly, even without using the click count smoothing technique, the model incorporating click count-based features (strong baseline 1) outperforms the model incorporating only unigram language model-based features (weak baseline). This is in part because the Naver CQA vertical search engine used for collecting clickthrough data outperforms the weak baseline. Since click count-based features are extracted from clicks on documents highly ranked by the vertical search engine, the performance of the strong baseline can be assumed to be similar to or better than that of the real vertical search engine. Consequently, all models incorporating click count-based features significantly outperform the weak baseline.

Interestingly, the model using click counts smoothed by the random walk and discounting technique (strong baseline 2) fails to show any significant difference from the model using the original click counts (string baseline 1). The random walk/discounting technique has been proved to be effective for sparseness problems in Web search clickth logs, but click logs for verticals in aggregated search are often much sparser than click logs from Web search because of the effect of rank cut. Therefore, relying only on sparse log information without considering contents of documents may not be enough for addressing highly sparse click logs.

Click count regularization shows consistent improvements over both strong baselines. Moreover, all improvements are statistically significant. When using only the topic similarity-based affinity matrix, regularization shows statistically significant improvements on the strong baseline for all evaluation metrics. When using both of topic similarity and quality similarity, the results show the best

**Table 1.** Experimental results for evaluating effectiveness of regularized click counts. All results by employing click count regularization are statistically significantly better than the weak baseline and two strong baselines.

	NDCG@5	NDCG@10	MAP
Text [weak baseline]	0.4944	0.5204	0.4476
Text + Original Click Counts [strong baseline 1]	0.6261	0.6160	0.5127
Text + RandomWalk/Discounting [strong baseline 2]	0.6177	0.6159	0.5110
Text + Regularized Click Counts			
topic	0.6327	0.6254	0.5252
topic+quality	0.6327	0.6260	0.5259

**Table 2.** Experimental results for evaluating effectiveness of regularized click counts when only clicks on a document for a query are allowed. All results by employing click count regularization are statistically significantly better than the weak baseline and two strong baselines.

	NDCG@5	NDCG@10	MAP
Text [weak baseline]	0.4944	0.4108	0.4476
Text + Original Click Counts [strong baseline 1]	0.5508	0.4225	0.4749
Text + RandomWalk/Discounting [strong baseline 2]	0.5592	0.4271	0.4769
Text + Regularized Click Counts			
topic	0.5739	0.4360	0.4915
topic+quality	0.5747	0.4373	0.4921

performance for most evaluation metrics although the improvements on the results only by topic similarity are somewhat marginal. This shows that given that two documents are topically similar, the fact that their quality is equally high may add a hint about the closeness of documents in terms of click-likelihood.

## 5.4 Robustness

The CQA vertical that we used returns 5 results in an aggregated search result page. Indeed, in many real aggregated search applications, five results are quite a lot, and it could be argued that this number of results should be sufficient to collect user behavior information without click count regularization. However, the number of verticals tends to increase every year and the page length that users are willing to scan is not long. Therefore, there is a good chance that search portals will reduce the number of results from each vertical in the aggregated view. Indeed, we already find that only one or two results from a vertical are displayed in aggregated search services of major search portals. Furthermore, in the mobile search environment, the resources for displaying search results are significantly more limited.

We want to see whether our smoothing technique works in such extreme but still likely cases. To simulate this situation, we get rid of most records from the click logs so that click logs for only one document with the largest number of clicks for each query remains. Although we keep only one document with the largest click

count to simulate a situation where the cut-off rank is 1, this setting should be somewhat relaxed in a real situation because dynamic features of ranking functions may cause other documents to be ranked at 1 over a time span. However, for now, we just assume a simple situation, that is, only one document for a query in click logs.

Using this modified click log, we repeated the same experiments. The same parameters trained with the original click logs are used. Table 2 shows the results. As we see, click regularization works well even when there are clicks on only one document for a query. Click count regularization leads to statistically significant improvements on the strong baselines in all metrics without regard to types of affinity matrices. Furthermore, in this extremely sparse setting, performance gain of our regularization techniques over the strong baselines becomes noticeable. In sum, these results show the robustness of click count regularization in that the technique can address even clickthrough data collected when a more strict resource constraint for aggregated search results is imposed on a vertical search engine.

## 6 Conclusion

In this paper, we described skews that exist in click log data of a vertical recorded in an aggregated search interface, i.e. rank cut, in addition to the well-known rank bias. In particular, rank cut can cause a serious sparseness problem for clickthrough data. To address these issues, we proposed click count regularization as a click count smoothing technique. This technique addresses rank cut using spectral graph analysis. Through experiments, we demonstrated that click count regularization can yield significant improvements compared to a strong baseline. Furthermore, the robustness of click count regularization was empirically shown by experiments in a simulated situation with only a single retrieved document.

For future work, we will consider various types of queries and verticals. In this work, we focused on a general framework to address skews in vertical clickthrough data and somewhat ignored the various properties of queries and verticals. For example, while some queries in verticals may require diversity of results, others do not. Therefore, we will consider different approaches depending on types of queries and verticals. Furthermore, more unique features of each vertical could be considered as part of defining new affinity relationships.

## Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by NHN Corp. and in part by NSF grant #IIS-0711348. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

## References

1. Acton, F.S.: Numerical Methods that Work, 2nd edn. The Mathematical Association of America (1997)
2. Agichtein, E., Brill, E., Dumais, S.: Improving web search ranking by incorporating user behavior information. In: SIGIR 2006, pp. 19–26 (2006)
3. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: NIPS, vol. 14, pp. 585–591 (2001)
4. Bengio, Y., Delalleau, O., Le Roux, N.: Label propagation and quadratic criterion. In: Chapelle, O., Schölkopf, B., Zien, A. (eds.) Semi-Supervised Learning, pp. 193–216. MIT Press, Cambridge (2006)
5. Chapelle, O., Zhang, Y.: A dynamic bayesian network click model for web search ranking. In: WWW 2009, pp. 1–10 (2009)
6. Chung, F.R.K.: Spectral Graph Theory. American Mathematical Society, Providence (1997)
7. Croft, W.B., Lafferty, J.: Language Modeling for Information Retrieval. Kluwer Academic Publishers, Norwell (2003)
8. Diaz, F.: Regularizing ad hoc retrieval scores. In: CIKM 2005, pp. 672–679 (2005)
9. Dupret, G.E., Piwowarski, B.: A user browsing model to predict search engine click data from past observations. In: SIGIR 2008, pp. 331–338 (2008)
10. Gao, J., Yuan, W., Li, X., Deng, K., Nie, J.Y.: Smoothing clickthrough data for web search ranking. In: SIGIR 2009, pp. 355–362 (2009)
11. Jeon, J., Croft, W.B., Lee, J.H., Park, S.: A framework to predict the quality of answers with non-textual features. In: SIGIR 2006, pp. 228–235 (2006)
12. Joachims, T.: Optimizing search engines using clickthrough data. In: KDD 2002, pp. 133–142 (2002)
13. Joachims, T.: Transductive learning via spectral graph partitioning. In: ICML 2003, pp. 290–297 (2003)
14. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Gay, G.: Accurately interpreting clickthrough data as implicit feedback. In: SIGIR 2005, pp. 154–161 (2005)
15. Lafferty, J., Lebanon, G.: Diffusion kernels on statistical manifolds. The Journal of Machine Learning Research 6, 129–163 (2005)
16. Li, X., Wang, Y.Y., Acero, A.: Learning query intent from regularized click graphs. In: SIGIR 2008, pp. 339–346 (2008)
17. Murdock, V., Lalmas, M.: Workshop on aggregated search. SIGIR Forum 42(2), 80–83 (2008)
18. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the web. Tech. Rep. 1999-66, Stanford InfoLab (1999)
19. Radlinski, F., Joachims, T.: Active exploration for learning rankings from clickthrough data. In: KDD 2007, pp. 570–579 (2007)
20. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: Proceedings of the International Conference on Intelligence Analysis (2005)
21. Szummer, M., Jaakkola, T.: Partially labeled classification with markov random walks. In: Dietterich, T., et al. (eds.) Advances in Neural Information Processing Systems, vol. 14. MIT Press, Cambridge (2001)
22. Van Rijsbergen, C.J.: Information Retrieval, 2nd edn. Dept. of Computer Science, University of Glasgow (1979)
23. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: SIGIR 2001, pp. 334–342 (2001)
24. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Tech. Rep. CMU-CALD-02-107, Carnegie Mellon University (2002)



# Automatic People Tagging for Expertise Profiling in the Enterprise

Pavel Serdyukov<sup>1,\*</sup>, Mike Taylor<sup>2</sup>, Vishwa Vinay<sup>2</sup>,  
Matthew Richardson<sup>3</sup>, and Ryen W. White<sup>3</sup>

<sup>1</sup> Yandex, Moscow, Russia  
pserdukov@yandex.ru

<sup>2</sup> Microsoft Research, Cambridge, UK

<sup>3</sup> Microsoft Research, Redmond, USA

{mitaylor,vvinay,mattri,ryenw}@microsoft.com

**Abstract.** In an enterprise search setting, there is a class of queries for which people, rather than documents, are desirable answers. However, presenting users with just a list of names of knowledgeable employees without any description of their expertise may lead to confusion, lack of trust in search results, and abandonment of the search engine. At the same time, building a concise meaningful description for a person is not a trivial summarization task. In this paper, we propose a solution to this problem by automatically tagging people for the purpose of profiling their expertise areas in the scope of the enterprise where they are employed. We address the novel task of automatic people tagging by using a machine learning algorithm that combines evidence that a certain tag is relevant to a certain employee acquired from different sources in the enterprise. We experiment with the data from a large distributed organization, which also allows us to study sources of expertise evidence that have been previously overlooked, such as personal click-through history. The evaluation of the proposed methods shows that our technique clearly outperforms state of the art approaches.

## 1 Introduction

Members of large organizations frequently search for other people rather than documents. The need for well-informed colleagues is often critical, but manual expert identification through browsing documents or via professional social connections becomes more challenging with every new-hired employee. Expert finding algorithms have been developed to address this problem and retrieve ranked lists of people (rather than documents) in response to a search query. The task of expert finding has recently drawn significant attention from the academic community. However, there is still little reported research on industrial application scenarios of expert finding, which present new challenges and opportunities for the task.

One such challenge is the need to summarize the output of an expert finding system by presenting a concise description of expertise for each returned

---

\* The work was done while this author was visiting Microsoft Research, Cambridge.

employee. Surprisingly, this task, referred to as *expertise profiling*, has been almost completely neglected by researchers, yet the information presented in such profiles is important to users in deciding which people results to select.

It is challenging to summarize the expertise of a person, since evidence about an employee’s knowledge could be spread over many disparate sources in the enterprise. It is also difficult to find pieces of text that would summarize even parts of personal expertise in just a few sentences. At the same time, *tagging* of resources traditionally allows for their concise and meaningful summarization. Automatic *tagging* (*tag suggestion*, or *tag recommendation*) methods have been recently proposed for photos [14] or Web pages [15], but are lacking for people.

In this paper, we propose the novel task of automatic people tagging for the purpose of expertise profiling. In addition our work makes the following contributions:

- We conduct our study in an operational setting within an organization of more than 100,000 employees, using personal data of more than a thousand of them.
- We approach the problem of tag suggestion with a machine learning algorithm that ranks tags by their probability of being a good descriptor of personal expertise for a given employee. We demonstrate that combining expertise evidence extracted from various sources in the enterprise greatly outperforms a state-of-the-art expert finding method adopted for the same purpose, as well as a static ranking of tags by popularity.
- We demonstrate (first, to our knowledge) the usefulness of the enterprise search system’s *personal* click-through data for expertise evidence mining.

The remainder of this paper is structured as follows. The related research on expert finding and tag suggestion is reviewed in the next section. In Section 3 we describe our approach for automatic tagging of employees and provide details about the dataset, including a description of all expertise evidence sources examined in this work. Section 4 demonstrates the evaluation of the proposed method. We discuss these results and our work in general in Section 5. Finally, Section 6 summarizes our findings and outlines directions for future research.

## 2 Related Work

Automatic tagging methods were recently proposed for scientific documents/Web pages [15] and photos [14,7]. This research has been based either on analysis of existing tags with the purpose of extending the given set using tag co-occurrence statistics, or on the “propagation” of tags from tagged to untagged resources using their link graph. Such an approach would be difficult to employ in an enterprise setting, where large volumes of tags and tagged resources are typically unavailable for mining. There is other research that considers tags as topics and treats the tagging task as topical classification using co-occurring tags or terms of tagged documents as features [4]. Such an approach is not only computationally expensive due to the number of classes, but often suffers from the lack

of training data for infrequent tags. To the best of our knowledge, there is no research on automatic tag suggestion for people as resources or on automatic tagging of resources whose textual context would be so diffuse as in the case of people. However, the idea of people tagging, as a form of social bookmarking, is not entirely new. Farrel et al. [3] conducted user studies after running a pilot people tagging system at IBM for over a year. They reported the overall satisfaction of users with the application and found that their tags were accurate descriptions of users' interests and expertise. However, they did not provide the users with a candidate set of tags potentially relevant for their expertise.

Expert finding is a well-researched problem with a variety of approaches including language-model based [1,12], data fusion [8] and graph based [13,6] techniques. The only study to date on using result click-through behavior to enhance expert finding was presented by Macdonald and White [9]. However, they only used clicks as priors for documents, considering employees related to more frequently-clicked documents with respect to a given query as more likely to be experts on its topic. In contrast to their work, we not only analyze all queries leading to the documents authored by the employee under study, but actually focus on analyzing the utility of personal click-through data generated by the employee, including their search queries and clicked documents. Some research has also highlighted the importance of combining expertise evidence of various kind found inside [10] or outside [11] the enterprise using either a linear combination of measured estimates or rank aggregation techniques.

Progress on *expertise profiling* has been slow. Balog and de Rijke [2] experimented with the dataset from the Text Retrieval Conference (TREC 2005) for expert finding task, but instead of ranking candidate experts in response to queries, they ranked queries with respect to candidates. They used only 50 queries, which were actually the names of working groups in the organization under study and (almost) all members of these groups were considered as relevant for the pilot run of the expert task at TREC 2005. In this way, they tried to basically route people to relevant working groups. In their follow-up work, they used a different collection with 1491 test queries [1]. However, their work is limited in that queries were not actually tags (so, were carefully selected by an editor and were not overlapping in meaning) and were ranked using only one feature - the same measure which was used to rank candidate experts for the expert finding task using the same test queries. Our method extends this work by handling more noisy data and employing a more realistic and systematic machine-learning based approach to the problem.

## 3 Ranking Tags for Expertise Profiling

### 3.1 Problem Definition

We consider a scenario where a representative sample of users in the enterprise has already described their expertise with tags, covering the overall organizational expertise to a reasonable extent. The goal of the proposed system is to



constructed profiles. In the absence of relevance judgments for each candidate tag, we considered those tags specified by users in their profiles as *relevant* (positive examples) and all other tags not used to describe their expertise as *non-relevant* (negative examples). Our goal was then to predict how likely a tag is to be relevant given an employee using features and expertise evidence sources described later in this section.

### 3.3 Extracting Features from Expertise Evidence Sources

Expert finding methods traditionally rely on aggregates of relevance measures calculated over all informational units related to the employee [13]. In most cases, for example, they regard the sum of relevance scores of documents mentioning the employee as a measure of personal expertise on the given topic. In this work, we decided not to rely only on one stream of textual context of an employee or only on one measure indicating the strength of relation between a person and a tag. Using different definitions of a tag’s relevance to an employee’s expertise and all accessible streams of evidence, we extracted the features described further in this section.

To align with previous research [12], we used a language modeling approach to information retrieval to obtain an estimate for the probability of relevance  $P(e, t)$  of the tag  $t$  in respect to the personal expertise of the employee  $e$  given an evidence stream  $S$  (e.g., a set of authored documents):

$$P(e, t) = \sum_{D \in S} P(e, t|D)P(D) \quad (1)$$

$$P(e, t|D) = P(e|D)P(t|D) = P(e|D) \prod_{w \in t} P(w|D) \quad (2)$$

where  $w$  is the word from the tag  $t$ ,  $P(D)$  is the document’s  $D$  prior probability of relevance, whose distribution is uniform,  $P(e|D)$  is the probability of relation between the person  $e$  and the document  $D$ , which we considered binary in our work, as was often done earlier [8,6]. The probability to generate the term  $w$  from the document’s language model [5]:

$$P(w|D) = (1 - \lambda_G) \frac{c(w, D)}{|D|} + \lambda_G P(w|G), \quad (3)$$

where  $c(w, D)$  is the count of the term  $w$  in the document  $D$ ,  $|D|$  is its length,  $\lambda_G$  is the probability that term  $w$  will be generated from the global language model  $P(w|G)$ , which is estimated over the entire set of existing documents for all employees. Following previous studies, we set the  $\lambda_G$  to 0.5 in our experiments. Alternatively, apart from the language model based estimate of tag relevance given a document (**LM**) we considered the simple (**Binary**) model, which assumes that the probability  $P(t|D) = 1.0$ , when the tag  $t$  appears in the document  $D$  as a *phrase* at least once, and  $P(t|D) = 0$  otherwise.

There is an important difference between scoring tags for expertise profiling and scoring employees for expert finding. It is clear that some tags will appear

frequently in sources related to employees, and will not be descriptive of their expertise. Such tags will tend to dominate the top of the ranking just because they are generally very frequent. At the same time, it is intuitively important not only to be “rich in a tag”, to be an expert on the topic that it covers, but also to be richer than an average employee. In addition to the features described above, we also calculated their deviations from the averages measured on the set of training profiles:

$$P(e, t)^{dev} = P(e, t) - \frac{1}{|train|} \sum_{e' \in train} P(e', t) \quad (4)$$

Note that such transformation would not affect the rank ordering for expert finding, where employees are ranked given a tag (query), so the subtraction of any tag-specific constant for each employee’s score does not affect their rank. However, it does change the ranking for tags, since each tag has a different average score in the training set. As we show in our experiments, such transformation is also beneficial for performance.

As a result, given an employee and a tag, we calculated two scores, based on probabilistic (**LM**) and on binary model of relevance (**Binary**), for each set of informational units in each stream related to the employee and used these *scores* with their *deviations* as individual features for tag-employee pairs.

### 3.4 Additional Features

There are also features that are obtainable for a tag even with no information about a particular employee. We experimented with such features, including profile frequency of a tag in the training set, inverted document frequency, and the tag length in words or characters. According to our preliminary evaluation on the validation set (see Section 3.2), only the frequency of a tag in expertise profiles was predictive of the tag’s relevance, which we further used in our experiments as a feature.

### 3.5 Expertise Evidence Sources

We used a variety of information streams related to an employee to calculate the above mentioned features:

- **Authored and Related enterprise documents.** We crawled all documents authored by each person and also related documents which contained the employee’s full name and email address. This approach for discovering relationships among employees and documents is well known in expert finding research and we follow the traditional path here [8,13]. Authored documents are found by examining their metadata. As a result, we had 226 authored and 76 related documents per person on average, including articles, presentations, spreadsheets, etc. We considered each document field as an individual source of expertise evidence: **Title**, **File Name**, **Summary**, **Content**<sup>1</sup>.

<sup>1</sup> Bolded and capitalized names are used later to refer to the specific type of evidence.

- **Web documents.** Recent studies have shown that the evidence located outside the enterprise might be even more valuable than that found by analyzing internal sources [11]. We used the API of a major Web search engine to search for full names and email addresses of employees on the Web. Unfortunately, we could find only on average four related Web documents per person. The same above-mentioned document fields served as evidence sources.
- **Discussion lists.** Each employee in the enterprise under study is able to ask for assistance by sending emails to one of a set of existing discussion lists within the organization, which are monitored by other employees interested or knowledgeable on the topic of the list. While we did not have access to the content of questions and answers (while they are disseminated via discussion lists, their largest part stays strictly personal and the rest is not stored permanently), we had access to the employees' subscriptions - 172 per person on average - and used the *Name* of the related discussion list as a source of evidence.
- **Enterprise search click-through.** We used six months of search logs from January 2010 through June 2010 inclusive, obtained from thousands of users querying the Intranet of the organization under study. Click-through expertise evidence was extracted from three sources for each employee: 1) All *Personal Queries* issued by the employee: 67 unique queries per person on average; 2) Above-mentioned fields (title, file name, summary, content) of documents *Clicked* for these queries: 47 unique documents on average per person; 3) *Queries* of any employees that led to clicks on the documents authored by the employee: 12 unique queries on average per person.

## 4 Experiments

### 4.1 Baselines and Evaluation Measures

We consider two baselines in our work, also used as features to train our learning algorithm.

- **Application-specific baseline.** In the case, when there is no information about employees (for example, about those just entering the company), the only source of evidence about tag relevance is its prior probability in the training set. That is why we regard the *ProfileFrequency* of a tag to be one of our baselines.
- **State-of-the-art baseline.** Another baseline is taken from the only existing work on expertise profiling and represents a sum of scores of related documents with respect to the tag [2]. During preliminary experimentation with this baseline, we observed that it became relatively competitive only when using authored documents. Hereafter, we refer to this baseline and feature as *AuthoredContentLM*.

Since we regard our task as a problem of ranking tags, we evaluate our performance using the set of standard IR measures:

- Precisions at 1, 5 and 10 ranked candidate tags (P@1, P@5 and P@10),
- Success at 5 (S@5), showing the ability of the system to predict at least one relevant tag among the top ranked 5,
- Average Precision (AP).

We focus mainly on precision and success measures, since we consider them to be correlated with user satisfaction under the setup described in Section 3.1. In other words, since we assume that the purpose of the tags is to help users know more about each other, errors in the top ranks may lead to a greater effect on the impression of an employee’s personal expertise.

## 4.2 Learning for Tag Ranking

We experimented with a number of state-of-the-art classification algorithms on our validation set (see Section 3.2) and finally decided to use the output of the logistic regression function to rank tags, based on its performance and stability. Each pair of a tag and the profile where the tag appears served as a positive training example and all other non-matching pairs served as negative examples. In total, we used 4,098 positive and 60,000 negative examples. We sampled our negative examples from the entire set containing 900,000 negatives to avoid severe imbalance and the size of the sample was tuned on the validation set as well. Each combination of one of two expertise measures and their deviations (see Section 3.3) and one of the expertise evidence streams (see Section 3.5) resulted in a feature. In total, we trained using 78 features. Later in this section, we analyze the influence of features and streams on the performance of our learning model. Please note that we refer to specific features using bolded names of measures, evidence sources and specific document fields mentioned in Sections 3.3 and 3.5. For example, feature *RelatedSummaryLM* is obtained by summing the *LM* based probabilities of relevance of a tag in respect to *Summary* fields of *Related* enterprise documents.

However, first, we demonstrate the individual performance of the top most predictive features from each stream, including baselines, to give an idea about which of them appeared to be the most useful for the learning algorithm (Table 1, left part). We also show the performance of deviations from the average for these features (see Section 3.3, Equation 4), which prove that such a feature transformation is useful in almost all cases (Table 1, right part). As we see from Table 1, neither of the two baselines is able to outperform the strongest features extracted from distribution lists, as well as filenames and titles of authored documents. Features extracted from the set of personal queries appear to be the most useful among click-through features. They outperform the state-of-the-art performance baseline feature, but still perform worse than the application-specific baseline. Features of the streams that failed to provide competitive evidence for expertise mining are not included in the table due to space constraints.

As we also see in Table 1, our learning algorithm using *ALL* features is able to learn a ranking function which greatly outperforms both baselines. It also greatly outperforms the strongest feature *ListNamesBinary* (see Table 1), improving P@5 by 40%, AP by 42% and S@5 by 24%. It demonstrates the importance of



**Table 1.** Performance of feature groups

Stream	Feature performance			Deviation performance		
	P@5	AP	S@5	P@5	AP	S@5
<i>ProfileFrequency</i> (baseline)	0.066	0.046	0.253	-	-	-
<i>AuthoredContentLM</i> (baseline)	0.044	0.030	0.180	0.081	0.057	0.310
<b><i>ListNamesBinary</i></b>	<b>0.122</b>	<b>0.086</b>	<b>0.437</b>	<b>0.125</b>	<b>0.087</b>	<b>0.437</b>
<i>AuthoredFileNamesBinary</i>	0.071	0.058	0.3	0.093	0.066	0.367
<i>AuthoredTitlesLM</i>	0.072	0.053	0.283	0.085	0.059	0.313
<i>PersonalQueriesBinary</i>	0.055	0.040	0.210	0.059	0.041	0.220
<i>QueriesToAuthoredBinary</i>	0.059	0.038	0.230	0.069	0.043	0.307
<i>RelatedSummaryLM</i>	0.035	0.024	0.163	0.059	0.041	0.257
<i>ClickedTitlesBinary</i>	0.021	0.018	0.087	0.033	0.025	0.133
<i>WebTitlesLM</i>	0.023	0.012	0.093	0.023	0.013	0.097
<b>ALL features</b>	<b>0.171</b>	<b>0.124</b>	<b>0.543</b>	-	-	-

**Table 2.** Performance of feature groups

Stream	P@1	P@5	P@10	AP	S@5
<b>ALL</b>	<b>0.266</b>	<b>0.171</b>	<b>0.122</b>	<b>0.124</b>	<b>0.543</b>
- <i>ProfileFrequency</i>	0.240	0.138	0.102	0.110	0.460
- <i>List</i>	0.146	0.096	0.073	0.074	0.370
- <i>Authored</i>	0.130	0.078	0.065	0.063	0.300
- <i>PersonalQueries</i>	0.090	0.057	0.046	0.047	0.250
- <i>Related</i>	0.060	0.053	0.044	0.030	0.220
- <i>Clicked</i>	0.033	0.046	0.039	0.025	0.180
- <i>Web</i>	0.010	0.034	0.032	0.019	0.143
- <i>QueriesToAuthored</i>	0	0.025	0.023	0.007	0.123

combining all sources of evidence in the enterprise into one inference mechanism for the high-quality expertise mining and tag suggestion.

We also studied the contribution of features from each stream via feature ablation, removing the next most useful stream (according to P@5) at each step (until only one stream is left) and observing the change in performance. Rows with (-) in Table 2 demonstrate the order in which the groups of features were removed. As appeared, profile frequency was the most important feature in this regard, since it caused the most severe drop in performance, followed by distribution lists, authored documents and personal queries. Since features can interact in the learning model, we also experimented with removing each feature group at a time while leaving the other ones and observing the drops in performance, but it resulted in the same relative importance of streams.

### 4.3 Importance of Click-Through

One of the goals of this work was to study the importance of the expertise evidence mined from queries and clicks. Although Tables 1 and 2 show that some of the click-through streams are useful and certainly not inferior to many other

streams, it was still important to understand how the system would perform entirely without the click-through data, including personal queries, fields of clicked documents and queries to the documents authored by the employee. As we see from Table 3, the performance indeed drops when we remove all click-through features, for almost all measures by around 6-9%. It confirms the intuition that personal clickthrough history is suitable for mining personal expertise and its contribution is valuable for our inference mechanism.

However, as we noticed from the performance analysis of individual features and streams, some of them are very strong, while actually being very specific to the enterprise under study. We can imagine an enterprise with no such thing as an initial set of user profiles (considering that the controlled vocabulary was extracted from a different source, e.g., a query log) and distribution lists. For example, no enterprises studied previously in related academic research (see Section 2) actually considered these sources of expertise evidence. Such a “typical” enterprise would however be able to gather additional click-through evidence by monitoring users of its Intranet search system. We simulate the performance of our algorithm in such an enterprise by removing lists and profile frequency features. As we see in Table 3, we observe a severe drop in performance when we also exclude click-through features at the next step after that: over P@1 by 37%, over P@5 by 19%, over P@10 by 13%, over AP by 32%, over S@5 by 16%.

**Table 3.** Performance with/without click-through features

<b>Stream</b>	<b>P@1</b>	<b>P@5</b>	<b>P@10</b>	<b>AP</b>	<b>S@5</b>
<i>ALL</i>	0.266	0.171	0.122	0.124	0.543
<i>ALL - Click-through</i>	0.266	0.160	0.112	0.117	0.513
Typical enterprise: <i>ALL - Lists - ProfileFrequency</i>	0.146	0.096	0.073	0.074	0.37
Typical enterprise - <i>Click-through</i>	0.093	0.078	0.056	0.050	0.310
Click-through (only)	0.09	0.061	0.047	0.048	0.247

## 5 Discussion

The findings presented in the previous section show that it is possible to effectively provide members of an organization with a relevant ranking of tags describing their expertise. Building such a system has multiple advantages, the main one of which is that we now have information for people who have not gone through the effort of tagging themselves. Encouraging people to use such a system will have benefits that go beyond the profiling, i.e., related task like expertise search will improve as well. It was also clear that none of the text streams of expertise evidence is sufficient alone to attain the maximum performance of the tag ranking. For example, features extracted from personal click-through, while not being the most predictive on their own, appeared to be highly valuable as a part of the complete feature set.

Some of our findings contradicted previous research. We were particularly surprised by the poor performance of the features extracted from Web documents,

considering that Serdyukov and Hiemstra [11] found that Web documents provided the most valuable source of expertise evidence when they experimented with expert finding in a large research institution. It seems that it is possible to obtain a sufficient amount of Web-based expertise evidence only for organizations motivating their employees for regular public activities (e.g., research labs advocating the publication of research results), and this is not often the case for most commercial organizations.

We also analyzed the tags that our algorithm failed to predict over all test employees. In most cases, these were tags whose relevance was hard to evaluate using enterprise data. There were generally two classes of these tags: 1) too personal tags (e.g., “ice cream”, “cooking”, “dancing”, “judaism”), and 2) abstract tags (e.g., “customer satisfaction”, “public speaking”, “best practices”). While, in the first case, such personal tags simply could not be found in work-related evidence sources, in the second case, abstract tags were too general to be often used in the text or even in personal queries.

Another source of “errors” in prediction was the consequence of our definition of relevance used due to the origin of our evaluation set. We tried to predict the exact tag used by the employee to describe own expertise, so only these tags were regarded as relevant. However, this is a much more difficult task than the task of prediction of any relevant tags. In our case, for example, “machine learning” and “data mining”, or “networking” and “networks”, were not necessarily both relevant for the same employee, while it is unusual to possess expertise on only one of these subjects. The success measure used in our work (S@5) is more robust to this issue, since it gives full credit for those test employees for whom the model predicted at least one of correct tags in their profile.

## 6 Conclusions and Further Work

In this paper we have proposed a technique for automatic tagging of employees in the enterprise for the purpose of expertise profiling. Assuming that an initial set of employees have described themselves with tags, we investigated the utility of different sources of evidence of a person’s expertise as predictors of tags they are likely to use. We were able to train a classifier that produced candidate expertise terms for an as-yet untagged person, thereby providing an automatic profiling mechanism. We have thoroughly tested this technique using data from a large organization and a variety of expertise evidence sources, including those that have not been studied before in the scope of the tasks focused on expertise location, such as personal click-through history.

Our experiences suggest that when asked to describe their own expertise, the words chosen by employees of an enterprise can only partially be inferred from the enterprise content that can be associated with them. Modeling of the remaining *knowledge* of a person remains a challenge. Our experiments also indicated that the problem of ranking of tags for people involve considerations that are familiar to other retrieval tasks, such as the need to diversify the ranked list. Besides, it is important to investigate not only how to build personal profiles, but also how

to make these summaries query-dependent to dynamically appropriate them as result snippets for a people search engine. Addressing such concerns will be the subject of our future research.

**Acknowledgments.** We would like to thank Milad Shokouhi, Emine Yilmaz and Gjergji Kasneci for lots of helpful comments and inspiring discussions.

## References

1. Balog, K., Bogers, T., Azzopardi, L., de Rijke, M., van den Bosch, A.: Broad expertise retrieval in sparse data environments. In: *SIGIR 2007*, pp. 551–558. ACM, New York (2007)
2. Balog, K., de Rijke, M.: Determining expert profiles (with an application to expert finding). In: *IJCAI 2007*, pp. 2657–2662. Morgan Kaufmann Publishers, San Francisco (2007)
3. Farrell, S., Lau, T., Nusser, S., Wilcox, E., Muller, M.: Socially augmenting employee profiles with people-tagging. In: *UIST 2007: Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, pp. 91–100. ACM, New York (2007)
4. Heymann, P., Ramage, D., Garcia-Molina, H.: Social tag prediction. In: *SIGIR 2008*, pp. 531–538. ACM, New York (2008)
5. Hiemstra, D., de Jong, F.M.G.: Statistical language models and information retrieval: Natural language processing really meets retrieval. *Glott International* 5(8), 288–293 (2001)
6. Karimzadehgan, M., White, R.W., Richardson, M.: Enhancing expert finding using organizational hierarchies. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) *ECIR 2009*. LNCS, vol. 5478, pp. 177–188. Springer, Heidelberg (2009)
7. Liu, D., Hua, X.-S., Yang, L., Wang, M., Zhang, H.-J.: Tag ranking. In: *WWW 2009*, pp. 351–360. ACM, New York (2009)
8. Macdonald, C., Ounis, I.: Voting for candidates: adapting data fusion techniques for an expert search task. In: *CIKM 2006*, pp. 387–396. ACM, New York (2006)
9. Macdonald, C., White, R.W.: Usefulness of click-through data in expert search. In: *SIGIR 2009*, pp. 816–817. ACM, New York (2009)
10. Petkova, D., Croft, W.B.: Hierarchical language models for expert finding in enterprise corpora. In: *ICTAI 2006: Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, pp. 599–608. IEEE, Los Alamitos (2006)
11. Serdyukov, P., Hiemstra, D.: Being Omnipresent to Be Almighty: The Importance of the Global Web Evidence for Organizational Expert Finding. In: *FCHER 2008: Proceedings of the SIGIR 2008 Workshop on Future Challenges in Expertise Retrieval* (2008)
12. Serdyukov, P., Hiemstra, D.: Modeling documents as mixtures of persons for expert finding. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008*. LNCS, vol. 4956, pp. 309–320. Springer, Heidelberg (2008)
13. Serdyukov, P., Rode, H., Hiemstra, D.: Modeling multi-step relevance propagation for expert finding. In: *CIKM 2008*, pp. 1133–1142. ACM, New York (2008)
14. Sigurbjörnsson, B., van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: *WWW 2008*, pp. 327–336. ACM, New York (2008)
15. Song, Y., Zhuang, Z., Li, H., Zhao, Q., Li, J., Lee, W.-C., Giles, C.L.: Real-time automatic tag recommendation. In: *SIGIR 2008*, pp. 515–522. ACM, New York (2008)

# Text Classification: A Sequential Reading Approach

Gabriel Dulac-Arnold, Ludovic Denoyer, and Patrick Gallinari

University Pierre et Marie Curie - UPMC, LIP6  
Case 169, 4 Place Jussieu - 75005 Paris, France  
`firstname.lastname@lip6.fr`

**Abstract.** We propose to model the text classification process as a sequential decision process. In this process, an agent learns to classify documents into topics while reading the document sentences sequentially and learns to stop as soon as enough information was read for deciding. The proposed algorithm is based on a modelisation of Text Classification as a Markov Decision Process and learns by using Reinforcement Learning. Experiments on four different classical mono-label corpora show that the proposed approach performs comparably to classical SVM approaches for large training sets, and better for small training sets. In addition, the model automatically adapts its reading process to the quantity of training information provided.

## 1 Introduction

Text Classification (TC) is the act of taking a set of labeled text documents, learning a correlation between a document's contents and its corresponding labels, and then predicting the labels of a set of unlabeled test documents as best as possible. TC has been studied extensively, and is one of the older specialties of Information Retrieval. Classical statistical TC approaches are based on well-known machine learning models such as generative models — Naive Bayes for example [1][2] — or discriminant models such as Support Vector Machines [3]. They mainly consider the *bag of words* representation of a document (where the order of the words or sentences is lost) and try to compute a category score by looking at the entire document content. *Linear* SVMs in particular — especially for multi-label classification with many binary SVMs — have been shown to work particularly well [4]. Some major drawbacks to these *global* methods have been identified in the literature:

- These methods take into consideration a document's entire word set in order to decide to which categories it belongs. The underlying assumption is that the category information is homogeneously dispatched inside the document. This is well suited for corpora where documents are short, with little noise, so that global word frequencies can easily be correlated to topics. However, these methods will not be well suited in predicting the categories of large documents where the topic information is concentrated in only a few sentences.

- Additionally, for these methods to be applicable, the entire document must be known at the time of classification. In cases where there is a cost associated with acquiring the textual information, methods that consider the entire document cannot be efficiently or reliably applied as we do not know at what point their classification decision is well-informed while considering only a subset of the document.

Considering these drawbacks, some attempts have been made to use the sequential nature of these documents for TC and similar problems such as passage classification. The earliest models developed especially for sequence processing extend Naive Bayes with Hidden Markov Models. Denoyer et al. [5] propose an original model which aims at modeling a document as a sequence of irrelevant and relevant sections relative to a particular topic. In [6], the authors propose a model based on recurrent Neural Networks for document routing. Other approaches have proposed to extend the use of linear SVMs to sequential data, mainly through the use of string kernels [7]. Finally, sequential models have been used for Information Extraction [8,9], passage classification [10,11], or the development of search engines [12,13].

We propose a new model for Text Classification that is less affected by the aforementioned issues. Our approach models an agent that sequentially reads a text document while concurrently deciding to assign topic labels. This is modeled as a sequential process whose goal is to classify a document by focusing on its relevant sentences. The proposed model learns not only to classify a document into one or many classes, but also *when* to label, and when to stop reading the document. This last point is very important because it means that the systems is able to learn to label a document with the correct categories as soon as possible, without reading the entire text.

The contributions of this paper are three-fold:

1. We propose a new type of sequential model for text classification based on the idea of sequentially reading sentences and assigning topics to a document.
2. Additionally, we propose an algorithm using Reinforcement Learning that learns to focus on relevant sentences in the document. This algorithm also learns when to stop reading a document so that the document is classified as soon as possible. This characteristic can be useful for documents where sentence acquisition is expensive, such as large Web documents or conversational documents.
3. We show that on popular text classification corpora our model outperforms classical TC methods for small training sets and is equivalent to a baseline SVM for larger training sets while only reading a small portion of the documents. The model also shows its ability to classify by reading only a few sentences when the classification problem is easy (large training sets) and to learn to read more sentences when the task is harder (small training sets).

This document is organized as follows: In Section 2, we present an overview of our method. We formalize the algorithm as a Markov Decision Process in Section 3 and detail the approach for both multi-label and mono-label TC.

We then present the set of experiments made on four different text corpora in Section 4.

## 2 Task Definition and General Principles of the Approach

Let  $\mathcal{D}$  denote the set of all possible textual documents, and  $\mathcal{Y}$  the set of  $C$  categories numbered from 1 to  $C$ . Each document  $d$  in  $\mathcal{D}$  is associated with one or many categories of  $\mathcal{C}$ . This label information is only known for a subset of documents  $\mathcal{D}_{train} \subset \mathcal{D}$  called training documents, composed of  $N_{train}$  documents denoted  $\mathcal{D}_{train} = (d_1, \dots, d_{N_{train}})$ . The labels of document  $d_i$  are given by a vector of scores  $y^i = (y_1^i, \dots, y_C^i)$ . We assume that:

$$y_k^i = \begin{cases} 1 & \text{if } d_i \text{ belongs to category } k \\ 0 & \text{otherwise} \end{cases} . \quad (1)$$

The goal of TC is to compute, for each document  $d$  in  $\mathcal{D}$ , the corresponding score for each category. The classification function  $f_\theta$  with parameters  $\theta$  is thus defined as :

$$f_\theta : \begin{cases} \mathcal{D} : [0; 1]^C \\ d \rightarrow y^d \end{cases} . \quad (2)$$

Learning the classifier consists in finding an optimal parameterization  $\theta^*$  that reduces the mean loss such that:

$$\theta^* = \operatorname{argmin}_\theta \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} L(f_\theta(d_i), y^{d_i}), \quad (3)$$

where  $L$  is a loss function proportional to the classification error of  $f_\theta(d_i)$ .

### 2.1 Overview of the Approach

This section aims to provide an intuitive overview of our approach. The ideas presented here are formally presented in Section 3, and will only be described in a cursory manner below.

**Inference.** We propose to model the process of text classification as a sequential decision process. In this process, our classifier reads a document sentence-by-sentence and can decide — at each step of the reading process — if the document belongs to one of the possible categories. This classifier can also chose to stop reading the document once it considers that the document has been correctly categorized.

In the example described in Fig. 1, the task is to classify a document composed of 4 sentences. The documents starts off unclassified, and the classifier begins

<sup>1</sup> In this article, we consider both the mono-label classification task, where each document is associated with exactly one category, and the multi-label task where a document can be associated with several categories.

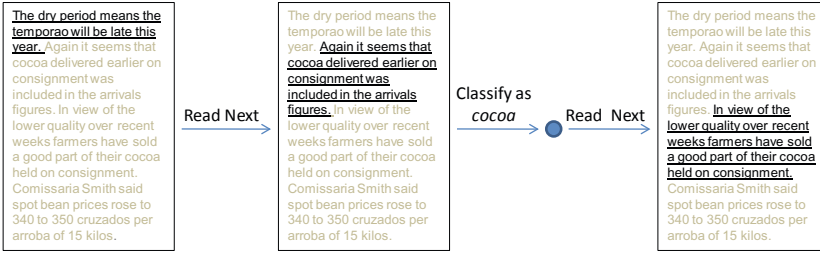


Fig. 1. Inference on a document

by reading the first sentence of the document. Because it considers that the first sentence does not contain enough information to reliably classify the document, the classifier decides to read the following sentence. Having now read the first two sentences, the classifier decides that it has enough information at hand to classify the document as *cocoa*.

The classifier now reads the third sentence and — considering the information present in this sentence — decides that the reading process is finished; the document is therefore classified in the *cocoa* category.

Had the document belonged to multiple classes, the classifier could have continued to assign other categories to the document as additional information was discovered.

In this example, the model took four **actions**: *next*, *classify as cocoa*, *next* and then *stop*. The choice of each action was entirely dependent on the corresponding **state** of the reading process. The choice of actions given the state, such as those picked while classifying the example document above, is called the **policy** of the classifier. This policy — denoted  $\pi$  — consists of a mapping of states to actions relative to a score. This score is called a Q-value — denoted  $Q(s, a)$  — and reflects the worth of choosing action  $a$  during state  $s$  of the process. Using the Q-value, the inference process can be seen as a **greedy process** which, for each timestep, chooses the best action  $a^*$  defined as the action with the highest score w.r.t.  $Q(s, a)$ :

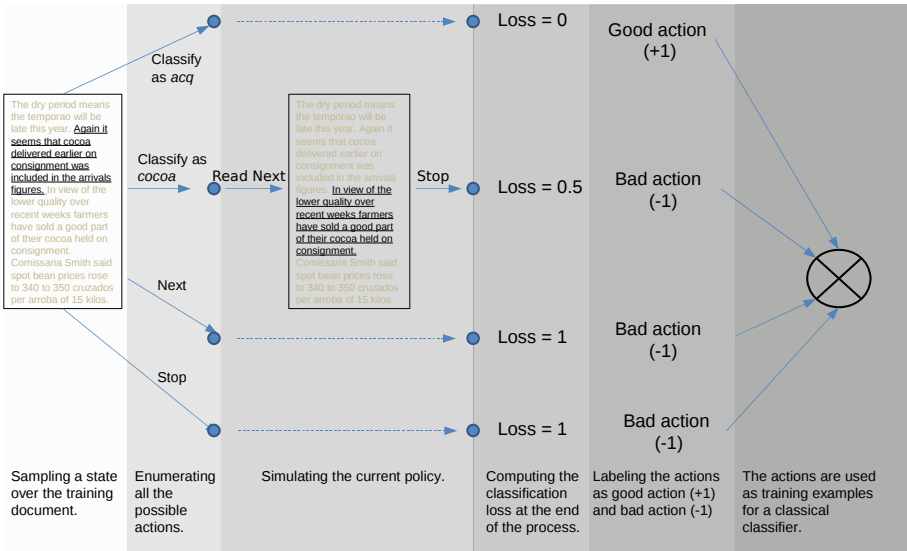
$$a^* = \operatorname{argmax}_a Q(s, a). \quad (4)$$

**Training.** The learning process consists in computing a *Q-function*<sup>2</sup> which minimizes the classification loss (as in equation (3)) of the documents in the training set. The learning procedure uses a monte-carlo approach to find a set of *good* and *bad* actions relative to each state. *Good* actions are actions that result in a small classification loss for a document. The *good* and *bad* actions are then learned by a statistical classifier, such as an SVM.

An example of the training procedure on the same example document as above is illustrated in Fig 2. To begin with, a random state of the classification process is picked. Then, for each action possible in that state, the current policy is run

<sup>2</sup> The *Q-function* is an approximation of  $Q(s, a)$ .





**Fig. 2.** Learning the sequential model. The different steps of one learning iteration are illustrated from left to right on a single training document.

until it stops and the final classification loss is computed. The training algorithm then builds a set of *good* actions — the actions for which the simulation obtains the minimum loss value — and a set of remaining *bad* actions. This is repeated on many different states and training documents until, at last, the model learns a classifier able to discriminate between *good* and *bad* actions relative to the current state.

## 2.2 Preliminaries

We have presented the principles of our approach and given an intuitive description of the inference and learning procedures. We will now formalize this algorithm as a Markov Decision Process (MDP) for which an optimal policy is found via Reinforcement Learning. Note that we will only go over notations pertinent to our approach, and that this section lacks many MDP or Reinforcement Learning definitions that are not necessary for our explanation.

**Markov Decision Process.** A Markov Decision Process (MDP) is a mathematical formalism to model sequential decision processes. We only consider deterministic MDPs, defined by a 4-tuple:  $(\mathcal{S}, \mathcal{A}, T, r)$ . Here,  $\mathcal{S}$  is the set of possible states,  $\mathcal{A}$  is the set of possible actions, and  $T: \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$  is the state transition function such that  $T(s, a) \rightarrow s'$  (this symbolizes the system moving from state  $s$  to state  $s'$  by applying action  $a$ ). The reward,  $r: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ , is a value that reflects the quality of taking action  $a$  in state  $s$  relative to the agent's ultimate goal. We will use  $\mathcal{A}(s) \subseteq \mathcal{A}$  to refer to the set of possible actions available to an agent in a particular state  $s$ .

An agent interacts with the MDP by starting off in a state  $s \in \mathcal{S}$ . The agent then chooses an action  $a \in \mathcal{A}(s)$  which moves it to a new state  $s'$  by applying the transition  $T(s, a)$ . The agent obtains a reward  $r(s, a)$  and then continues the process until it reaches a terminal state  $s_{final}$  where the set of possible actions is empty i.e  $\mathcal{A}(s_{final}) = \emptyset$ .

**Reinforcement Learning.** Let us define  $\pi : \mathcal{S} \rightarrow \mathcal{A}$ , a stochastic policy such that  $\forall a \in \mathcal{A}(s)$ ,  $\pi(s) = a$  with probability  $P(a|s)$ . The goal of RL is to find an optimal policy  $\pi^*$  that maximizes the cumulative reward obtain by an agent. We consider here the finite-horizon context for which the cumulative reward corresponds to the sum of the reward obtained at each step by the system, following the policy  $\pi$ . The goal of Reinforcement Learning is to find an optimal policy denoted  $\pi^*$  which maximizes the cumulative reward obtained for all the states of the process i.e.:

$$\pi^* = \operatorname{argmax}_{\pi} \sum_{s_0 \in \mathcal{S}} \mathbb{E}_{\pi} \left[ \sum_{t=0}^T r(s_t, a_t) \right]. \quad (5)$$

Many algorithms have been developed for finding such a policy, depending on the assumptions made on the structure of the MDP, the nature of the states (discrete or continuous), etc. In many approaches, a policy  $\pi$  is defined through the use of a *Q-function* which reflects how much reward one can expect by taking action  $a$  on state  $s$ . With such a function, the policy  $\pi$  is defined as:

$$\pi = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q(s, a). \quad (6)$$

In such a case, the learning problem consists in finding the optimal value  $Q^*$  which results in the optimal policy  $\pi^*$ .

Due to the very large number of states we are facing in our approach, we consider the Approximated Reinforcement Learning context where the  $Q$  function is approximated by a parameterized function  $Q_{\theta}(s, a)$ , where  $\theta$  is a set of parameters such that:

$$Q_{\theta}(s, a) = \langle \theta, \Phi(s, a) \rangle, \quad (7)$$

where  $\langle \cdot, \cdot \rangle$  denotes the dot product and  $\Phi(s, a)$  is a feature vector representing the state-action pair  $(s, a)$ . The learning problem consists in finding the optimal parameters  $\theta^*$  that results in an optimal policy:

$$\pi^* = \operatorname{argmax}_{a \in \mathcal{A}(s)} \langle \theta^*, \Phi(s, a) \rangle. \quad (8)$$

### 3 Text Classification as a Sequential Decision Problem

Formally, we consider that a document  $d$  is composed of a sequence of sentences such that  $d = (\delta_1^d, \dots, \delta_{n_d}^d)$ , where  $\delta_i^d$  is the  $i$ -th sentence of the document and  $n_d$  is the total number of sentences making up the document. Each sentence  $\delta_i^d$  has a corresponding feature vector — a normalized tf-idf vector in our case — that describes its content.

### 3.1 MDP for Multi-label Text Classification

We can describe our sequential decision problem using an MDP. Below, we describe the MDP for the multilabel classification problem, of which monolabel classification is just a specific instance:

- Each state  $s$  is a triplet  $(d, p, \hat{y})$  such that:
  - $d$  is the document the agent is currently reading.
  - $p \in [1, n_d]$  corresponds to the current sentence being read; this implies that  $\delta_1^d$  to  $\delta_{p-1}^d$  have already been read.
  - $\hat{y}$  is the set of currently assigned categories — categories previously assigned by the agent during the current reading process — where  $\hat{y}_k = 1$  iff the document has been assigned to category  $k$  during the reading process, 0 otherwise.
- The set of actions  $\mathcal{A}(s)$  is composed of:
  - One or many **classification** actions denoted *classify as k* for each category  $k$  where  $\hat{y}_k = 0$ . These actions correspond to assigning document  $d$  to category  $k$ .
  - A **next sentence** action denoted *next* which corresponds to reading the next sentence of the document.
  - A **stop action** denoted *stop* which corresponds to finishing the reading process.
- The set of transitions  $T(s, a)$  act such that:
  - $T(s, \text{classify as } k)$  sets  $\hat{y}_k \leftarrow 1$ .
  - $T(s, \text{next})$  sets  $p \leftarrow p + 1$ .
  - $T(s, \text{stop})$  halts the decision process.
- The reward  $r(s, a)$  is defined as:

$$r(s, a) = \begin{cases} F_1(y, \hat{y}) & \text{if } a \text{ is a } \textit{stop} \text{ action} \\ 0 & \text{otherwise} \end{cases}, \quad (9)$$

where  $y$  is the real vector of categories for  $d$  and  $\hat{y}$  is the predicted vector of categories at the end of the classification process. The  $F_1$  score of a single document is defined as:

$$F_1(y, \hat{y}) = 2 \cdot \frac{p(y, \hat{y}) \cdot r(y, \hat{y})}{p(y, \hat{y}) + r(y, \hat{y})} \quad (10)$$

$$\text{with} \quad (11)$$

$$p(y, \hat{y}) = \sum_{k=0}^C \mathbb{1}(\hat{y}_k = y_k) / \sum_{k=0}^C \hat{y}_k \quad \text{and} \quad r(y, \hat{y}) = \sum_{k=0}^C \mathbb{1}(\hat{y}_k = y_k) / \sum_{k=0}^C y_k. \quad (12)$$

**MDP for Mono-label Text Classification.** In mono-label classification, we restrict the set of possible actions. The *classify as k* action leads to a stopping state such that  $A(s) = \{\textit{stop}\}$ . This brings the episode to an end after the attribution of a single label. Note that in the case of a mono-label system — where only one category can be assigned to a document — the reward corresponds to a classical accuracy measure: 1 if the chosen category is correct, and 0 otherwise.

### 3.2 Features over States

We must now define a feature function which provides a vector representation of a state-action pair  $(s, a)$ . The purpose of this vector is to be able to present  $(s, a)$  to a statistical classifier to know whether it is *good* or *bad*. Comparing the scores of various  $(s, a)$  pairs for a given state  $s$  allows us to choose the best action for that state.

Classical text classification methods only represent documents by a global — and usually tf-idf weighted — vector. We choose, however, to include not only a global representation of the sentences read so far, but also a local component corresponding to the most recently read sentence. Moreover, while in state  $s$ , a document may have been already assigned to a set of categories; the global feature vector  $\Phi(s, a)$  must describe all this information. The vector representation of a state  $s$  is thus defined as  $\Phi(s)$ :

$$\Phi(s) = \left( \begin{array}{c} \sum_{i=1}^p \delta_i^d \\ p \end{array} \delta_p^d \hat{y}_0 \dots \hat{y}_C \right). \quad (13)$$

$\Phi(s)$  is the concatenation of a set of sub-vectors describing: the mean of the feature vectors of the read sentences, the feature vector of the last sentence, and the set of already assigned categories.

In order to include the action information, we use the block-vector trick introduced by [14] which consists in projecting  $\Phi(s)$  into a higher dimensional space such that:

$$\Phi(s, a) = (0 \dots \phi(s) \dots 0). \quad (14)$$

The position of  $\Phi(s)$  inside the global vector  $\Phi(s, a)$  is dependent on action  $a$ . This results in a very high dimensional space which is easier to classify in with a linear model.

### 3.3 Learning and Finding the Optimal Classification Policy

In order to find the best classification policy, we used a recent Reinforcement Learning algorithm called *Approximate Policy Iteration with Rollouts*. In brief, this method uses a monte-carlo approach to evaluate the quality of all the possible actions amongst some random sampled states, and then learns a classifier whose goal is to discriminate between the *good* and *bad* actions relative to each state. Due to a lack of space, we do not detail the learning procedure here and refer to the paper by Lagoudakis et al [15]. An intuitive description of the procedure is given in Section 2.1.

## 4 Experimental Results

We have applied our model on four different popular datasets: three mono-label and one multi-label. All datasets were pre-processed in the same manner: all

punctuation except for periods were removed, SMART stop-words [16] and words less than three characters long were removed, and all words were stemmed with Porter stemming. Baseline evaluations were performed with libSVM [17] on normalized tf-idf weighted vectorial representations of each document as has been done in [3]. Published performance benchmarks can be found in [18] and [19]. The datasets are:

- The Reuters-21578<sup>3</sup> dataset which provides two corpora:
  - The **Reuters8** corpus, a mono-label corpus composed of the 8 largest categories.
  - The **Reuters10** corpus, a multi-label corpus composed of the 10 largest categories.
- The WebKB<sup>4</sup>[20] dataset is a mono-label corpus composed of Web pages dispatched into 4 different categories.
- The 20 Newsgroups<sup>5</sup> (20NG) dataset is a mono-label corpus of news composed of 20 classes.

**Table 1.** Corpora statistics

Corpus	Nb of documents	Nb of categories	Nb of sentences by doc.	Task
R8	7678	8	8.19	Mono-label
R10	12 902	10	9.13	Multi-label
Newsgroup	18 846	20	22.34	Mono-label
WebKB	4 177	4	42.36	Mono-label

#### 4.1 Evaluation Protocol

Many classification systems are *soft classification* systems that compute a score for each possible category-document pair. Our system is a *hard classification* system that assigns a document to one or many categories, with a score of either 1 or 0. The evaluation measures used in the literature, such as the *breakeven* point, are not suitable for *hard classification* models and cannot be used to evaluate and compare our approach with other methods. We have therefore chosen to use the *micro-F1* and *macro-F1* measures. These measures correspond to a classical  $F_1$  score computed for each category and averaged over the categories. The *macro-F1* measure does not take into account the size of the categories, whereas the *micro-F1* average is weighted by the size of each category. We averaged the different models' performances on various train/test splits that were randomly generated from the original dataset. We used the same approach both for evaluating our approach and the baseline approaches to be able to compare our results properly. For each training size, the performance of the models were averaged over 5 runs. The hyper-parameters of the SVM and the

<sup>3</sup> <http://web.ist.utl.pt/%7Eacardoso/datasets/>

<sup>4</sup> <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/>

<sup>5</sup> <http://people.csail.mit.edu/jrennie/20Newsgroups/>

hyper-parameters of the RL-based approach were manually tuned. What we present here are the best results obtained over the various parameter choices we tested. For the RL approach, each policy was learned on 10,000 randomly generated states, with 1 rollout per state, using a random initial policy. It is important to note that, in a practical sense, the RL method is not much more complicated to tune than a classical SVM since it is rather robust regarding the values of the hyper-parameters.

### 4.2 Experimental Results

Our performance figures use SVM to denote baseline Support Vector Machine performance, and STC (Sequential Text Classification) to denote our approach. In the case of the mono-label experiments (Figure 3 and 4-left), performance of both the SVM method and our method are comparable. It is important to note, however, that in the case of small training sizes (1%, 5%), the STC approach outperforms SVM by 1-10% depending on the corpus. For example, on the R8 dataset we can see that for both *F1* scores, STC is better by  $\sim 5\%$  with a training size of 1%. This is also visible with the NewsGroup dataset, where STC is better by 10% for both metrics using a 1% training set. This shows that STC is particularly advantageous with small training sets.

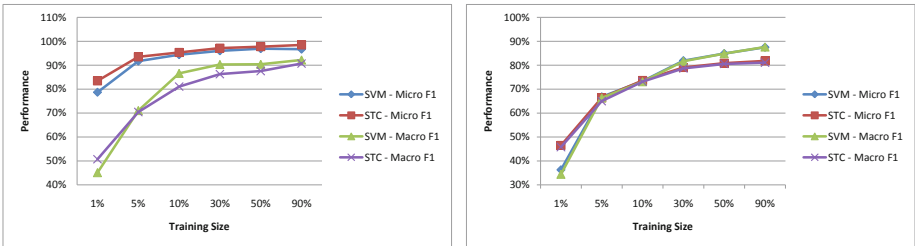


Fig. 3. Performances over the R8 Corpus (left) and NewsGroup Corpus (right)

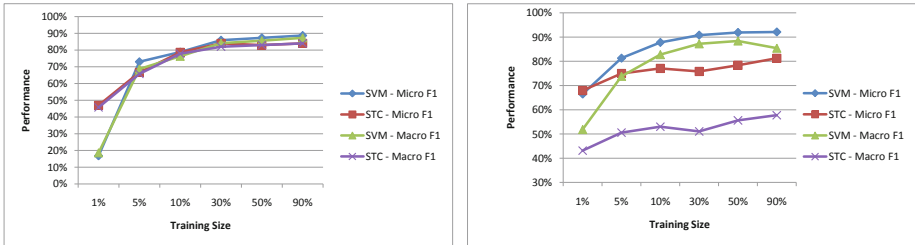
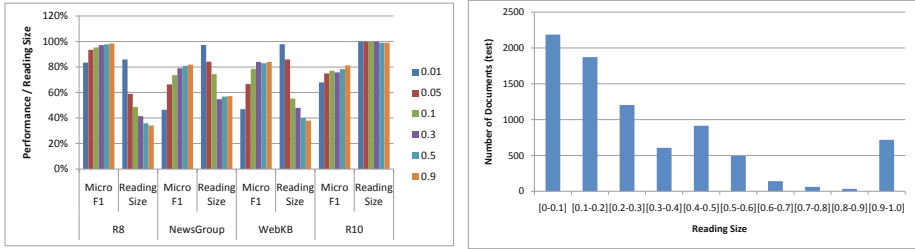


Fig. 4. Performances over the WebKB Corpus (left) and R10 Corpus (right)



**Fig. 5.** Overview of the Reading Sizes for all the corpora (left). Number of documents and Reading Sizes on R8 with 30% of documents as a training set (right).

The reading process’ behaviour is explored in Figure 5. Here, *Reading Size* corresponds to the mean percentage of sentences read for each document<sup>6</sup>. We can see that *Reading Size* decreases as the training size gets bigger for mono-label corpora. This is due to the fact that the smaller training sizes are harder to learn, and therefore the agent needs more information to properly label documents. In the right-hand side of Figure 5, we can see a histogram of number of documents grouped by *Reading Size*. We notice that although there is a mean *Reading Size* of 41%, most of the documents are barely read, with a few outliers that are read until the end. The agent is clearly capable of choosing to read more or less of the document depending on its content.

In the case of multi-label classification, results are quite different. First, we see that for the R10 corpus, our model’s performance is lower than the baseline on large training sets. Moreover, the multi-label model reads all the sentences of the document during the classification process. This behaviour seems normal because when dealing with multi-label documents, one cannot be sure that the remaining sentences will not contain relevant information pertaining to a particular topic. We hypothesize that the lower performances are due to the much larger action space in the multi-label problem, and the fact that we are learning a single model for all classes instead of one independent models per class.

## 5 Conclusions

We have presented a new model that learns to classify by sequentially reading the sentences of a document, and which labels this document as soon as it has collected enough information. This method shows some interesting properties on different datasets. Particularly in mono-label TC, the model automatically learns to read only a small part of the documents when the training set is large, and the whole documents when the training set is small. It is thus able to adapt its behaviour to the difficulty of the classification task, which results in obtaining

<sup>6</sup> If  $l_i$  is the number of sentences in document  $i$  read during the classification process, and  $n_i$  is the total number of sentences in this document. Let  $N$  be the number of test documents, then the reading size value is  $\frac{1}{N} \sum_i \frac{l_i}{n_i}$ .

faster systems for easier problems. The performances obtained are close to the performance of a baseline SVM model for large training sets, and better for small training sets.

This work opens many new perspectives in the Text Classification domain. Particularly, it is possible to imagine some additional MDP actions for the classification agent allowing the agent to parse the document in a more complex manner. For example, this idea can be extended to learn to classify XML documents reading only the relevant parts.

## Acknowledgments

This work was partially supported by the French National Agency of Research (Lampada ANR-09-EMER-007).

## References

1. Lewis, D., Ringuette, M.: A comparison of two learning algorithms for text categorization. In: Third Annual Symposium on Document Analysis, pp. 1–14 (1994)
2. Lewis, D., Schapire, R., Callan, J., Papka, R.: Training algorithms for linear text classifiers. In: ACM SIGIR, pp. 120–123 (1996)
3. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, Springer, Heidelberg (1998)
4. Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization. In: Proceedings of CIKM (1998)
5. Denoyer, L., Zaragoza, H., Gallinari, P.: HMM-based passage models for document classification and ranking. In: Proceedings of ECIR 2001, pp. 126–135 (2001)
6. Wermter, S., Arevian, G., Panchev, C.: Recurrent neural network learning for text routing, vol. 2, pp. 898–903 (1999)
7. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. *J. Mach. Learn. Res.* 2, 419–444 (2002)
8. Leek, T.R.: Information extraction using hidden markov models (1997)
9. Amini, M.-R., Zaragoza, H., Gallinari, P.: Learning for sequence extraction tasks. In: RIAO, pp. 476–490 (2000)
10. Kaszkiel, M., Zobel, J., Sacks-Davis, R.: Efficient passage ranking for document databases. *ACM Trans. Inf. Syst.* 17(4), 406–439 (1999)
11. Jiang, J., Zhai, C.: Extraction of coherent relevant passages using hidden markov models. *ACM Trans. Inf. Syst.* 24(3), 295–319 (2006)
12. Miller, D.R.H., Leek, T., Schwartz, R.M.: Bbn at trec7: Using hidden markov models for information retrieval. In: Proceedings of TREC-7, pp. 133–142 (1999)
13. Bendersky, M., Kurland, O.: Utilizing passage-based language models for document retrieval. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 162–174. Springer, Heidelberg (2008)
14. Har-Peled, S., Roth, D., Zimak, D.: Constraint classification: A new approach to multiclass classification. In: Cesa-Bianchi, N., Numao, M., Reischuk, R. (eds.) ALT 2002. LNCS (LNAI), vol. 2533, pp. 365–379. Springer, Heidelberg (2002)
15. Lagoudakis, M.G., Parr, R.: Reinforcement learning as classification: Leveraging modern classifiers. In: ICML (2003)



16. Salton, G. (ed.): The SMART Retrieval System - Experiments in Automatic Document Processing. Prentice Hall, Englewood (1971)
17. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for SVMs (2001)
18. Sebastiani, F.: Machine learning in automated text categorization. *ACM Computing Surveys* 34(1), 1–47 (2002)
19. Kumar, M.A., Gopal, M.: A comparison study on multiple binary-class SVM methods for unilabel text categorization. *Pattern Recognition Letters* 31(11), 1437–1444 (2010)
20. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., Slattery, S.: Learning to extract symbolic knowledge from the World Wide Web. In: *World* (1998)

# Domain Adaptation for Text Categorization by Feature Labeling

Cristina Kadar and José Iria

IBM Research Zurich,  
Säumerstrasse 4, CH-8804 Rüschlikon, Switzerland  
cristina.kadar@gmail.com, jir@zurich.ibm.com

**Abstract.** We present a novel approach to domain adaptation for text categorization, which merely requires that the source domain data are weakly annotated in the form of labeled features. The main advantage of our approach resides in the fact that labeling words is less expensive than labeling documents. We propose two methods, the first of which seeks to minimize the divergence between the distributions of the source domain, which contains labeled features, and the target domain, which contains only unlabeled data. The second method augments the labeled features set in an unsupervised way, via the discovery of a shared latent concept space between source and target. We empirically show that our approach outperforms standard supervised and semi-supervised methods, and obtains results competitive to those reported by state-of-the-art domain adaptation methods, while requiring considerably less supervision.

**Keywords:** Domain Adaptation, Generalized Expectation Criteria, Weakly-Supervised Latent Dirichlet Allocation.

## 1 Introduction

The task of domain adaptation is fundamental to real-world text categorization problems, because the simplifying assumption, often made, that documents in the training set are drawn from the same underlying distribution as documents in the test set rarely holds in practice. As a consequence, statistical models derived from training data drawn from the “source” domain typically do not perform well on test data drawn from the “target” domain. For example, [18] report that a text classification model trained on a Yahoo! directory performed poorly on a Weblog classification problem, since the distribution of terms differed significantly.

At the heart of the difficulty in applying machine learning to new domains lies the fact that labeling problem examples is expensive. In particular, annotations of documents in the target domain are usually unavailable and expensive to acquire. Recently, a new labeling paradigm was introduced that enables learning from labeled features instead of labeled instances [6]. This provides two advantages: it reduces the amount of time spent in annotating, and therefore the cost, and it allows experts to more naturally, and thus more accurately, express their knowledge about the domain.

The feature labeling paradigm is particularly appealing for the domain adaptation task because it is often possible for domain experts to tell which features from the source domain are expected to apply robustly also in the target domain. This is easier and less time consuming than labeling documents. Unfortunately, approaches to domain adaptation have not considered the use of the feature labeling paradigm so far.

In this paper, we present a novel approach to domain adaptation for text categorization, which merely requires that the source data are weakly annotated in the form of labeled features. We propose two domain adaptation methods under this approach. The first method seeks to minimize the divergence between the distributions of the source domain, which contains labeled features, and the target domain, which contains only unlabeled data. The second method is similar to the first one, but can additionally make use of the labeled features to guide the discovery of a latent concept space, which is then used to augment the original labeled features set.

The contributions of the paper are fourfold: (i) we present, to the best of our knowledge, the first approach to domain adaptation for text categorization that relies on labeled words instead of labeled documents; (ii) we propose two different methods in order to analyse the merits of the approach (iii) we study the effect of the number of labeled features on the experimental results and verify that competitive results can be achieved even with a low number of labeled features; (iv) and we empirically show that our approach, despite only using a weak form of supervision, outperforms standard supervised and semi-supervised methods, and obtains results competitive with those previously reported by state-of-the-art methods that require the classic, more expensive, form of supervision – that of labeling documents.

The remainder of the paper is structured as follows. A brief review of related work on domain adaptation is given in the next section. In Section 3 we introduce the proposed domain adaptation methods. A complete description of the experimental setting is given in Section 4, and in Section 5 a comparative evaluation of the methods is presented, followed by a discussion on the results obtained. We conclude with a mention to our plans for future work.

## 2 Related Work

There are roughly two variants of the domain adaptation problem, which have been addressed in the literature: the supervised case and the semi-supervised case. In the former, we have at our disposal labeled documents from the source domain, and also a small amount of labeled documents from the target domain. The goal is to take advantage of both labeled datasets to obtain a model that performs well on the target domain. For example, [5, 7] work under this setting. The semi-supervised case differs in that no labeled documents in target exist, therefore the goal is to take advantage of an unannotated target corpus, see, e.g., [3, 9, 19, 4]. In this paper, we address the semi-supervised problem.

The problem of domain adaptation can be seen as that of finding a shared latent concept space that captures the relation between the two domains [16].

Therefore, several recent approaches sought an appropriate feature representation that is able to encode such shared concept space. [5] uses standard machine learning methods to train classifiers over data projected from both source and target domains into a high-dimensional feature space, via a simple heuristic nonlinear mapping function. In [14], the authors approach the problem from dimensionality reduction viewpoint. The method finds a low-dimensional latent feature space where the distributions between the source domain data and the target domain data are as close to each other as possible, and project onto this latent feature space the data from both domains. Standard learning algorithms can then be applied over the new space. A probabilistic approach in the same vein can be found in [19], where the authors propose an extension to the traditional probabilistic latent semantic analysis (PLSA) algorithm. The proposed algorithm is able to integrate the labeled source data and the unlabeled target data under a joint probabilistic model which aims at exploiting the common latent topics between two domains, and thus transfer knowledge across them through a topic-bridge to aid text classification in the target domain. Other relevant approaches following the same underlying principle include the feature extraction method described in [15], the method based on latent semantic association presented in [8] and the linear transformation method in [4] that takes into account the empirical loss on the source domain and the embedded distribution gap between the source and target domains.

Our approach may also be considered to belong to the above family of approaches in that we model a shared latent space between the domains, but with two major differences. First, it requires only labeled features instead of labeled instances. Second, the modeling of the latent space is not unsupervised, but partially supervised instead – by taking advantage of the availability of labeled features.

### 3 Domain Adaptation Using Labeled Features

Rather than requiring documents in the source and target domains to be examined and labeled, our approach to the domain adaptation problem leverages a small set of words that domain experts indicate to be positively correlated with each class – the labeled features. We adopt the *generalized expectation criteria* method [13,6] to translate this kind of domain knowledge into constraints on model expectations for certain word-class combinations. In what follows we briefly introduce this method, using the notation in [13], and then show how it can be used for domain adaptation.

A generalized expectation (GE) criterion is a term in a parameter estimation objective function that assigns scores to values of a model expectation. Let  $x$  be the input,  $y$  the output, and  $\theta$  the parameters for a given model. Given a set of unlabeled data  $\mathcal{U} = \{x\}$  and a conditional model  $p(y|x;\theta)$ , a GE criterion  $G(\theta;\mathcal{U})$  is defined by a score function  $V$  and a constraint function  $G(x,y)$ :

$$G(\theta;\mathcal{U}) = V(E_{\mathcal{U}}[E_{p(y|x;\theta)}[G(x,y)]]).$$

The GE formulation is generic enough to enable exploring many different choices of score functions and constraint functions. In this paper, we maximize the GE term together with an entropy regularization term in the objective function, although this can be easily combined with an empirical loss term to form a composite objective function that takes into account labeled instances as well. Moreover, we use *label regularization*, that is, the constraints are expectations of model marginal distributions on the expected output labels. As such, we use estimated label marginal distributions  $\tilde{g}_{x,y} = \tilde{p}(y)$  and consider constraints of the form  $G(x, y) = \mathbf{1}(y)$ . Model divergence from these constraints can be computed by using, for example, KL-divergence [11]:

$$G(\theta; \mathcal{U}) = -D(\tilde{p}(y) || E_{\mathcal{U}}[\mathbf{1}(y)p(y|x; \theta)]).$$

In order to use GE for domain adaptation, we derive criteria that encourage agreement between the source and target expectations. Let  $\mathcal{S}$  be source domain data and  $\mathcal{T}$  be target domain data, both unlabeled. We compute the model divergence for the task of domain adaptation by:

$$G(\theta; \mathcal{S}, \mathcal{T}) = - \sum_{i \in F(S \cup T)} D(\hat{p}(y|x_i > 0) || \tilde{p}_{\theta}(y|x_i > 0)), \quad (1)$$

where  $F$  is a function that returns the set of features in the input data,  $p(y|x_i > 0) = \frac{1}{C_i} \mathbf{1}(y) \mathbf{1}(x_i > 0)$  is an indicator of the presence of feature  $i$  in  $x$  times an indicator vector with 1 at the index corresponding to label  $y$  and zero elsewhere, and  $C_i = \sum_x \mathbf{1}(x_i > 0)$  is a normalizing constant;  $\tilde{p}_{\theta}$  denotes the predicted label distribution on the set of instances that contain feature  $i$  and  $\hat{p}$  are reference distributions derived from the labeled features. We estimate these reference distributions using the method proposed by [17]: let there be  $n$  classes associated with a given feature out of  $L$  total classes; then each associated class will have probability  $q_{maj}/n$  and each non-associated class has probability  $(1 - q_{maj})/(L - n)$ , where  $q_{maj}$  is set by the domain experts to indicate the correlation between the feature and the class.

To encourage the model to have non-zero values on parameters for unlabeled features that co-occur often with a labeled feature, we select as regularizer the Gaussian prior on parameters, which prefers parameter settings with many small values over settings with a few large values. The combined objective function is finally:

$$\mathcal{O} = - \sum_{i \in F(S \cup T)} D(\hat{p}(y|x_i > 0) || \tilde{p}_{\theta}(y|x_i > 0)) - \sum_j \frac{\theta_j^2}{2\sigma^2}, \quad (2)$$

consisting of a GE term for each for each labeled feature  $i$ , and a zero-mean  $\sigma^2$ -variance Gaussian prior on parameters.

We designed two methods that follow the proposed feature labeling approach to text categorization and the GE formulation above. As per equation (1), both methods are multi-class and semi-supervised (in that they make use of the unlabeled target domain data). The first method, which we will designate as *TransferLF*, directly uses the input labeled features to derive the reference

distributions  $\hat{p}$  (in the way described earlier). Then, given the latter and unlabeled source and target domain datasets, it estimates the classification model parameters by using an optimization algorithm, taking equation (2) as the objective function.

The second method, which we will designate as *TransferzLDALF*, is similar to the first one, but additionally aims at augmenting the set of input labeled features with new labeled features derived from the target domain data. In the same vein as related work in section 2, to discover and label new features our idea is to find a shared latent concept space that captures the relation between the two domains and bridges source and target features. This can be achieved in an unsupervised manner by using latent topic models such as Latent Dirichlet Allocation (LDA) [2]; however, we are interested in encouraging the recovery of topics that are more relevant to the domain expert’s modeling goals, as expressed by the labeled features provided, than the topics which would otherwise be recovered in an unsupervised way. Weak supervision in LDA was recently introduced in works such as [1,20]. With this goal in mind, we rehash the approach in [1], which adds supervision to LDA in the form of so-called *z-labels*, i.e., knowledge that the topic assignment for a given word position is within a subset of topics. Thus, in addition to their role in GE, we use the input labeled features as *z-labels*, in order to obtain feature clusters (containing both source and target features) where each cluster respects to one topic from the set of topics found in the labeled features. We are then able to augment the original labeled features set with the  $k$  most probable target domain features present in each cluster, in hope that the additional GE constraints lead to improved performance.

The algorithm for inducing a text categorization classifier for both methods is shown below. The first two steps only apply to *TransferzLDALF*.

---

**Algorithm 1.** TransferLF and TransferzLDALF

---

**Input:** labeled features  $\mathcal{L}$ , unlabeled source  $\mathcal{S}$  and target  $\mathcal{T}$  domain data

**Output:** induced classifier  $\mathcal{C}$

---

*TransferzLDALF* only:

- (1)  $\mathcal{L}_{\mathcal{LDA}}$  = labeled features from weakly-supervised LDA using input  $\mathcal{L}$ ,  $\mathcal{S}$  and  $\mathcal{T}$
- (2) Augment  $\mathcal{L}$  with  $k$  target domain features per topic from  $\mathcal{L}_{\mathcal{LDA}}$

*TransferLF* and *TransferzLDALF*:

- (3) Compute reference distributions  $\hat{p}(y|x_i > 0)$  from  $\mathcal{L}$
  - (4) Estimate model parameters by running optimization algorithm according to eq. (2)
  - (5) **return** induced classifier  $\mathcal{C}$
- 

## 4 Experiments

The first of the datasets chosen for our empirical analysis is K. Lang’s original 20-newsgroups [1] dataset [12]. It contains approximately 20,000 documents

<sup>1</sup> <http://www.cs.umass.edu/~mccallum/code-data.html>

that correspond to English-language posts to 20 different newsgroups. There are roughly 1000 documents in each category. The topic hierarchy for this dataset contains four major groups: *sci* (scientific), *rec* (recreative), *talk* (discussion) and *comp* (computers), with 3 to 5 topics under each group. The second dataset used in our experiments is the SRAA<sup>1</sup> corpus. It contains messages about simulated auto racing, simulated aviation, real autos and real aviation from 4 discussion groups. We used the first 4,000 documents from each of the classes in this dataset.

For the purposes of evaluating domain adaptation, we gather documents drawn from related topics, having different distributions. For example, the newsgroups *rec.autos* and *rec.motorcycles* are both related to cars, whereas the newsgroups *rec.sport.baseball* and *rec.sport.hockey* both describe games. Plus, moving to the first level of the 20-newsgroups taxonomy, broader categories may also be built: recreational, talk, computers and scientific. The SRAA data set is split in a similar manner into four categories: auto, aviation, real, simulated. Table 1 summarizes the characteristics of the datasets used in the experiments, indicating the source vs. target splits and the KL-divergence [11] measuring the distribution gap between the domains [2].

Minimal preprocessing was applied on the data: lowercasing the input and removing a list of English stopwords. Each document is represented as a vector of words and their frequency in the corpus. We use the MALLET [3] toolkit to solve the optimization problem using L-BFGS, a quasi-Newton optimization method that estimates the model parameters.

Human domain expertise is replaced in our experiments by an oracle-labeler – an experimental setup also adopted in, e.g., [6]. Making use of the true instance labels, the oracle computes the mutual information of the features within each class, and, if above a given threshold, labels the feature with the class under which it occurs most often, and also with any other class under which it occurs at least half as often. In the experiments we use as threshold the mean of the mutual information scores of the top  $100L$  most predictive features, where  $L$  is the number of classes; and  $q_{maj} = 0.9$  as the majority of the probability mass to be distributed among classes associated to a labeled feature. The oracle is very conservative in practice – refer to Table 3 for the actual number of labeled features for each source domain.

Finally, zLDA [4] was chosen as an implementation of the semi-supervised LDA method. We use the original labeled features as seeds for their latent topics and run the algorithm in its standard setup, as reported in [1]:  $\alpha = .5$ ,  $\beta = .1$ , 2000 samples. Table 2 shows an example concerning the *Cars vs Hardware* experiment. The oracle identified and labeled 17 and 40 features, respectively. They all come from the source domains: *rec.autos* and *comp.sys.pc.ibm.hardware*, respectively. With these as input, zLDA identifies new associated features that are specific to the target (e.g. *bike* for *rec.motorcycles* and *apple* for *comp.sys.mac.hardware*).

<sup>2</sup> It may be noted that the obtained KL-divergence values are considerably larger than if we were to split randomly, which would yield values close to zero.

<sup>3</sup> <http://www.mallet.cs.umass.edu>

<sup>4</sup> <http://pages.cs.wisc.edu/~andrzejew/software.html>

**Table 1.** Characteristics of the datasets used for evaluating the proposed approach

Dataset	Source Data	Target Data	KL divergence
Cars vs Games	rec.autos rec.sport.baseball	rec.motorcycles rec.sport.hockey	0.5679
Cars vs. Hardware	rec.autos comp.sys.ibm.pc.hardware	rec.motorcycles comp.sys.mac.hardware	0.4136
Cars vs Games vs Hardware vs OS	rec.autos rec.sport.baseball comp.sys.ibm.pc.hardware comp.windows.x	rec.motorcycles rec.sport.hockey comp.sys.mac.hardware comp.os.ms-windows.misc	0.4579
Cars vs Games vs Hardware vs OS vs Politics vs Religion	rec.autos rec.sport.baseball comp.sys.ibm.pc.hardware comp.windows.x talk.politics.mideast soc.religion.christian	rec.motorcycles rec.sport.hockey comp.sys.mac.hardware comp.os.ms-windows.misc talk.politics.misc talk.religion.misc	0.3701
Comp vs Sci	comp.graphics comp.os.ms-windows.misc sci.crypt sci.electronics	comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x sci.med sci.space	0.3897
Rec vs Talk	rec.autos rec.motorcycles talk.politics.guns talk.politics.misc	rec.sport.baseball rec.sport.hockey talk.politics.mideast talk.religion.misc	0.5101
Comp vs Rec	comp.graphics comp.sys.ibm.pc.hardware comp.sys.mac.hardware rec.motorcycles rec.sport.hockey	comp.os.ms-windows.misc comp.windows.x rec.autos rec.sport.baseball	0.4741
Comp vs Talk	comp.graphics comp.sys.mac.hardware comp.windows.x talk.politics.mideast talk.religion.misc	comp.sys.ibm.pc.hardware comp.sys.mac.hardware talk.politics.guns talk.politics.misc	0.2848
Auto vs Aviation	rec.autos.simulators rec.aviation.simulators	rec.autos.misc rec.aviation.student	0.8152
Real vs Simulated	rec.autos.misc rec.autos.simulators	rec.aviation.student rec.aviation.simulators	0.6532



**Table 2.** Initial labeled features and discovered zLDA features for *Cars vs Hardware*

Class	initial seed words	top 18 words in topic
Cars	article writes car cars wheel miles toyota honda driving engine oil engines ford rear year auto autos	writes article car good <b>bike</b> time back people cars make year thing engine <b>ride</b> years <b>road</b> work front
Hardware	advance windows disk system drives computer dx software bus mode os ibm memory machine monitor dos hardware board chip card cards ram mb pc interface vlb mhz cache ide cpu controller port modem motherboard gateway scsi video isa bios floppy	system drive problem computer work <b>mac</b> card mail <b>apple</b> software mb good time pc problems disk board bit

**Table 3.** Classification accuracies and the amount of labeled information (either instances or features) used in different sets of experiments. Note that for the *TransferzLDALF* method, the reported results correspond to selecting a fixed number of 18 features per topic (cf. learning curves), but the features outputted by zLDA can overlap and thus the size of the feature set used is smaller when merged.

Dataset	# source labeled instances	MaxEnt	# source labeled features	TransferLF on source	TransferLF	# zLDA labeled features	TransferLF with zLDA features
Cars vs Games	2000	90.3	52	84.7	<b>96.1</b>	29	92.8
Cars vs. Hardware	2000	90.7	57	88.2	<b>94.2</b>	32	88.7
Cars vs Games vs Hardware vs OS	4000	76.0	109	72.3	<b>80.9</b>	60	78.8
Cars vs Games vs Hardware vs OS vs Politics vs Religion	6000	67.1	167	63.0	69	81	<b>70.2</b>
Comp vs Sci	4000	71.8	59	76.1	78.4	30	<b>82.2</b>
Rec vs Talk	3874	77.9	60	74.3	74.5	29	<b>92.8</b>
Comp vs Rec	5000	87.9	70	86.1	<b>91.3</b>	32	86.7
Comp vs Talk	5000	93.3	67	91	<b>94.1</b>	33	94.0
Auto vs Aviation	8000	77.2	48	78.0	86.9	29	<b>91.6</b>
Real vs Simulated	8000	63.9	54	60.4	59.7	30	<b>77.7</b>

## 5 Results and Discussion

The results are presented using *accuracy* as evaluation metric:  $Acc = (tp+tn)/d$ , where  $tp$  are the true positives,  $tn$  the true negatives, and  $d$  the total number of documents in the corpus. In all comparisons, care was taken to reproduce the original authors' experimental setting with rigour.

**Table 4.** Performance comparison with [4]

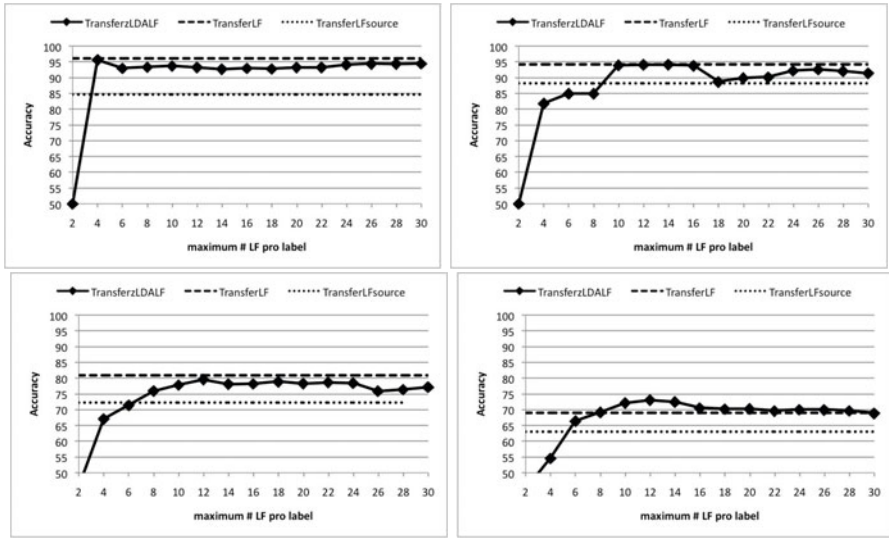
Dataset	TSVM	MMD	TransferLF	TransferLF with zLDA features
Cars vs Games	87.4	94.5	96.1	92.8
Cars vs Hardware	92.5	94.6	94.2	88.7
Cars vs Games vs Hardware vs OS	75.4	82.4	80.9	78.8

**Table 5.** Performance comparison with [19]

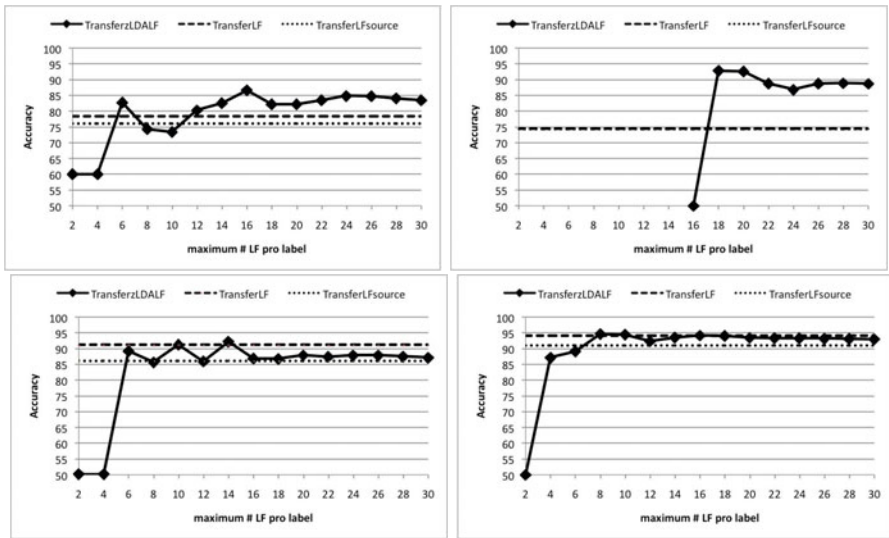
Dataset	TSVM	TPLSA	TransferLF	TransferLF with zLDA features
Comp vs Sci	81.7	98.9	78.4	82.2
Rec vs Talk	96	97.7	74.5	92.8
Comp vs Rec	90.2	95.1	91.3	86.7
Comp vs Talk	90.3	97.7	94.1	94.0
Auto vs Aviation	89.8	94.7	86.9	91.6
Real vs Simulated	87	88.9	59.7	77.7

Table 3 presents the results obtained from running the experiments on the several configurations shown in Table 1. We present results concerning two classifiers which are induced from the source domain data only: a standard supervised maximum entropy classifier as a baseline, and our proposed *TransferLF* method prevented from looking at the target domain data. The results show that our feature labeling approach to domain adaptation invariably outperforms the baseline non-domain-adaptation maximum entropy approach, while, in addition, greatly reduces the supervision requirements – compare the number of labeled features against the number of labeled instances used to induce the classifiers. It should be remarked that this is observed not only in the binary classification case, but also in the multi-class classification case. The results also suggest that the semi-supervised nature of the proposed methods is a differentiating factor, since *TransferLF* using source domain data only consistently underperforms.

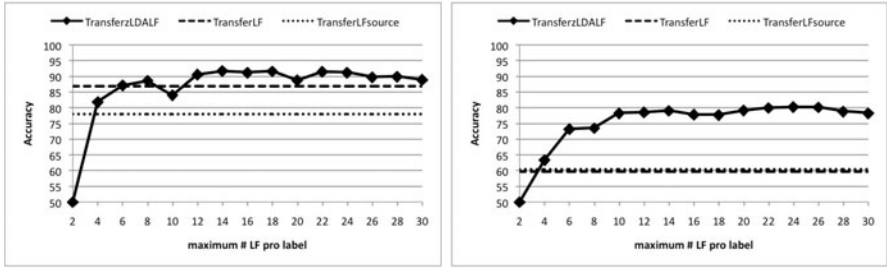
Tables 4 and 5 compare our approach with semi-supervised and latent semantic analysis-based techniques for domain adaptation in the literature. Transductive Support Vector Machines (TSVM) [10] are used as our baseline semi-supervised text classification approach. Refer to Section 2 for a brief description of MMD [4] and TPLSA [19]. It can be observed that the performance of the proposed methods is comparable with that of TSVM, which, again, is remarkable given that only a few labeled features are required to achieve that. The state-of-the-art MMD and TPLSA approaches still obtain higher accuracy



**Fig. 1.** Learning curves for the first dataset generated from 20-newsgroups. From left to right descending: *Cars vs Games*, *Cars vs Hardware*, *Cars vs Games vs Hardware* vs *OS*, and *Cars vs Games vs Hardware vs OS vs Politics vs Religion*.



**Fig. 2.** Learning curves for the second dataset generated from 20-newsgroups. From left to right descending: *Comp vs Sci*, *Rec vs Talk*, *Comp vs Rec*, and *Comp vs Talk*.



**Fig. 3.** Learning curves for the dataset generated from SRAA. From left to right: *Auto vs Aviation* and *Real vs Simulated*.

in general, which is not surprising given that their supervision requirements are much greater, but it is still very interesting to see how the results obtained by the feature labeling approach remain competitive. This is important, since in many application domains the reduction of the annotation effort is an enabling factor, at the expense of a only few accuracy points.

Finally, Figures 1, 2 and 3 show the learning curves obtained by varying the number of labeled features input to the *TransferzLDALF* method. From these curves we are able to obtain a deeper insight into the supervision requirements of our proposed approach. We conclude that as little as 5 features per topic are enough to achieve performances close to the plateau of the curve, as seen in some of the experiments, and that, on average, around 18 features per topic are enough to achieve top accuracy for the majority of the experiments.

## 6 Conclusions and Future Work

In this paper, we presented a novel approach to domain adaptation for text categorization that aims at reducing the effort in porting existing statistical models induced from corpora in one domain to other related domains. Our approach is based on the new paradigm of labeling words (as opposed to labeling whole documents), which is less time consuming and more natural for domain experts. It is our expectation that the proposed approach will introduce quantifiable benefits in several information retrieval application domains.

There are several possible avenues for future work that we would like to explore. First, we will study the interplay between labeled features and labeled documents through a thorough set of experiments which will allow us to analyse the behaviour of the induced model under varying amounts of labeled features and labeled documents in both source and target. Second, we plan to design a bootstrapping algorithm that makes use of labeled features to iteratively refine models of both source and target. Finally, we are currently developing a prototype system that implements our approach in the context of a real-world problem of classifying the textual part of tickets reporting on IT system problems.

## References

1. Andrzejewski, D., Zhu, X.: Latent Dirichlet Allocation with Topic-in-Set Knowledge. In: NAACL-SSLNLP (2009)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *Journal of Machine Learning* 3, 993–1022 (2003)
3. Blitzer, J., McDonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: EMNLP (2006)
4. Chen, B., Lam, W., Tsang, I., Wong, T.L.: Extracting discriminative concepts for domain adaptation in text mining. In: KDD (2009)
5. Daume III., H.: Frustratingly easy domain adaptation. In: ACL (2007)
6. Druck, G., Mann, G., McCallum, A.: Learning from labeled features using generalized expectation criteria. In: SIGIR (2008)
7. Finkel, J.R., Manning, C.D.: Hierarchical bayesian domain adaptation. In: NAACL (2009)
8. Guo, H., Zhu, H., Guo, Z., Zhang, X., Wu, X., Su, Z.: Domain adaptation with latent semantic association for named entity recognition. In: NAACL (2009)
9. Jiang, J., Zhai, C.: Instance weighting for domain adaptation in nlp. In: ACL (2007)
10. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machines. In: ICML (1999)
11. Kullback, S., Leibler, R.A.: On Information and Sufficiency. *Annals of Mathematical Statistics* 22(1), 79–86 (1951)
12. Lang, K.: NewsWeeder: Learning to Filter Netnews. In: ICML (1995)
13. Mann, G.S., McCallum, A.: Generalized Expectation Criteria for Semi-Supervised Learning with Weakly Labeled Data. *Journal of Machine Learning* 11, 955–984 (2010)
14. Pan, S.J., Kwok, J.T., Yang, Q.: Transfer learning via dimensionality reduction. In: AAAI (2008)
15. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. In: IJCAI (2009)
16. Ben-David, S., Blitzer, J., Crammer, K., Pereira, F.: Analysis of representations for domain adaptation. In: NIPS (2006)
17. Schapire, R., Rochery, M., Rahim, M., Gupta, N.: Incorporating prior knowledge into boosting. In: ICML (2002)
18. Ni, X., Xue, G.-R., Ling, X., Yu, Y., Yang, Q.: Exploring in the Weblog Space by Detecting Informative and Affective Articles. In: WWW (2007)
19. Xue, G., Dai, W., Yang, Q., Yu, Y.: Topic-bridged PLSA for cross-domain text classification. In: SIGIR (2008)
20. Xu, G., Yang, S.-H., Li, H.: Named Entity Mining from Click-Through Data Using Weakly Supervised Latent Dirichlet Allocation. In: KDD (2009)

# TEMPER: A Temporal Relevance Feedback Method

Mostafa Keikha, Shima Gerani, and Fabio Crestani

University of Lugano, Lugano, Switzerland

{mostafa.keikha, shima.gerani, fabio.crestani}@usi.ch

**Abstract.** The goal of a blog distillation (blog feed search) method is to rank blogs according to their recurrent relevance to the query. An interesting property of blog distillation which differentiates it from traditional retrieval tasks is its dependency on time. In this paper we investigate the effect of time dependency in query expansion. We propose a framework, TEMPER, which selects different terms for different times and ranks blogs according to their relevancy to the query over time. By generating multiple expanded queries based on time, we are able to capture the dynamics of the topic both in aspects and vocabulary usage. We show performance gains over the baseline techniques which generate a single expanded query using the top retrieved posts or blogs irrespective of time.

## 1 Introduction

User generated content is growing very fast and becoming one of the most important sources of information on the Web. Blogs are one of the main sources of information in this category. Millions of people write about their experiences and express their opinions in blogs everyday.

Considering this huge amount of user generated data and its specific properties, designing new retrieval methods is necessary to facilitate addressing different types of information needs that blog users may have. Users' information needs in blogosphere are different from those of general Web users. Mishne and de Rijke [1] analyzed a blog query log and accordingly they divided blog queries into two broad categories called context and concept queries. In context queries users are looking for contexts of blogs in which a Named Entity occurred to find out what bloggers say about it, whereas in concept queries they are looking for blogs which deal with one of searcher's topics of interest. In this paper we focus on the blog distillation task (also known as blog feed search) [2] where the goal is to answer topics from the second category [2].

Blog distillation is concerned with ranking blogs according to their recurring central interest to the topic of a user's query. In other words, our aim is to discover relevant blogs for each topic [2] that a user can add to his reader and read them in future [3].

<sup>1</sup> In this paper we use words "feed" and "blog" interchangeably.

<sup>2</sup> In this paper we use words "topic" and "query" interchangeably.

An important aspect of blog distillation, which differentiates it from other IR tasks, is related to the temporal properties of blogs and topics. Distillation topics are often multifaceted and can be discussed from different perspectives [4]. Vocabulary usage in the relevant documents to a topic can change over time in order to express different aspects (or sub-topics) of the query. These dynamics might create term mismatch problem during the time, such that a query term may not be a good indicator of the query topic in all different time intervals. In order to address this problem, we propose a time-based query expansion method which expands queries with different terms at different times. This contrasts other applied query expansion methods in blog search where they generate only one single query in the expansion phase [5,4]. Our experiments on different test collections and different baseline methods indicate that time-base query expansion is effective in improving the retrieval performance and can outperform existing techniques.

The rest of the paper is organized as follows. In section 2 we review state of the art methods in blog retrieval. Section 3 describes existing query expansion methods for blog retrieval in more detail. Section 4 explains our time-based query expansion approach. Experimental results over different blog data sets are discussed in section 6. Finally, we conclude the paper and describe future work in section 7.

## 2 Related Work

The main research on the blog distillation started after 2007, when the TREC organizers proposed this task in the blog track [3]. Researchers have applied different methods from areas that are similar to blog distillation, like ad-hoc search, expert search and resource selection in distributed information retrieval.

The most simple models use ad-hoc search methods for finding relevant blogs to a specific topic. They treat each blog as one long document created by concatenating all of its posts together [6,4,7]. These methods ignore any specific property of blogs and mostly use standard IR techniques to rank blogs. Despite their simplicity, these methods perform fairly well in blog retrieval.

Some other approaches have been applied from expert search methods in blog retrieval [8,2]. In these models, each post in a blog is seen as evidence that the blog has an interest in the query topic. In [2], MacDonald *et al.* use data fusion models to combine this evidence and compute a final relevance score for the blog, while Balog *et al.* adapt two language modeling approaches of expert finding and show their effectiveness in blog distillation [8].

Resource selection methods from distributed information retrieval have been also applied to blog retrieval [4,9,7]. Elsas *et al.* deal with blog distillation as a recourse selection problem [4,9]. They model each blog as a collection of posts and use a Language Modeling approach to select the best collection. A similar approach is proposed by Seo and Croft [7], which they call Pseudo Cluster Selection. They create topic-based clusters of posts in each blog and select blogs that have the most similar clusters to the query.

Temporal properties of posts have been considered in different ways in blog retrieval. Nunes *et al.* define two new measures called “temporal span” and “temporal dispersion” to evaluate “how long” and “how frequently” a blog has been writing about a topic [10]. Similarly Macdonald and Ounis [2] use a heuristic measure to capture the recurring interests of blogs over time. Some other approaches give higher scores to more recent posts before aggregating them [11,12]. All these proposed methods and their improvements show the importance and usefulness of temporal information in blog retrieval. However, none of the mentioned methods investigates the effect of time on the vocabulary change for a topic. We employ the temporal information as a source to distinguish between different aspects of topic and terms that are used for each aspect. This leads us to a time-based query expansion method where we generate multiple expanded queries to cover multiple aspects of a topic over time.

Different query expansion possibilities for blog retrieval have been explored by Elsas *et al.* [4] and Lee *et al.* [5]. Since we use these methods as our baselines, we will discuss them in more detail in the next section.

### 3 Query Expansion in Blog Retrieval

Query expansion is known to be effective in improving the performance of the retrieval systems [13,14,15]. In general the idea is to add more terms to an initial query in order to disambiguate the query and solve the possible term mismatch problem between the query and the relevant documents. Automatic Query Expansion techniques usually assume that top retrieved documents are relevant to the topic and use their content to generate an expanded query. In some situations, it has been shown that it is better to have multiple expanded queries as apposed to the usual single query expansion, for example in server-based query expansion technique in distributed information retrieval [16].

An expanded query, while being relevant to the original query, should have as much coverage as possible on all aspects of the query. If the expanded query is very specific to some aspect of the original query, we will miss part of the relevant documents in the re-ranking phase. In blog search context, where queries are more general than normal web search queries [4], the coverage of the expanded query gets even more important. Thus in this condition, it might be better to have multiple queries where each one covers different aspects of a general query.

Elsas *et al.* made the first investigation on the query expansion techniques for blog search [4]. They show that normal feedback methods (selecting the new terms from top retrieved posts or top retrieved blogs) using the usual parameter settings is not effective in blog retrieval. However, they show that expanding query using an external resource like Wikipedia can improve the performance of the system. In a more recent work, Lee *et al.* [5] propose new methods for selecting appropriate posts as the source of expansion and show that these methods can be effective in retrieval. All these proposed methods can be summarized as follows:

- Top Feeds: Uses all the posts of the top retrieved feeds for the query expansion. This model has two parameters including number of selected feeds and number of the terms in the expanded query [4].



- Top Posts: Uses the top retrieved posts for the query expansion. Number of the selected posts and number of the terms to use for expansion are the parameters of this model [4].
- FFBS: Uses the top posts in the top retrieved feeds as the source for selecting the new terms. Number of the selected posts from each feed is fixed among different feeds. This model has three parameters; number of the selected feeds, number of the selected posts in each feed and number of the selected terms for the expansion [5].
- WFBS: Works the same as FFBS. The only difference is that number of the selected posts for each feed depends on the feed rank in the initial list, such that more relevant feeds contribute more in generating the new query. Like FFBS, WFBS has also three parameters that are number of the selected feeds, total number of the posts to be used in the expansion and number of the selected terms [5].

Among the mentioned methods, “Top Feeds” method has the possibility to expand the query with non-relevant terms. The reason is that all the posts in a top retrieved feed are not necessarily relevant to the topic. On the other hand, “Top Posts” method might not have enough coverage on all the sub-topics of the query, because the top retrieved posts might be mainly relevant to some dominant aspect of the query. FFBS and WFBS methods were originally proposed in order to have more coverage than the “Top Posts” method while selecting more relevant terms than the “Top Feeds” method [5]. However, since it is difficult to summarize all the aspects of the topic in one single expanded query, these methods would not have the maximum possible coverage.

## 4 TEMPER

In this section we describe our novel framework for time-based relevance feedback in blog distillation called TEMPER. TEMPER assumes that posts at different times talk about different aspects (sub-topics) of a general topic. Therefore, vocabulary usage for the topic is time-dependant and this dependancy can be considered in a relevance feedback method. Following this intuition, TEMPER selects time-dependent terms for query expansion and generated one query for each time point. We can summarize the TEMPER framework in the following 3 steps:

1. Time-based representation of blogs and queries
2. Time-based similarity between a blogs and a query
3. Ranking blogs according to the their overall similarity to the query.

In the remainder of this section, we describe our approach in fulfilling each of these steps.

### 4.1 Time-Based Representation of Blogs and Queries

**Initial Representation of Blogs and Queries.** In order to consider time in the TEMPER framework, we first need to represent blogs and queries in the time space.

For a blog representation, we distribute its posts based on their publish date. In order to have a daily representation of the blog, we concatenate all the posts that have the same date.

For a query representation, we take advantage of the top retrieved posts for the query. Same as blog representation, we select the top  $K$  relevant posts for the query and divide them based on their publish date while concatenating posts with the same date. In order to have a more informative representation of the query, we select the top  $N$  terms for each day using the KL-divergence between the term distribution of the day and the whole collection [17].

Note that in the initial representation, there can be days that do not have any term distribution associated with them. However, in order to calculate the relevance of a blog to a query, TEMPER needs to have the representation of the blog and query in all the days. We employ the available information in the initial representation to estimate the term distributions for the rest of the days. In the rest of this section, we explain our method for estimating these representations.

**Term Distributions over Time.** TEMPER generates a representation for each topic or blog for each day based on the idea that a term at each time position propagates its count to the other time positions through a proximity-based density function. By doing so, we can have a virtual document for a blog/topic at each specific time position. The term frequencies of such a document is calculated as follows:

$$tf'(t, d, i) = \sum_{j=1}^T tf(t, d, j)K(i, j) \quad (1)$$

where  $i$  and  $j$  indicate time position (day) in the time space.  $T$  denotes the time span of the collection.  $tf'$  shows the term frequency of term  $t$  in blog/topic  $d$  at day  $i$  and it is calculated based on the frequency of  $t$  in all days.  $K(i, j)$  decreases as the distance between  $i$  and  $j$  increases and can be calculated using kernel functions that we describe later.

The proposed representation of document in the time space is similar to the proximity-based method where they generate a virtual document at each position of the document in order to capture the proximity of the words [18,19]. However, here we aim to capture the temporal proximity of terms. In this paper we employ the laplace kernel function which has been shown to be effective in a previous work [19] together with the Rectangular (square) kernel function. In the following formulas, we present normalized kernel functions with their corresponding variance formula.

## 1. Laplace Kernel

$$k(i, j) = \frac{1}{2b} \exp \left[ \frac{-|i - j|}{b} \right] \quad (2)$$

where  $\sigma^2 = 2b^2$

## 2. Rectangular Kernel

$$k(i, j) = \begin{cases} \frac{1}{2a} & \text{if } |i - j| \leq a \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where  $\sigma^2 = \frac{a^2}{3}$

### 4.2 Time-Based Similarity Measure

By having the daily representation of queries and blogs, we can calculate the daily similarity between these two representations and create a *daily similarity vector* for the blog and the query. The final similarity between the blog and the query is then calculated by summing over the daily similarities:

$$sim_{temporal}(B, Q) = \sum_{i=1}^T sim(B, Q, i) \quad (4)$$

where  $sim(B_i, Q_i)$  shows the similarity between a blog and a query representation at day  $i$  and  $T$  shows the time span of the collection in days.

Another popular method in time series similarity calculation is to see each time point as one dimension in the time space and use the euclidian length of the *daily similarity vector* as the final similarity between the two representations [20]:

$$sim_{temporal}(B, Q) = \sqrt{\sum_{i=1}^T sim(B, Q, i)^2} \quad (5)$$

We use the cosine similarity as a simple and effective similarity measure for calculating similarity between the blog and the topic representations at the specific day  $i$ :

$$sim(B, Q, i) = \frac{\sum_w tf(w, B, i) \times tf(w, Q, i)}{\sqrt{\sum_w tf(w, B, i)^2 \times \sum_w tf(w, Q, i)^2}} \quad (6)$$

The normalized value of the temporal similarity over all blogs is then used as  $P_{temporal}$ .

$$P_{temporal}(B|Q) = \frac{sim_{temporal}(B, Q)}{\sum_{B'} sim_{temporal}(B', Q)} \quad (7)$$

Finally in order to take advantage of all the available evidence regarding the blog relevance, we interpolate the temporal score of the blog with its initial relevance score.

$$P(B|Q) = \alpha P_{initial}(B|Q) + (1 - \alpha) P_{temporal}(B|Q) \quad (8)$$

where  $\alpha$  is a parameter that controls the amount of temporal relevance that is considered in the model. We use the Blogger Model method for the initial

**Table 1.** Effect of cleaning the data set on Blogger Model. Statistically significant improvements at the 0.05 level is indicated by †.

Model	Cleaned	MAP	P@10	Bpref
BloggrModel	No	0.2432	0.3513	0.2620
BloggrModel	Yes	0.2774†	0.4154 †	0.2906†

ranking of the blogs [8]. The only difference with the original Blogger Model is that we set the prior of a blog to be proportional to the log of the number of its posts, as opposed to the uniform prior that was used in the original Blogger Model. This log-based prior has been used and shown to be effective by Elsas *et al.* [4].

## 5 Experimental Setup

In this section we first explain our experimental setup for evaluating the effectiveness of the proposed framework.

**Collection and Topics.** We conduct our experiments over three years worth of TREC blog track data from the blog distillation task, including TREC’07, TREC’08 and TREC’09 data sets. The TREC’07 and TREC’08 data sets include 45 and 50 assessed queries respectively and use Blog06 collection. The TREC’09 data set uses Blog08, a new collection of blogs, and has 39 new queries [3]. We use only the title of the topics as the queries.

The Blogs06 collection is a crawl of about one hundred thousand blogs over an 11-weeks period [22], and includes blog posts (permalinks), feed, and homepage for each blog. Blog08 is a collection of about one million blogs crawled over a year with the same structure as Blog06 collection [21]. In our experiments we only use the permalinks component of the collection, which consist of approximately 3.2 million documents for Blog06 and about 28.4 million documents for Blog08.

We use the Terrier Information Retrieval system [4] to index the collection with the default stemming and stopwords removal. The Language Modeling approach using the dirichlet-smoothing has been used to score the posts and retrieve top posts for each query.

**Retrieval Baselines.** We perform our feedback methods on the results of the Blogger Model method [8]. Therefore, Blogger Model is the first baseline against which, we will compare the performance of our proposed methods. The second set of baselines are the query expansion methods proposed in previous works [4,5]. In order to have a fair comparison, we implemented the mentioned query

<sup>3</sup> Initially there were 50 queries in TREC 2009 data set but some of them did not have relevant blogs for the selected facets and are removed in the official query set [21]. We do not use of the facets in this paper however we use the official query set to be able to compare with the TREC results.

<sup>4</sup> <http://ir.dcs.gla.ac.uk/terrier/>

**Table 2.** Evaluation results for the implemented models over TREC09 data set

Model	MAP	P@10	Bpref
BloggerModel	0.2774	0.4154	0.2906
TopFeeds	0.2735	0.3897	0.2848
TopPosts	0.2892	0.4230	0.3057
FFBS	0.2848	0.4128	0.3009
WFBS	0.2895	0.4077	0.3032
TEMPER-Rectangular-Sum	0.2967 †	0.4128	0.3116 †
TEMPER-Rectangular-Euclidian	0.3014 † ‡ *	<b>0.4435</b> *	0.3203 † ‡ *
TEMPER-Laplace-Sum	0.3086 †	0.4256	<b>0.3295</b> †
TEMPER-Laplace-Euclidian	<b>0.3122</b> † ‡ *	0.4307	0.3281 † *

expansion methods on top of Blogger Model. We tuned the parameters of these models using 10-fold cross validation in order to maximize MAP.

The last set of baselines are provided by TREC organizers as part of the blog facet distillation task. We use these baselines to see the effect of TEMPER in re-ranking the results of other retrieval systems.

**Evaluation.** We used the blog distillation relevance judgements provided by TREC for evaluation. We report the Mean Average Precision (MAP) as well as binary Preference (bPref), and Precision at 10 documents (P@10). Throughout our experiments we use the Wilcoxon signed ranked matched pairs test with a confidence level of 0.05 level for testing statistical significant improvements.

## 6 Experimental Results

In this section we explain the experiments that we conducted in order to evaluate the usefulness of the proposed method. We mainly focus on the results of TREC09 data set, as it is the most recent data set and has enough temporal information which is an important feature for our analysis. However, in order to see the effect of the method on the smaller collections, we briefly report the final results on the TREC07 and TREC08 data sets.

Table 1 shows the evaluation results of Blogger Model on TREC09 data set. Because of the blog data being highly noisy, we carry out a cleaning step on the collection in order to improve the overall performance of the system. We use the cleaning method proposed by Parapar *et al.* [23]. As we can see in Table 1, cleaning the collection is very useful and improves the MAP of the system about 14%. We can see that the results of Blogger Model on the cleaned data is already better than the best TREC09 submission on the title-only queries.

Table 2 summarizes retrieval performance of Blogger Model and the baseline query expansion methods along with different settings of TEMPER on the TREC 2009 data set. The best value in each column is bold face. A dag(†), a ddag(‡) and a star(\*) indicate statistically significant improvement over Blogger Model, TopPosts and WFBS respectively. As can be seen from the table, none of the query expansion baselines improves the underlying Blogger Model significantly.

**Table 3.** Evaluation results for the implemented models over TREC08 data set

Model	MAP	P@10	Bpref
BloggerModel	0.2453	0.4040	0.2979
TopPosts	0.2567	0.4080	0.3090
WFBS	0.2546	0.3860	0.3087
TEMPER-Laplace-Euclidian	<b>0.2727</b> † ‡ *	<b>0.4380</b> † ‡ *	<b>0.3302</b> † *

**Table 4.** Evaluation results for the implemented models over TREC07 data set

Model	MAP	P@10	Bpref
BloggerModel	0.3354	0.4956	0.3818
TopPosts	0.3524 †	0.5044	0.3910
WFBS	0.3542 †	<b>0.5356</b> † ‡	0.3980
TEMPER-Laplace-Euclidian	<b>0.3562</b> †	0.5111	<b>0.4011</b>

**Table 5.** Comparison with the best TREC09 title-only submissions

Model	MAP	P@10	Bpref
TEMPER-Laplace-Euclidian	<b>0.3122</b>	<b>0.4307</b>	<b>0.3281</b>
TREC09-rank1 (buptpris 2009)	0.2756	0.3206	0.2767
TREC09-rank2 (ICTNET)	0.2399	0.3513	0.2384
TREC09-rank3 (USI)	0.2326	0.3308	0.2409

From table 2 we can see that TEMPER with different settings (using rectangular/laplace kernel, sum/euclidean similarity method) improves Blogger Model and the query expansion methods significantly. These results show the effectiveness of time-based representation of blogs and query and highlights the importance of time-based similarity calculation of blogs and topics.

In tables 3 and 4 we present similar results over TREC08 and TREC07 data sets. Over the TREC08 dataset, it can be seen that TEMPER improves Blogger Model and different query expansion methods significantly. Over the TREC07 dataset, TEMPER improves Blogger Model significantly. However, the performance of TEMPER is comparable with the other query expansion methods and the difference is not statistically significant.

As it was mentioned in section 5, we also consider the three standard baselines provided by TREC10 organizers in order to see the effect of our proposed feedback method on retrieval baselines other than Blogger Model. Table 8 shows the results of TEMPER over the TREC baselines. It can be seen that TEMPER improves the baselines in most of the cases. The only baseline that TEMPER does not improve significantly is stdbaseline1<sup>5</sup>.

Tables 5, 6 and 7 show the performance of TEMPER compared to the best title-only TREC runs in 2009, 2008 and 2007 respectively. It can be seen from the tables that TEMPER is performing better than the best TREC runs over the

<sup>5</sup> Note that the stdbaselines are used as blackbox and we are not yet aware of the underlying method.

**Table 6.** Comparison with the best TREC08 title-only submissions

Model	MAP	P@10	Bpref
TEMPER-Laplace-Euclidian	0.2727	0.4380	0.3302
TREC08-rank2 (CMU-LTI-DIR)	<b>0.3056</b>	0.4340	0.3535
TREC08-rank1 (KLE)	0.3015	<b>0.4480</b>	<b>0.3580</b>
TREC08-rank3 (UAms)	0.2638	0.4200	0.3024

**Table 7.** Comparison with the best TREC07 title-only submissions

Model	MAP	P@10	Bpref
TEMPER-Laplace-Euclidian	0.3562 †	0.5111	<b>0.4011</b>
TREC07-rank1 (CMU)	<b>0.3695</b>	<b>0.5356</b>	0.3861
TREC07-rank2 (UGlasgow)	0.2923	0.5311	0.3210
TREC07-rank3 (UMass)	0.2529	0.5111	0.2902

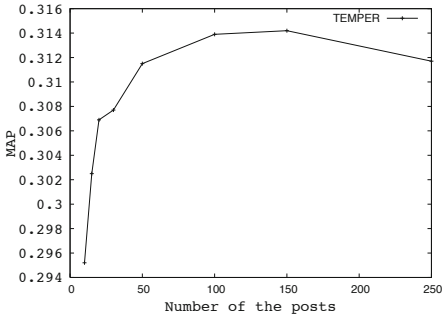
**Table 8.** Evaluation results for the standard baselines on TREC09 data set. Statistically significant improvements are indicated by †.

Model	MAP	P@10	Bpref
stdBaseline1	0.4066	0.5436	0.4150
TEMPER-stdBaseline1	0.4114	0.5359	0.4182
stdBaseline2	0.2739	0.4103	0.2845
TEMPER-stdBaseline2	0.3009†	0.4308 †	0.3158†
stdBaseline3	0.2057	0.3308	0.2259
TEMPER-stdBaseline3	0.2493†	0.4026†	0.2821†

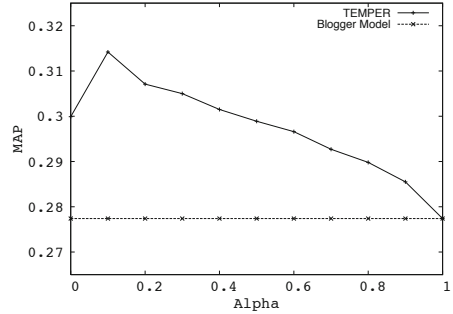
TREC09 dataset. The results over the TREC08 and TREC07 are comparable to the best TREC runs and can be considered as the third and second best reported results over TREC08 and TREC07 datasets respectively. TEMPER has four parameters including : number of the posts selected for expansion, number of the terms that are selected for each day, standard deviation ( $\sigma$ ) of the kernel functions and  $\alpha$  as the weight of the initial ranking score.

Among these parameters, we fix number of the terms for each day to be 50, as used in a previous work [4]. Standard deviation of the kernel function is estimated using top retrieve posts for each query. Since the goal of the kernel function is to model the distribution of distance between two consequent relevant posts, we assume the distances between selected posts (top retrieved posts) as the samples of this distribution. We then use the standard deviation of the sample as an estimation for  $\sigma$ .

The other two parameters are tuned using 10-fold cross validation method. Figure 1 and 2 show sensitivity of the system to these parameters. It can be seen that the best performance is gained by selecting about 150 posts for expansion while any number more than 50 gives a reasonable result. The value of  $\alpha$  depends on the underneath retrieval model. We can see that TEMPER outperforms Blogger Model for all values of  $\alpha$  and the best value is about 0.1.



**Fig. 1.** Effect of number of the posts used for expansion on the performance of TEMPER



**Fig. 2.** Effect of alpha on the performance of TEMPER

## 7 Conclusion and Future Work

In this paper we investigated blog distillation where the goal is to rank blogs according to their recurrent relevance to the topic of the query. We focused on the temporal properties of blogs and its application in query expansion for blog retrieval. Following the intuition that term distribution for a topic might change over time, we propose a time-based query expansion technique. We showed that it is effective to have multiple expanded queries for different time points and score the posts of each time using the corresponding expanded query. Our experiments on different blog collections and different baseline methods showed that this method can improve the state of the art query expansion techniques.

Future work will involve more analysis on temporal properties of blogs and topics. In particular, modeling the evolution of topics over time can help us to better estimate the topics relevance models. This modeling over time can be seen as a temporal relevance model which is an unexplored problem in blog retrieval.

## Acknowledgement

This work was supported by Swiss National Science Foundation (SNSF) as XMI project (ProjectNr. 200021-117994/1).

## References

1. Mishne, G., de Rijke, M.: A study of blog search. In: Lalmas, M., MacFarlane, A., Rüger, S.M., Tombros, A., Tsirikka, T., Yavlinsky, A. (eds.) ECIR 2006. LNCS, vol. 3936, pp. 289–301. Springer, Heidelberg (2006)
2. Macdonald, C., Ounis, I.: Key blog distillation: ranking aggregates. In: Proceedings of CIKM 2008, pp. 1043–1052 (2008)
3. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the trec-2007 blog track. In: Proceedings of TREC 2007 (2008)



4. Elsas, J.L., Arguello, J., Callan, J., Carbonell, J.G.: Retrieval and feedback models for blog feed search. In: Proceedings of SIGIR 2008, pp. 347–354 (2008)
5. Lee, Y., Na, S.H., Lee, J.H.: An improved feedback approach using relevant local posts for blog feed retrieval. In: Proceedings of CIKM 2009, pp. 1971–1974 (2009)
6. Efron, M., Turnbull, D., Ovalle, C.: University of Texas School of Information at TREC 2007. In: Proceedings of TREC 2007 (2008)
7. Seo, J., Croft, W.B.: Blog site search using resource selection. In: Proceedings of CIKM 2008, pp. 1053–1062. ACM, New York (2008)
8. Balog, K., de Rijke, M., Weerkamp, W.: Bloggers as experts: feed distillation using expert retrieval models. In: Proceedings of SIGIR 2008, pp. 753–754 (2008)
9. Arguello, J., Elsas, J., Callan, J., Carbonell, J.: Document representation and query expansion models for blog recommendation. In: Proceedings of ICWSM 2008 (2008)
10. Nunes, S., Ribeiro, C., David, G.: Feup at trec 2008 blog track: Using temporal evidence for ranking and feed distillation. In: Proceedings of TREC 2008 (2009)
11. Ernsting, B., Weerkamp, W., de Rijke, M.: Language modeling approaches to blog postand feed finding. In: Proceedings of TREC 2007 (2007)
12. Weerkamp, W., Balog, K., de Rijke, M.: Finding key bloggers, one post at a time. In: Proceedings of ECAI 2008, pp. 318–322 (2008)
13. Cao, G., Nie, J.Y., Gao, J., Robertson, S.: Selecting good expansion terms for pseudo-relevance feedback. In: Proceedings of SIGIR 2008, pp. 243–250. ACM, New York (2008)
14. Lavrenko, V., Croft, W.B.: Relevance based language models. In: Proceedings of SIGIR 2001, pp. 120–127. ACM, New York (2001)
15. Salton, G.: The SMART Retrieval System—Experiments in Automatic Document Processing. Prentice-Hall, Inc., Upper Saddle River (1971)
16. Shokouhi, M., Azzopardi, L., Thomas, P.: Effective query expansion for federated search. In: Proceedings of SIGIR 2009, pp. 427–434. ACM, New York (2009)
17. Zhai, C., Lafferty, J.D.: Model-based feedback in the language modeling approach to information retrieval, pp. 403–410 (2001)
18. Lv, Y., Zhai, C.: Positional language models for information retrieval. In: Proceedings SIGIR 2009, pp. 299–306 (2009)
19. Gerani, S., Carman, M.J., Crestani, F.: Proximity-based opinion retrieval. In: Proceedings of SIGIR 2010, pp. 403–410 (2010)
20. Keogh, E.J., Pazzani, M.J.: Relevance feedback retrieval of time series data. In: Proceeding of SIGIR 1999, pp. 183–190 (1999)
21. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the TREC-2009 Blog Track. In: Proceedings of TREC 2009 (2009)
22. Macdonald, C., Ounis, I.: The TREC Blogs06 collection: Creating and analysing a blog test collection. Department of Computer Science, University of Glasgow Tech Report TR-2006-224 (2006)
23. Parapar, J., López-Castro, J., Barreiro, Á.: Blog Posts and Comments Extraction and Impact on Retrieval Effectiveness. In: Proceeding of Spanish Conference on Information Retrieval 2010 (2010)

# Terms of a Feather: Content-Based News Recommendation and Discovery Using Twitter\*

Owen Phelan, Kevin McCarthy, Mike Bennett, and Barry Smyth

CLARITY: Centre for Sensor Web Technologies  
School of Computer Science & Informatics  
University College Dublin  
firstname.lastname@ucd.ie

**Abstract.** User-generated content has dominated the web's recent growth and today the so-called *real-time web* provides us with unprecedented access to the real-time opinions, views, and ratings of millions of users. For example, Twitter's 200m+ users are generating in the region of 1000+ tweets per second. In this work, we propose that this data can be harnessed as a useful source of recommendation knowledge. We describe a social news service called *Buzzer* that is capable of adapting to the conversations that are taking place on Twitter to ranking personal RSS subscriptions. This is achieved by a content-based approach of mining trending terms from both the public Twitter timeline and from the timeline of tweets published by a user's own Twitter friend subscriptions. We also present results of a live-user evaluation which demonstrates how these ranking strategies can add better item filtering and discovery value to conventional recency-based RSS ranking techniques.

## 1 Introduction

The real-time web (RTW) is emerging as new technologies enable a growing number of users to share information in multi-dimensional contexts. Sites such as *Twitter* ([www.twitter.com](http://www.twitter.com)), *Foursquare* ([www.foursquare.com](http://www.foursquare.com)) are platforms for real-time blogging, messaging and live video broadcasting to friends and a wider global audience. Companies can get instantaneous feedback on products and services from RTW sites such as *Blippr* ([www.blippr.com](http://www.blippr.com)). Our research focusses on the real-time web, in all of its various forms, as a potentially powerful source of recommendation data. For example, we consider the possibility of mining user profiles based on their Twitter postings. If so, we can use this profile information as a way to rank items, user recommendation, products and services for these users, even in the absence of more traditional forms of preference data or transaction histories [6]. We may also provide a practical solution to the cold-start problem [13] of sparse profiles of users' interests, an issue that has plagued many item discovery and recommender systems to date.

---

\* This work is gratefully supported by Science Foundation Ireland under Grant No. 07/CE/11147 CLARITY CSET.

Online news is a well-trodden research field, with many good reasons why IR and AI techniques have the potential to improve the way we consume news online. For a start there is the sheer volume of news stories that users must deal with, plus we have varied tastes and preferences with respect to what we are interesting in reading about. At the same time, news is a biased form of media that is increasingly driven by the stories that are capable of selling advertising. Niche stories that may be of interest to a small portion of readers often get buried. All of this has contributed to a long history of using *recommender systems* to help users navigate through the sea of stories that are published everyday based on learned profiles of users. For example, *Google News* (<http://news.google.com>) is a topically segregated mashup of a number of feeds, with automatic ranking strategies based on user interactions (click-histories & click-thrus) [5]. It is an example of a hybrid technique for news recommendation, as it utilises a user’s search keywords from Google itself as a support for explicit ratings. Another popular example is *Digg* ([www.digg.com](http://www.digg.com)), whose webpage rating service generally leads to a high overlap of selected topical news items [12].

This paper extends some of the previous work presented in [15], which described an early prototype of the Buzzer system, in two ways. First, we describe a more comprehensive and robust recommendation framework that has been extended both in terms of the different sources of recommendation knowledge and the recommendation strategies that it users. Secondly, we describe the result of a live-user evaluation with 35 users over a 1 month period, and based on more than 30,000 news stories and in excess of 50 million Twitter messages, the results of which describe interesting usage patterns compared to recency benchmarks.

## 2 Background

Many research opportunities remain when considering how to adapt recommendation techniques to tackle the so-called *information explosion* on the web. Digg, for example, mines implicit click-thrus of articles as well as ratings and user-tagging folksonomies as a basis of content retrieval for users [12]. One of the byproducts of Digg’s operation is that users’ browsing and sharing activities generally involve socially or temporally topical items, so as such it has been branded as a sort of news service [12]. Difficulties arise where it is necessary for many users to implicitly (click, share, tag) and explicitly (star or digg) rate items many times for those items to emerge as topical things. Also, there would be considerable item churn, that is, the corpus of data is constantly updating and item relevances are constantly fluctuating. The space of documents themselves could be defined by Brusilovsky and Henze as an *Open Corpus Adaptive Hypermedia System* in that there is an open corpus of documents (though topic specific), that can constantly change and expand [3].

Google News is a popular service that uses (mostly unpublished) recommendation techniques to filter 4500 partner news providers to present an aggregated view for registered users of popular and topical content [5]. Items are usually between several seconds to 30 days old, and appear on the “front page” based on click-thrus and key-word term Google—search queries. The ranking itself is

mostly based on click-thru rates of items, higher ranked items have more clicks. Issues arise with new and topical items struggling to get to the “front page”, as it is necessary for a critical-mass of clicks from many users. Das et al. [5] mostly describe scalability of the system as an issue with the service, and propose several techniques known to be capable of dealing with such issues. These included MinHash, Probabilistic Latent Semantic Indexing (PLSI) and Latent Semantic Hashing (LSH) as component algorithms in an overall hybrid system.

Content-based approaches are widely discussed in many branches of recommender systems [2,9,13,14]. Examples such as the News@Hand semantic system by Cantador et al. [4] show encouraging moves towards considering the content of the news items themselves. The authors use semantic annotation and ontologies to structure items into groups, while matching this to similarly structured user profiles of preferred items — unfortunately the success of these are based on the quality and existence of established domain ontologies. Our approach is to look at the most atomic components of the content, the individual terms themselves.

There is currently considerable research attention being paid to Twitter and the real-time web in general. RTW services provide access to new types of information and the real-time nature of these data streams provide as many opportunities as they do challenges. In addition, companies like Twitter and Yahoo have adopted a very open approach to making their data available and Twitter’s developer API provides researchers with access to a huge volume of information for example. It is no surprise then that the recent literature includes analyses of Twitter’s real-time data, largely with a view to developing an early understanding of why and how people are using services like Twitter [7,8,11]. For instance, the work of Kwak et al. [11] describes a very comprehensive analysis of Twitter users and Twitter usage, covering almost 42m users, nearly 1.5bn social connections, and over 100m tweets. In this work, the authors have examined reciprocity and homophily among Twitter users, they have compared a number of different ways to evaluate user influence, as well as investigating how information diffuses through the Twitter ecosystem as a result of social relationships and retweeting behaviour. Similarly, Krishnamurthy et al. identify classes of Twitter users based on behaviours and geographical dispersion [10]. They highlight the process of producing and consuming content based on retweet actions, where users source and disseminate information through the network.

We are interested in the potential to use near-ubiquitous user-generated content as a source of preference and profiling information in order to drive recommendation, as such in this research context Buzzer is termed a content-based recommender. User-generated content is inherently noisy but it is plentiful, and recently researchers have started to consider its utility in recommendation. There has been some recent work [17] on the role of tags in recommender systems, and researchers have also started to leverage user-generated reviews as a way to recommend and filter products and services. For example, Acair et al. look at the use of user-generated movie reviews from IMDb as part of a movie recommender system [1] and similar ideas are discussed in [18].

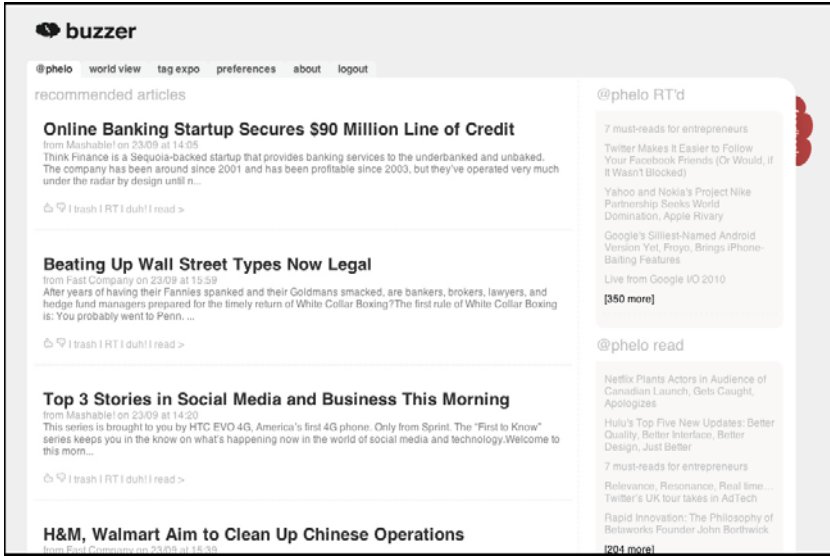


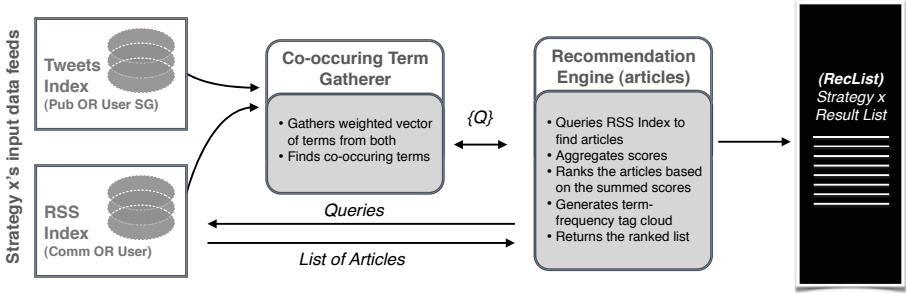
Fig. 1. A screenshot of Buzzer, with personalized news results RT'd for a given user

Both of these instances of related work look to mine review content as an additional source of recommendation knowledge (in a similar way to the content-boosted collaborative filtering technique in Melville et al. [13]), but they rely on the availability of detailed item reviews, which may run to hundreds of words but which may not always be available. In this paper, we consider trending and emerging topics on user-generated content sites like twitter as a way to automatically derive recommendation data for topical news and web-item discovery.

### 3 The Buzzer System

People talk about news and events on Twitter all of the time. They share web pages about news stories. They express their views on recent stories. They even report on emerging news stories as they happen. Surely then it is logical to think of Twitter as a source of news information and news preferences? The challenge of course is that Twitter is borderline chaotic: tweets are little more than impressions of information through fleeting moments of time. Can we really hope to make sense of this signal and noise and harness the chaos as a way to search, filter and rank news stories? This is the objective of the research presented in this paper. Specifically, we aim to mine Twitter information, from both public data streams, and the streams of related users, as a way to identify discriminating terms that are capable of being used to highlight breaking and interesting news stories.

As such the *Buzzer* system adopts a content-based technique to recommending news articles, but instead of using structured user profiles we use unstructured real-time feeds from Twitter. In effect, the user messages (*tweets*) themselves act



**Fig. 2.** Generating results for a given strategy. System mines a specified RSS and Twitter source and uses the co-occurring technique described to generate a set of results, which will be interleaved with other sets to produce the final list shown to users.

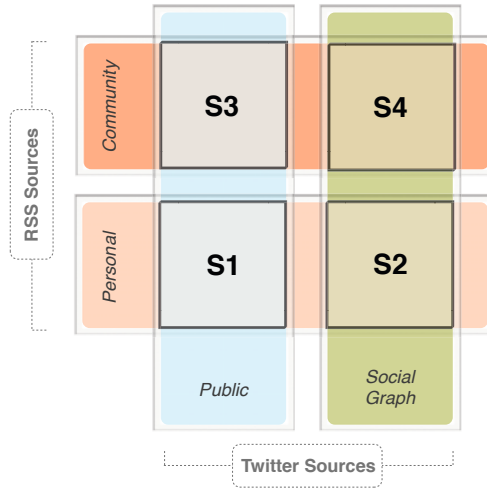
as an implicit ratings system for promoting and filtering content for retrieval in a large space of items of varied topicality or relevance to users.

### 3.1 System Architecture

The high-level Buzzer system architecture is presented in Figure 2. In summary, Buzzer generates two content indexes, one from Twitter (including public tweets and Buzzer-user tweets as discussed below) and one from the RSS feeds of Buzzer users. Buzzer looks for correlations between the terms that are present in tweets and RSS articles and ranks articles accordingly. In this way, articles with content that appear to match the content of recent Twitter chatter (whether public or user related) will receive high scores during recommendation. Figure 3 shows a sample list of recommendations for a particular user. Buzzer itself is developed as a web application and can take the place of a user’s normal RSS reader: the user continues to have access to their favourite RSS feeds but in addition, by syncing Buzzer with their Twitter account, they have the potential to benefit from a more informative ranking of news stories based on their inferred interests.

### 3.2 Strategies

Each Buzzer user brings two types of information to the system — (1) their RSS feeds; (2) their Twitter social graph — and this suggests a number of different ways of combining tweets and RSS during recommendation. In this paper, we explore 4 different news retrieval strategies ( $S1 - S4$ ) as outlined in Figure 3. For example, stories/articles can be mined from a user’s personal RSS feeds or from the RSS feeds of the wider Buzzer community. Moreover, stories can be ranked based on the tweets of the user’s own Twitter social graph, that is the tweets of their friends and followers, or from the tweets of the public Twitter timeline. This gives us 4 different retrieval strategies as follows (as visualized in Figure 3):



**Fig. 3.** Buzzer Strategy Matrix

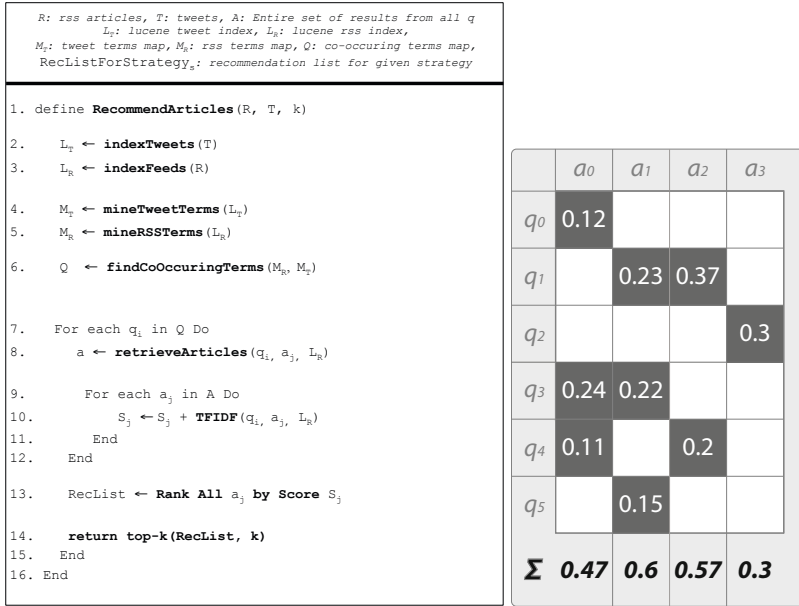
1.  $S1$  — *Public Twitter Feed / Personal RSS Articles*: mine tweets from the public timeline, searches the user’s index of RSS items.
2.  $S2$  — *Friends Twitter Feed / Personal RSS Articles*: mine tweets from people the user follows, searches the user’s index of RSS items.
3.  $S3$  — *Public Twitter Feed / Community Pool of RSS Articles*: mine tweets from the public timeline, searches the entire space of RSS items gathered from all users’s subscriptions.
4.  $S4$  — *Friends Twitter Feed / Community Pool of RSS Articles*: mine tweets from the public timeline, searches the entire space of RSS items gathered from all users’s subscriptions.

In the evaluation section of this paper we will add a 5th strategy as a standard benchmark (ranking stories by recency).

As explained in the pseudo-code in Figure 3(a), the system generates four distinct sets of results based on varied inputs. Given a user,  $u$ , and a set of RSS articles,  $R$ , and a set of Tweets,  $T$ , the system separately indexes both to produce two *Lucene* indexes<sup>1</sup>. The resulting index terms are then extracted from these RSS and Twitter indexes as the basis to produce RSS and Twitter term vectors,  $M_R$  and  $M_T$ , respectively.

We then identify the set of terms,  $Q$ , that co-occur in  $M_T$  and  $M_R$ ; these are the words that are present in the latest tweets and the most recent RSS stories and they provide the basis for our technique. Each term,  $q_i$ , is used as a query against the RSS index to retrieve the set of articles  $A$  that contain  $q_i$  along with their associated TF-IDF (term frequency inverse document frequency) score [16]. Thus each co-occurring term,  $q_i$  is associated with a set of articles  $a_1, \dots, a_n$ , which

<sup>1</sup> Lucene (<http://lucene.apache.org>) is an open-source search API, it proved useful when dealing with efficiently storing these documents and has native TF-IDF support.



(a) Main algorithm for given strategy (b) Hit-matrix-like scoring

Fig. 4.

contain *t*, and the TF-IDF score for *q<sub>i</sub>* in each of *a<sub>1</sub>, ... a<sub>n</sub>* to produce a matrix as shown in Figure 3. To calculate an overall score for each article we simply compute the sum of the TF-IDF scores across all of the terms associated with that article as per Equation 1. In this way, articles which contain many tweet terms with higher TF-IDF scores are preferred to articles that contain fewer tweet terms with lower TF-IDF scores. Finally, retrieving the recommendation list is a simple matter of selecting the top *k* articles with the highest scores. Each time Buzzer mines an individual feed from a source, the articles are copied into both the user’s individual article pool, and a community pool. Each article has a differing relevance score in either pool, as their TF-IDF score changes based on the other content in the local directory with it.

$$Score(a_i) = \sum_{\forall q_i} element(a_i, q_i) \tag{1}$$

For a user, four results-lists are generated, and the fifth recency-based list is gathered by collecting the latest to 2-day old articles (as the update windows on each feed can vary). Items from each of these strategies are interleaved into a single results list for presentation to the user (this technique is dependent on our experimental setup, explained further in the next section). Finally, the user is presented with results and is encouraged to click on each item to navigate to the source website to read the rest of its contents. We capture this click-thru as well as other data such as username, the position in the list the result is, the score



and other data, and consider the act of clicking it as a metric for a successful recommendation.

## 4 User Evaluation

We undertook a live-user evaluation of the Buzzer system, designed to examine the recommendation effectiveness of its constituent social, RTW, retrieval and filtering techniques ( $S1 - S4$ ) alongside a more typical recency-based story recommendation strategy ( $S5$ ). Overall our interest is not so much concerned with whether one strategy is superior to others — because in reality we believe that different strategies are likely to have a role to play in news story recommendation — but rather to explore the reaction of users to the combination of strategies.

### 4.1 Evaluation Setup

As part of this evaluation, we re-developed the original Buzzer system [15] with a more comprehensive interface providing users with access to a full range of news consumption features. Individual users were able to easily add their favourite RSS feeds (or pick from a list of feeds provided by other users) and sync up their Twitter accounts, to provide Buzzer with access to their social graph. In addition, at news reading time users could choose to trash, promote, demote, and even re-tweet specific stories. Moreover, users could opt to consume their news stories from the Buzzer web site and/or sign up to a daily email digest of stories. In this evaluation we focus on the reaction of users to the daily digest of email stories since it provides us with a consistent and reasonably reliable (once-per-day) view of news consumption.

This version of Buzzer was configured to generate news-lists based on a combination of the 5 different recommendation strategies:  $S1 - S4$ , and  $S5$ , as described in Section 3. Each daily email digest contained 25 stories in 5 blocks of 5 stories each. Each block of 5 stories was made up of a random order of one story from each of  $S1 - S5$ ; this the first block of 5 stories contained the top-place recommendations from  $S1 - S5$ , in a random order, the second block contained the second-place stories from  $S1 - S5$ , in a random order, and so on. We did this to prevent any positional bias, whereby stories from one strategy might always appear ahead of, or below, some other strategy. Thus every email digest contained a mixture of news stories as summarized in Section 3.

The evaluation itself consisted of 35 active users; these were users who had registered with Buzzer, signed up to the email digest, and interacted with the system on at least two occasions. The results presented relate to usage data gathered during the 31 days of March 2010. During this timeframe we gathered a total of 56 million public tweets (for use in strategies  $S1$  and  $S3$ ) and 537,307 tweets from the social graphs of the 35 registered users (for use in  $S2$  and  $S4$ ).

In addition, the 35 users registered a total of 281 unique RSS feeds as story sources and during the evaluation period these feeds generated a total of 31,137 unique stories/articles. During the evaluation, Buzzer issued 1,085 emails. We considered participants were fairly active Twitter users, with 145 friends, 196 followers and 1241 tweets sent, on average.

## 4.2 Results

As mentioned above, our primary interest in this evaluation is understanding the response profile of participants across the different recommendation strategies. To begin with, Figure 5(a) presents the total per-strategy click-thrus received for stories across the 31 days of email digests, across the participants. It is interesting to note that, as predicted all of the strategies do receive click-thrus for their recommendations, as expected.

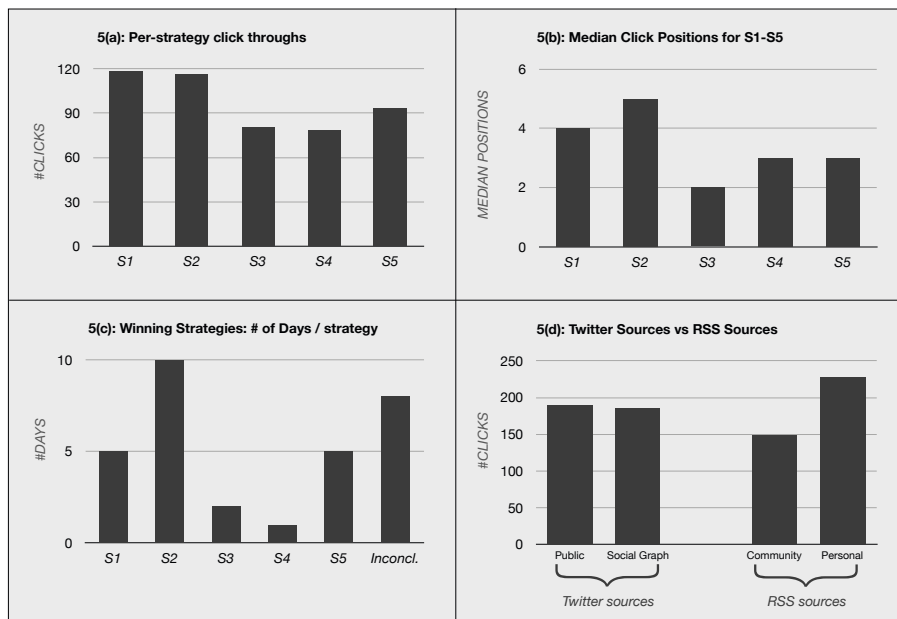
Overall, we can see that strategies *S1* and *S2* tend to outperform the other strategies; for example, *S1* and *S2* received about 110 click-thrus each, just over 35% more than strategies *S3* and *S4*, and about about 20% more than the default recency strategy, *S5*. Strategies *S1*, *S2*, and *S5* retrieve stories from the user's own registered RSS feeds, and so there is a clear preference among the users for stories from these sources. However, stories from these feeds that are retrieved based on real-time web activity (*S1* and *S2*) attract more click-thrus than when these stories are retrieved based on recency (*S5*). Clearly users are benefiting from the presentation of more relevant stories due to *S1* and *S2*.

Moreover it is interesting to note that there is little difference between the relevance of stories (as measured by click-thru) ranked by the users own social graph (*S2*) compared to those ranked by the Twitter public at large (*S1*). Of course both of these strategies mine the user's own RSS feeds to begin with and so there is an assumed relevance in relation to these stories, but clearly there is some value, for the end user, in receiving stories ranked by their friends' activities and by the activities of the wider public. Participants responded less frequently to stories ranked highly by strategies *S3* and *S4*, although it must be said that these strategies still manage to attract about 30% of total click-thrus.

This is perhaps to be expected; for a start, both of these strategies sourced their result lists from RSS feeds that were not part of the user's regular RSS-list; a typical user registered 15 or so RSS feeds as part of their Buzzer sign-up and the stories ranked by *S3* and *S4*, for a given user, came from the 250+ other feeds contributed by the community. By definition then these feeds are likely to be of lesser relevance to a given user (otherwise, presumably, they would have formed part of their RSS submission). Nevertheless, users did regularly respond favourably to recommendations derived from these RSS feeds.

We see little difference between the ranking strategies with only fractionally more click-thrus associated with stories ranked by the public tweets than for stories ranked by the tweets of the user's own social graph.

It is also useful to consider the median position of click-thrus in the result-lists across the different strategies. Figure 5(b) shows this data for each strategy, calculated across emails when there is at least one click-thru for the strategy in question. We see, for example, that the median click-thru position for *S1* is 4 and *S2* is 5, compared to 2 and 3 for *S3* and *S4*, respectively, and compared to 3 for *S5*. On the face of it strategies *S3* and *S4* seem to attract click-thrus for items positioned higher in the result lists. However, this could also be explained



**Fig. 5.** Main Results

by the fact that the high click-thru rates for  $S1, S2, S5$  mean that more items are selected per recommendation list, on average, and these additional items will have higher positions by definition.

Figure 5(c) shows depicts the *winning* strategy  $S_i$  on a given day  $d_j$  if  $S_i$  receives more click-thrus than any other strategy during  $d_j$ , across the 31 days of the evaluation. We can see that strategy  $S2$  (user’s personal RSS feeds ranked by the tweets of their social graph) wins out overall, dominating the click-thrus of 10 out of the 31 days. Recency ( $S5$ ) comes a close second (winning on 8 of the days). Overall strategies  $S3$  and  $S4$  do less well here, collectively winning on only 3 of 31 days.

## 5 Discussion and Conclusion

These results support the idea that each of the 5 recommendation strategies has a useful role to play in helping users to consume relevant and interesting news stories. Clearly there is an important opportunity to add value to the default recency-based strategy that is epitomized by  $S5$ . The core contribution of this work is to explore whether Twitter can be used as a useful recommendation signal and strategies  $S1 - S4$  suggest that this is indeed the case. It was not our expectation that any single strategy would win outright, mostly because each strategy focuses on the recommendation of different types of news stories, for different reasons, and for a typical user, we broadly expected that they would benefit from the combination of these strategies.

In Figure 5(d), we summarize the click-thru data according to the framework presented in Figure 3 by summing click-thru data across the diagram's rows and columns in order to describe aggregate click-thrus for different classes of recommendation strategies. For example, we can look at the impact of different Twitter sources (public vs. the user's social graph) for ranking stories. Filtering by the Twitter's public timeline ( $S1 + S3$ ) delivers a similar number of click-thrus (about 185) as when we filter by the user's social graph ( $S2 + S4$ ), and so we can conclude that both approaches to retrieval and ranking have considerable value. Separately, we can see that drawing stories from the larger community of RSS feeds ( $S3 + S4$ ) attracts fewer click-thrus (approximately 150) than stories that are drawn from the user's personal RSS feeds (strategies  $S1 + S2$ ), which attract about 225 click-thrus, which is acceptable and expected. These community strategies do highlight the opportunity for the user to engage and discover interesting and relevant content they potentially wouldn't have been exposed to from their own subscriptions.

We have explored a variety of different strategies by harnessing Twitter's public tweets, as well as the tweets from a user's social graph, for the filtering of stories from both personal and community RSS indexes. The results of the live-user evaluation are positive. They demonstrate how different recommendation strategies benefit users in different ways and overall we see that most users respond to recommendations that are derived from a variety of different strategies. Overall users are more responsive to stories that come from their favourite RSS feeds, whether ranked by public tweets or the tweets of their social graph, than stories that are derived from a wider community repository of RSS stories.

There are many opportunities for further work within the scope of this research. Some suggestions include considering preference rankings and click-thrus as part of the recommendation algorithm. Also, it will be interesting to consider whether the *reputation* of users on Twitter has a bearing on how useful their tweets are during ranking. Moreover, there are many opportunities to consider more sophisticated filtering and ranking techniques above and beyond the TF-IDF based approach adopted here. Finally, there are many other application domains that may also benefit from this approach to recommendation: product reviews and special offers, travel deals, URL recommendation, etc.

## References

1. Aciar, S., Zhang, D., Simoff, S., Debenham, J.: Recommender system based on consumer product reviews. In: WI 2006: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, Washington, DC, USA, pp. 719–723. IEEE Computer Society, Los Alamitos (2006)
2. Balabanovic, M., Shoham, Y.: Combining content-based and collaborative recommendation. *Communications of the ACM* 40, 66–72 (1997)
3. Brusilovsky, P., Henze, N.: Open corpus adaptive educational hypermedia. In: Brusilovsky, P., Kobsa, A., Nejdil, W. (eds.) *Adaptive Web 2007*. LNCS, vol. 4321, pp. 671–696. Springer, Heidelberg (2007)

4. Cantador, I., Bellogín, A., Castells, P.: News@hand: A Semantic Web Approach to Recommending News. In: Nejdl, W., Kay, J., Pu, P., Herder, E. (eds.) AH 2008. LNCS, vol. 5149, pp. 279–283. Springer, Heidelberg (2008)
5. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In: WWW 2007, pp. 271–280. ACM, New York (2007)
6. Esparza, S.G., O’Mahony, M.P., Smyth, B.: On the real-time web as a source of recommendation knowledge. In: RecSys 2010, Barcelona, Spain, September 26-30. ACM, New York (2010)
7. Huberman, B.A., Romero, D.M., Wu, F.: Social networks that matter: Twitter under the microscope. SSRN eLibrary (2008)
8. Java, A., Song, X., Finin, T., Tseng, B.: Why we twitter: understanding microblogging usage and communities. In: Proceedings of the Joint 9th WEBKDD and 1st SNA-KDD Workshop, pp. 56–65 (2007)
9. Kamba, T., Bharat, K., Albers, M.C.: The krakatoa chronicle - an interactive, personalized, newspaper on the web. In: Proceedings of the Fourth International World Wide Web Conference, pp. 159–170 (1995)
10. Krishnamurthy, B., Gill, P., Arlitt, M.: A few chirps about twitter. In: WOSP 2008: Proceedings of the First Workshop on Online Social Networks, pp. 19–24. ACM, NY (2008)
11. Kwak, H., Lee, C., Park, H., Moon, S.: What is twitter, a social network or a news media? In: WWW 2010, pp. 591–600 (2010)
12. Lerman, K.: Social Networks and Social Information Filtering on Digg. Arxiv preprint cs.HC/0612046 (2006)
13. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Eighteenth National Conference on Artificial Intelligence, pp. 187–192 (2002)
14. Pazzani, M., Billsus, D.: Content-based recommendation systems. In: The Adaptive Web, pp. 325–341 (2007)
15. Phelan, O., McCarthy, K., Smyth, B.: Using twitter to recommend real-time topical news. In: RecSys 2009: Proceedings of the Third ACM Conference on Recommender Systems, pp. 385–388. ACM, New York (2009)
16. Sebastiani, F.: Machine learning in automated text categorization. *ACM Comput. Surv.* 34, 1–47 (2002)
17. Sen, S., Vig, J., Riedl, J.: Tagommenders: connecting users to items through tags. In: WWW 2009: Proceedings of the 18th International Conference on World Wide Web, pp. 671–680. ACM, New York (2009)
18. Wietsma, R.T.A., Ricci, F.: Product reviews in mobile decision aid systems. In: PERMID, Munich, Germany, pp. 15–18 (2005)

# Topical and Structural Linkage in Wikipedia

Kelly Y. Itakura<sup>1,2</sup>, Charles L.A. Clarke<sup>1</sup>, Shlomo Geva<sup>2</sup>, Andrew Trotman<sup>3</sup>,  
and Wei Chi Huang<sup>2</sup>

<sup>1</sup> School of Computer Science, University of Waterloo, Canada

<sup>2</sup> School of Information Technology, Queensland University of Technology, Australia

<sup>3</sup> Department of Computer Science, University of Otago, New Zealand

**Abstract.** We explore statistical properties of links within Wikipedia. We demonstrate that a simple algorithm can predict many of the links that would normally be added to a new article, without considering the topic of the article itself. We then explore a variant of topic-oriented PageRank, which can effectively identify topical links within existing articles, when compared with manual judgments of their topical relevance. Based on these results, we suggest that linkages within Wikipedia arise from a combination of structural requirements and topical relationships.

## 1 Introduction

The internal link structure of Wikipedia differs substantially from the structure of the general Web. Understanding this structure may afford insights to the editors who develop its content, and may also assist the growing league of researchers employing Wikipedia as a convenient and general source of machine-usable knowledge regarding human language, society, history, science, and other subjects. In addition, we study links in Wikipedia with the aim of automatically suggesting out-going links from new articles, and new links for existing articles. We derive our inspiration from the INEX Link-the-Wiki track [4,5], which includes a task to restore links to a set of Wikipedia articles stripped of these links.

At INEX 2007, task participants returned ranked lists of suggested links for each stripped article. By treating the original links appearing in the articles as ground truth, precision and recall measures were applied to evaluate the results. For the INEX 2007 task, we implemented a simple statistical approach that substantially outperformed other approaches [6]. We call this approach the *structural threshold algorithm*, or just the *ST algorithm*. We were surprised by the effectiveness of the ST algorithm because it does not consider the topic of the article forming the source of the link, but only the anchor phrase and the target of the link. At INEX 2008, the ST algorithm was independently implemented by multiple participants, successfully demonstrating that the algorithm effectively recovers many of the links in the original articles.

Along with an evaluation against Wikipedia ground truth, the INEX 2008 evaluation process included separate manual judgments, in which assessors identified links that they believed were topically relevant to the articles. When runs

were measured against these manual judgments, it transpired that both the submitted runs and the original Wikipedia ground truth differed substantially from these manual judgments. Based on this experience, we hypothesize that many links in Wikipedia are primarily *structural* — that many anchor phrases are frequently linked to associated articles regardless of the topic of the linking article. On the other hand, some links are clearly *topical* in nature, created to express a specific relationship between a pair of articles.

## 2 Related Work

A small body of prior work has addressed the problem of link discovery in Wikipedia. This work often employs statistical similarity measures [8,9] or co-citations [1] to compute the probability that a phrase constitutes an anchor. Similar to the work of Mihalcea and Csomai [8] and the work of Milne and Witten [9], our ST algorithm depends on simple statistics. Like the work of Mihalcea and Csomai, the ST algorithm ranks anchor phrases by link probability, but unlike that work, the ST algorithm considers more than just article titles as potential anchor phrases. On the other hand, Milne and Witten, build on top of an ST-like algorithm by using the textual context of an article to disambiguate anchor phrases and destination pages. Gardner and Xiong [3] employ a sequence labeling approach, in which a label indicates the start and part of the links. They use a conditional random field as a classifier, training on both local and global data. Their results are comparable to those of Milne and Witten.

Mihalcea and Csomai, Milne and Witten, and Gardner and Xiong all treat existing links in Wikipedia as manually-crafted *ground truth*. The ST algorithm's performance against this ground truth is better than that of Mihalcea and Csomai and somewhat worse than the more complicated approach of Milne and Witten. However, by applying a variant of PageRank, we demonstrate that existing Wikipedia links should not necessarily be regarded as a gold standard. Instead, topical and structural linkages should be considered separately; otherwise, structural linkages tend to dominate the evaluation.

## 3 Structural Threshold Algorithm

To suggest links for a source article, the structural threshold (ST) algorithm first computes link probability estimates for potential anchor phrases and then applies a cutoff based on anchor density. In applying the ST algorithm, we imagine that this source article was newly added to Wikipedia. In reality, for the purposes of our experiments, the source article was removed from our Wikipedia corpus and stripped of links. For all of our experiments, we work with the Wikipedia corpus used at INEX 2008.

### 3.1 Link Probability Estimate

The ST algorithm first creates a list of all potential anchor phrases by considering all phrases appearing the source article up to some fixed length, after whitespace

normalization. For each potential anchor phrase  $a$ , we assign the destinations  $d$  in order of their frequency in Wikipedia. For each most frequent destination  $d$ , we compute the probability estimate  $\gamma$  that a phrase will be linked as an anchor as follows:

$$\gamma = \frac{\# \text{ of pages containing a link from anchor } a \text{ to a destination page } d}{\# \text{ of pages in which } a \text{ appears at least once}} .$$

Links are suggested by the ST algorithm in order of decreasing  $\gamma$  values. The ST algorithm essentially creates a ranked list of possible links from an article. Since Wikipedia articles do not link to all possible destinations, the actual list of links to add to an article might be determined by setting a threshold for  $\gamma$  based on article length, as we discuss next.

### 3.2 Anchor Density

To select a threshold for adding links to an page we consider the overall density of anchors in an average article. We define an *anchor density*  $\delta$  of a page  $p$  as follows.

$$\delta_p = \frac{\# \text{ of article linked from page } p}{\text{size of page } p \text{ without tags in KB}} .$$

The average anchor density of a corpus with  $N$  pages is, therefore,

$$\delta = \sum_p \frac{\delta_p}{N} .$$

We estimated overall anchor density for the corpus by allocating articles into 5KB bins, according to their size, and then computing the average for each of the bins. The density is roughly linear, with approximately 7.09 anchors for every KB [6]. Similar observations have been made by Zhang and Kamps [11]. Thus, instead of selecting links according to a  $\gamma$  threshold, we may add links to an article in  $\gamma$  order, until an anchor density of  $\delta = 7.09/KB$  is reached.

### 3.3 Performance

INEX 2007 participants generated ranked lists of possible links for each article, which were evaluated against Wikipedia ground truth using standard recall and precision measures. Table 1 compares the performance of the ST algorithm against the best performance achieved by other approaches. As a result of this success at INEX 2007, several other INEX 2008 participants incorporated the ST algorithm into their efforts.

To provide another performance comparison, we incorporated anchor density into an INEX 2008 run that used the ST algorithm, cutting the list of anchor phrases ranked by  $\gamma$  for each topic by  $\delta$  times the article's file size. We evaluated the effect of  $\delta$  by computing a set precision and recall against the Wikipedia ground truth. We obtained the precision of 62.88% and recall of 65.21%. Michalcea and Cosmai report results of 53.37% precision and 55.90% recall on an



**Table 1.** Performance of the structural threshold algorithm at INEX 2007

	MAP	rPrec	P@5	P@10	P@20
ST algorithm	0.61	0.63	0.85	0.82	0.75
Second-best run	0.32	0.42	0.77	0.68	0.58

equivalent task [8]. Milne and Witten, with a more complex anchor disambiguation technique, achieve precision of 74.4% and recall of 73.8% [9]. The simpler ST approach provide reasonable effectiveness despite its lack of any topical considerations.

## 4 Link Analysis

The ST algorithm identifies all anchor phrases with high  $\gamma$  regardless of the topic of an article. Thus, the anchor phrase “Peninsular War”, with  $\gamma = 0.898$ , would be suggested as an anchor for nearly any page in which it occurs, from “Napoleon” to “Otago” to “wine”, regardless of its topical relationship to those pages. According to Wikipedia ground truth, “Peninsular War” should be indeed linked into the article on “Otago”, a city in New Zealand. However, our manual assessors deemed the anchor phrase to be topically non-relevant; from an assessor’s point of view, this European war is not related to New Zealand. On the other hand, the page “Otakou”, from where “Otago” derived its name, appears to be topically relevant, and the manual assessors agree. However, it has a relatively low  $\gamma$  value of 0.63 and the Wikipedia ground truth considers it non-relevant.

### 4.1 KPR Algorithm

We enlist a variant of PageRank to estimate the topicality of anchor phrases. Instead of measuring the popularity of anchor phrases by the  $\gamma$  value, this variant balances both topicality and popularity by computing the contribution to KL-divergence of scores between a standard PageRank (PR) and a topic-oriented PageRank (TPR). Details of this algorithm, which we call the *KPR Algorithm*, are given as an example in Büttcher et al. [2], pages 526–528].

For the work reported in this paper, we use a process that computes KPR with all links present — no articles or links are removed. Our goal is to show that even though there are both topical and structural links in Wikipedia, the topical links indicated by KPR provide a better match to the manual judgments. For INEX 2009 (in work not reported here) we applied KPR to solve the link discovery problem [7]. In those experiments, we used a stripped Wikipedia corpus, employed the ST algorithm to provide initial linkages for the source article, and then applied KPR to compute the final linkages.

Table 2 lists the top-10 results ordered by KPR values for the source article “Otago”. PR, TPR, and  $\gamma$  values are included for comparison. Generally, the results appear to be closely related to the topic of “Otago”.

**Table 2.** Top 10 results for “Otago” ranked by KPR compared with manual relevance values

Article	Relevance	KPR	TPR	PR	$\gamma$
Dunedin	1	0.0171	4302	1.28	0.61
Central Otago Gold Rush	1	0.0166	3278	1.49	0.83
South Otago	1	0.0165	2962	0.62	0.44
New Zealand	1	0.0156	7524	321.33	0.68
Otakou	1	0.0151	2688	0.52	0.63
Balclutha, New Zealand	1	0.0151	279	0.77	0.55
Gabriel Read	1	0.0149	2643	0.52	0.71
Invercargill	0	0.0147	3299	3.84	0.80
South Island	1	0.0146	4259	23.37	0.79
Queenstown, New Zealand	1	0.0144	3007	2.12	0.78

**Table 3.** Wikipedia ground truth ranked by ST values ( $\gamma$ ) and KPR values as measured against manual assessments

	P@5	P@10	P@20
ST algorithm ( $\gamma$ )	54.80	56.20	51.03
KPR algorithm	66.40	63.40	59.93
$\Delta$	11.60	7.20	8.90
<i>p</i> -value	0.0007	0.0337	0.0003

## 4.2 Performance

The aim of the KPR algorithm is to identify articles that are topically related to a given source article. Since the assessors for INEX 2008 judged an article relevant only when it was topically related to a source article, we would expect that ranking links by the KPR algorithm would produce results closer to the manual judgments than ranking links by the ST algorithm. To test this hypothesis, we ranked INEX 2008 Wikipedia ground truth using both the KPR algorithm and the ST algorithm.

For each article in the INEX 2008 test set, we extracted its original links and applied both algorithms to these links. We computed P@5, P@10, and P@20 for each ranking against the manual assessments. The results are shown in Table 3. Even though these articles contain both topical links with high KPR values and structural links with high  $\gamma$  values, the manual assessors prefer those with high KPR values.

If we view the manual assessments as the gold standard, our INEX experience indicates that comparisons against Wikipedia ground truth is not an ideal method for assessing the performance of link suggestion algorithms. Moreover, manual assessments are not easily scalable to experiments with thousands of topics. However, our experience with the KPR algorithm indicates that KPR is a reasonable indicator of topical relevance.

These observations lead to the idea of applying the KPR algorithm to automatically construct a topically oriented assessment set. To create this set, we

first applied the KPR algorithm to articles from the INEX 2008 task. For each article, we took the top-10 links by KPR values and labeled them as topically relevant, as if they were judged that way by a human assessor. All other links were considered to be not topically relevant.

We use Kendall's  $\tau$  to compare the rankings under the two assessment sets. While the resulting value of  $\tau = 0.7354$  falls short of the level that might allow us to replace the manual assessments with automatic assessments [10], it suggests a close relationship between the two sets. Given the simplicity of this experiment, where the top-10 links are simply assumed to be relevant, additional development of this approach may produce a stronger correspondence.

## 5 Concluding Discussion

Links form the glue that binds Wikipedia together. As we demonstrate in this paper, these links appear to arise from a combination of structural requirements and topical relationships. Even in the absence of topical information, we can make reasonable predictions about the links appearing in an article. However, in order to predict manual assessments of topical relevance, the interconnections between articles must be considered.

## References

1. Adafre, S.F., de Rijke, M.: Discovering missing links in Wikipedia. In: 3rd International Workshop on Link Discovery, Chicago, pp. 90–97 (2005)
2. Büttcher, S., Clarke, C.L.A., Cormack, G.V.: Information Retrieval: Implementing and Evaluating Search Engines. MIT Press, Cambridge (2010)
3. Gardner, J.J., Xiong, L.: Automatic link detection: A sequence labeling approach. In: 18th CIKM, Hong Kong, pp. 1701–1704 (2009)
4. Huang, D.W.C., Xu, Y., Trotman, A., Geva, S.: Overview of INEX 2007 link the wiki track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) INEX 2007. LNCS, vol. 4862, pp. 373–387. Springer, Heidelberg (2008)
5. Huang, W.C., Geva, S., Trotman, A.: Overview of the INEX 2008 link the wiki track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2008. LNCS, vol. 5631, pp. 314–325. Springer, Heidelberg (2009)
6. Itakura, K.Y., Clarke, C.L.A.: University of waterloo at INEX2007: Adhoc and link-the-wiki tracks. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) INEX 2007. LNCS, vol. 4862, pp. 417–425. Springer, Heidelberg (2008)
7. Itakura, K.Y., Clarke, C.L.A.: University of waterloo at INEX 2009: Ad hoc, book, entity ranking, and link-the-wiki tracks. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 331–341. Springer, Heidelberg (2010)
8. Mihalcea, R., Csomai, A.: Wikify!: Linking documents to encyclopedic knowledge. In: 16th CIKM, Lisbon, pp. 233–242 (2007)
9. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: 17th CIKM, pp. 509–518. Napa Valley, California (2008)
10. Voorhees, E.M.: Variations in relevance judgments and the measurement of retrieval effectiveness. In: 21st SIGIR, pp. 315–323 (1998)
11. Zhang, J., Kamps, J.: Link detection in XML documents: What about repeated links. In: SIGIR 2008 Workshop on Focused Retrieval, pp. 59–66 (2008)

# An Analysis of Time-Instability in Web Search Results

Jinyoung Kim<sup>1</sup> and Vitor R. Carvalho<sup>2</sup>

<sup>1</sup> Department of Computer Science  
University of Massachusetts Amherst\*

`jykim@cs.umass.edu`

<sup>2</sup> Microsoft Bing

`vitor@cs.cmu.edu`

**Abstract.** Due to the dynamic nature of web and the complex architectures of modern commercial search engines, top results in major search engines can change dramatically over time. Our experimental data shows that, for all three major search engines (Google, Bing and Yahoo!), approximately 90% of queries have their top 10 results altered within a period of ten days. Although this instability is expected in some situations such as in news-related queries, it is problematic in general because it can dramatically affect retrieval performance measurements and negatively affect users' perception of search quality (for instance, when users cannot re-find a previously found document).

In this work we present the first large scale study on the degree and nature of these changes. We introduce several types of query instability, and several metrics to quantify it. We then present a quantitative analysis using 12,600 queries collected from a commercial web search engine over several weeks. Our analysis shows that the results from all major search engines have similar levels of instability, and that many of these changes are temporary. We also identified classes of queries with clearly different instability profiles — for instance, navigational queries are considerably more stable than non-navigational, while longer queries are significantly less stable than shorter ones.

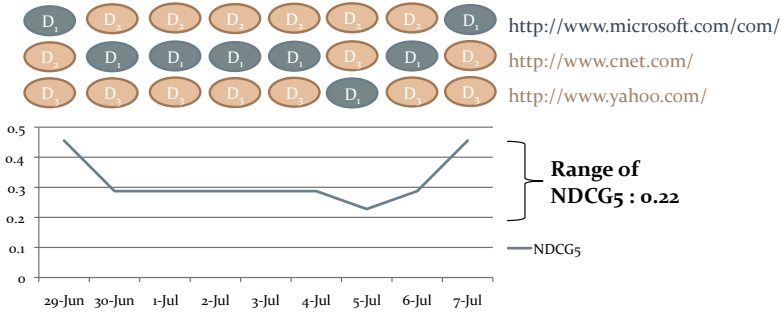
## 1 Introduction

It is natural for web search results to change. Web documents are frequently updated by users as well as publishers. Search engines try to capture as much of these changes as possible and experiment with new retrieval techniques continually. The problem is that, as noted by Selbert and Etzioni [4], the amount of changes in web search results far exceeds the variation of the web itself. That is, for the same query, the top search documents rankings fluctuate even without any real content change in top documents.

As an example, we show daily rank changes of top 3 documents for query 'com' in a major search engine. Figure 1 shows that the relevant document (about the

---

\* This work was done during the author's internship at Microsoft Bing.



**Fig. 1.** Daily change of top 3 documents for query ‘com’ and corresponding fluctuation in NDCG<sub>5</sub>

Microsoft COM object model) changed its position 4 times within 9 days of measurement, resulting in the fluctuation of 22 percentual points in NDCG<sub>5</sub>. This is a typical example of instability – a change in top results that lasted only for a few days. In fact, this issue of instability is quite common and is not confined to a single search engine. Our analysis in Section 5.1 shows overall instability trends for all three major search engines, which suggest that nearly 90% of general web queries experience some change at top 10 results within 10 days.

This instability causes serious issues for both end users and search engine companies. For end users, this can be a source of confusion and frustration, especially when they want to get back to earlier results. Teevan et al. [5] suggests that 40% of web queries are re-finding queries, and that rank changes make it hard for users to get back to previously clicked results. For search engine companies, which monitor search quality all the time, this variation in search results mean unstable performance measurement, making them difficult to track down performance precisely and make important decisions (e.g., shipping of new ranking algorithm).

In this paper, we aim to analyze this phenomenon of time-instability in web search results. By instability, we mean the day-to-day change in top 10 results of the same query over a period of time. Our notion of change includes the change in the position of existing documents as well as the insertions (i.e., new documents added to the top 10 list) and deletions (i.e., a document was removed from the top 10 list) of documents. We focus on this short-term change in top rank list because of its high visibility to users.

To better understand the problem, we conceptualize different aspects of instability, based on its relation to structural change of search engine, its source and its duration. We also suggest a set of metrics for quantifying the degree of instability. Then we present an analysis based on large-scale data (12,600 queries), which shows that top results from web search engines experience a considerable amount of changes, and that many of these changes are temporary. We also found that many characteristics of a query – length, frequency and intent – affect its degree of instability.

The rest of this paper is organized as follows. In the next section, we provide an overview of related work. We then introduce several types of instability in Section 3, followed by a description of the data set and metrics we used for measuring instability in Section 4. Results from our data analysis are presented in Section 5. We then conclude this work in Section 6.

## 2 Related Work

Several previous studies focused on the changes in the contents of webpages, instead of chronological changes in retrieval ranks of the same documents. For instance, Fetterly et al. [2] crawled 150 million pages weekly for 3 months and found that 65% of pages remain the same during the period. Ntoulas et al. [3] performed similar studies and showed that major source of change is the creation of new pages as opposed to the update in existing pages. Adar et al. [1] crawled 55,000 webpages hourly for 5 weeks and found that 34% of pages did not have any change during the period. They attributed the differences in document sampling method to much higher frequency of change. In contrast, our work focus on the changes in the rankings of search engine results as opposed to the changes in document contents.

Teevan et al. [5] studied the re-finding behavior of web search users and showed that as much as 40% of all queries are re-finding queries. They also concluded that rank change has detrimental impact on this type of task, finding that users are less likely to re-click the results and take more time to do so when a previously clicked result changed its position. Building on their insights, our work presents the first quantification and comparison of search results time instability in major commercial search engines in large scale, as well as a detailed analysis on various aspects of this instability.

Selberg et al. [4] analyzed the instability of web search results more than ten years ago. They showed that 54% of top 10 documents are replaced over a month, and that web search results change much faster than the web itself. They also suggested the caching of results to improve the response time as a possible cause of this instability. Our study is different in that we used large-scale data which include ranking scores and relevance judgments. Also, while they focused on changes as the set overlap in results between two time periods, our analysis include many varieties of instability as well as the correlating factors.

## 3 Types of Instability

### 3.1 Structural vs. Non-Structural

Modern commercial search engines are permanently experimenting with new ideas. Often search engines can introduce new features or ranking/indexing techniques that can cause sweeping changes to the rankings for a large number of queries. We refer to the instability derived from such expected and controlled

events as structural instability. In contrast, non-structural instability is the one found even when there is no such events. In this work, we focus on non-structural instability.

### 3.2 Indexing Issues vs. Ranking Issues

Another distinction we can make about instability concerns the source it is stemming from, which can be grouped into indexing issues and ranking issues. Indexing issues are related to the availability of a document in the search engine index. That is, if a document is not found in the index, there is no chance it will be ranked among the top 10 documents, even if a few moments earlier the document was readily available. Various factors can be behind this type of instability, including, but not limited to, different (re)crawling policies, spam detection policies, click fraud policies, architecture and capacity limitations, to name a few. From a top 10 results page perspective, the common characteristic of all indexing issues is that they manifest as insertions or deletions of documents at top 10 list, leaving no impact on the relative ranking of the rest of documents.

Ranking issues include fluctuation in the value of ranking features, which can be caused by document content changes, link structure changes, anchor text changes, among many other factors. In contrast to indexing issues, ranking issues cause relative positions of existing documents to change, leading to rank swaps among existing documents. Note that, in our definition, swaps are defined in terms of two documents. The query in Figure 11 exemplifies this kind of change, showing 4 swaps during 9 days of measurement.

### 3.3 Short-Term vs. Long-Term

As previously mentioned, changes in search results are somewhat expected, and our notion of instability includes only such fluctuations which last only for a few days. In this sense, it is meaningful to classify the change (insertion, deletion or swap) with regard to its duration. For insertions, the duration measures how many days inserted documents stay at top 10 results. Similarly, the duration of a swap measures the period during which two documents keep their relative positions after a swap has happened. In this paper we arbitrarily define short-term changes as the ones that are revoked within 5 days, whereas any change that lasts longer than 5 days is considered as long-term.

## 4 Measuring Instability

### 4.1 Data Set

We collected daily snapshots of the top 10 results for the same set of 12,600 random queries over a period of four weeks between June and July of 2010. These queries were randomly selected from the Microsoft Bing<sup>1</sup> search engine query logs. For each day and for each top 10 query-document pairs, we also

---

<sup>1</sup> <http://www.bing.com>

**Table 1.** The statistics of our two data collections

Name	Date	#Queries	Source
Bing Set	6/12 – 7/8	12,600	Bing internal
Big3 Set	8/2 – 8/16	1,000	Bing / Google / Yahoo! API

extracted the values of all its ranking scores. In addition, we also collected the human relevance judgments in a 5-grade scale (Perfect, Excellent, Good, Fair and Bad) for each query-document pair.

During the first two weeks of data collection, Bing deployed structural changes in the ranking/indexing techniques that significantly affected the profile of the collected top 10 query-documents. Yet no such changes happened during the last two weeks of data gathering. While our notion of instability focuses on the variation in results without any structural change, we compare the instability trend between two periods in Section 5.2.

News-related queries are expected to present large instability. Since we wanted to quantify the impact of these queries in our study, we identified from the collection any query that showed a *news* document (a document that was recently created) during the test period. Eventually, this process found 951 queries in which there were at least one insertion of *news* documents during the test period, which is about 7.7% of entire query set. In Section 5.4 and 5.5, we present how the characteristics of news queries are different from regular queries.

In order to compare instability trends across different commercial search engines, we also created a second data collection with the top 10 results for 1,000 queries issued at Google<sup>2</sup> and at Yahoo<sup>3</sup> using their APIs during two weeks in August 2010. These queries were also randomly selected from the Bing query logs. Note that we used the API results for Bing here as well to ensure consistency in measurement. Table 1 summarizes our data set, where we named the collection from Bing as Bing Set, the collection from three major search engines as Big3 Set.

## 4.2 Metrics of Instability

There can be several ways of measuring the instability in web search results during a time period between  $t$  and  $t+n$ . For instance, to measure how many of documents changed over time, we can use the ratio of set overlap between top  $k$  results of two rank lists.

$$\text{Overlap}_k(R_t, R_{t+n}) = \frac{|R_t \cap R_{t+n}|}{k} \quad (1)$$

If we are more concerned about the change in ranking, we can use the ratio of agreement in pairwise preference between two rank lists, which are defined as follows:

<sup>2</sup> <http://www.google.com>

<sup>3</sup> <http://www.yahoo.com>



**Table 2.** Pearson correlation coefficient between the measures of instability

Measure	PairAgree <sub>5</sub>	vNDCG <sub>5</sub>	rNDCG <sub>5</sub>
Overlap <sub>5</sub>	0.986	-0.131	-0.347
PairAgree <sub>5</sub>		-0.139	-0.362
vNDCG <sub>5</sub>			0.798

$$\text{PairAgree}_k(R_t, R_{t+n}) = \frac{|\text{PairPref}(R_t) \cap \text{PairPref}(R_{t+n})|}{k(k-1)/2} \tag{2}$$

Although above measures will summarize the change in the rank list itself, since we are concerned about the change in the quality of ranking in many cases, we can use the range of NDCG<sub>k</sub> ( henceforth rNDCG<sub>k</sub>) for each day during the period of our analysis. Alternatively, we can use the variance in NDCG<sub>k</sub> (vNDCG<sub>k</sub>) instead of the range.

$$\begin{aligned} r\text{NDCG}_k(R_t, R_{t+n}) = & \max(\text{NDCG}_k(R_t), \dots, \text{NDCG}_k(R_{t+n})) \\ & - \min(\text{NDCG}_k(R_t), \dots, \text{NDCG}_k(R_{t+n})) \end{aligned} \tag{3}$$

Table 2 shows the correlation among these measures on the Bing Set. Since the Overlap<sub>5</sub> and PairAgree<sub>5</sub> are defined between two rank lists, we used the data from the first and the last day of measurement. In the end, we decided to use the range of NDCG<sub>5</sub> as main metric since it correlates relatively well with other metrics, and captures what is more likely to be noticed by users: the worst-case scenario of instability in ranking quality.

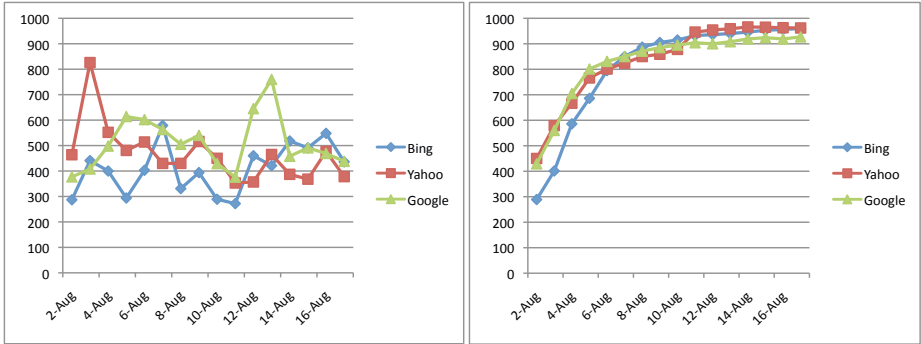
## 5 Instability Trends

### 5.1 Overall Trends

We first present overall level of instability for Big3 Set in Table 3. Although Yahoo! seems to have slightly more unstable results, all three search engines seem to have similar level of instability using the metrics we used. Over the two-week period of our measurement, around 20% of documents were replaced in top 10 results, and around 40% of pairwise preferences were changed. Average range of NDCG<sub>5</sub> was around 6%.

**Table 3.** Overall level of instability for three search engines

Measure	Bing	Google	Yahoo!
Overlap <sub>5</sub>	0.958	0.958	0.948
Overlap <sub>10</sub>	0.819	0.815	0.798
PairAgree <sub>5</sub>	0.911	0.911	0.895
PairAgree <sub>10</sub>	0.641	0.642	0.603
rNDCG <sub>5</sub>	0.057	0.058	0.063



**Fig. 2.** Number of queries with some change in the ranking of top 10 for Big3 Set. The left figure shows a day-to-day trend, and the right figure shows cumulative trend for the same data.

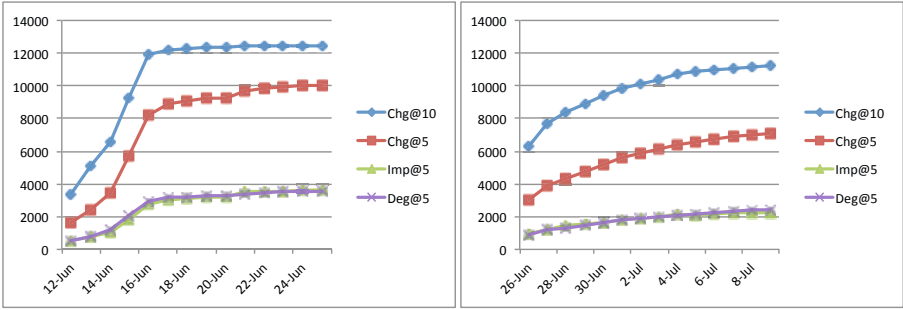
As the daily trend of instability, Figure 2 shows the number of queries (out of the 1,000 sample queries) with change (insertion or swap) in the top 10 results for three major search engines. Here, it is evident that nearly 30% of queries had some change at top 10 every day and that nearly 90% queries experienced change within 10 days.

Another observation from Figure 2 is that the daily amount of change among different search engines does not tend to correlate. (i.e. the day with big change in Google is not necessarily the same in Bing) This provides evidence that these instabilities are mostly due to internal factors specific to each search engine.

We further analyzed the correlation in query-level instability ( $rNDCG_5$ ) between each pair of search engines. The result showed little correlation (less than 0.1 in Pearson correlation coefficient), whereas the correlation between retrieval performance measured in  $NDCG_5$  was very high (around 0.7). In other words, an easy query in one search engine is likely to be easy in another search engine as well, yet a query with unstable result in one place is not necessarily unstable in another. This can be another evidence that instability in web search result is somewhat separate phenomenon from the dynamics of the web itself.

### 5.2 Structural vs. Non-Structural

Here we compare the instability of the search engine when affected by a structural change versus non-structural changes using Bing Set. The graph in Figure 3 shows the number of queries with some change in the ranking of top 10 ( $Chg@10$ ) and top 5 ( $Chg@5$ ), improvement ( $Imp@5$ ) and degradation ( $Deg@5$ ) at  $NDCG_5$  for former (left) and latter (right) 2-week period. It shows that all the queries had some change at top 10 at former period, and around 57% of all queries had changes in  $NDCG_5$  ( $Imp@5 + Deg@5$ ). Comparing the change in ranking and the change in ranking quality top 5, we can see that around 30% of the change in ranking results did not cause any change in  $NDCG_5$ , as exemplified by the query in Figure 1.



**Fig. 3.** The comparison between structural instability and non-structural instability. Each plot shows the cumulative number of queries with some change in the ranking of top 10 and top 5, improvement and degradation at NDCG<sub>5</sub> for Bing Set.

If you recall from Section 4.1 that Bing had a structural change in the former period, this level of change is not surprising. However, even in the latter period where we did not have any such event, we can still find that 90% of queries had some change at top 10, with 37% of queries had changes in NDCG<sub>5</sub>. This clearly indicates that the top search results experience a great deal of instability even without any structural change. Also, here you can see that the number of positive and negative changes are roughly equal, meaning that these changes does not impact the average retrieval effectiveness.

### 5.3 Insertions vs. Swaps

Next we present the amount of changes in the form of insertions and swaps. As mentioned in Section 3.2, an insertion means the addition of new document among the top 10 results, whereas a swap means the change in relative positions of two documents. Note that some of the insertions within top 10 results would be swaps if we consider rank positions beyond top 10.

The distribution of rank positions for insertions (left) and swaps in Figure 4 supports this point, showing a huge increase in insertions at rank position 9 and 10 overlapping with sudden drop at the number of swaps around the same rank positions. Otherwise, it shows that both the number of insertions and swaps steadily decrease as the rank position gets lower. We hypothesized that this relative stability at lower positions might be due to the fact that these documents have much higher scores than other documents, for which we provide some evidence in Section 5.5.

Figure 5 shows the daily trends in insertions and swaps for Bing Set, where it indicates around 5,000 insertions and 12,000 swaps per day in latter two weeks (without structural change in search engine). This means that approximately one document was inserted for every other query, and that one pair of documents swapped positions for every query. We also found that the number of ‘news’ documents inserted was around 600 per day, from which you can see that the majority of top 10 insertions happens from non-news sources.

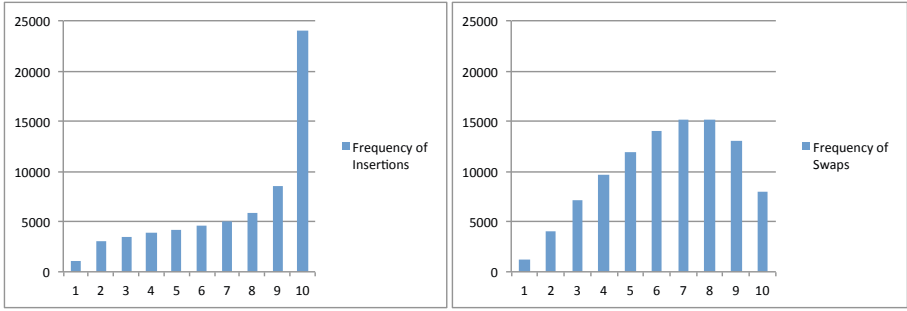


Fig. 4. The distribution of rank positions for insertions (left) and swaps (right)

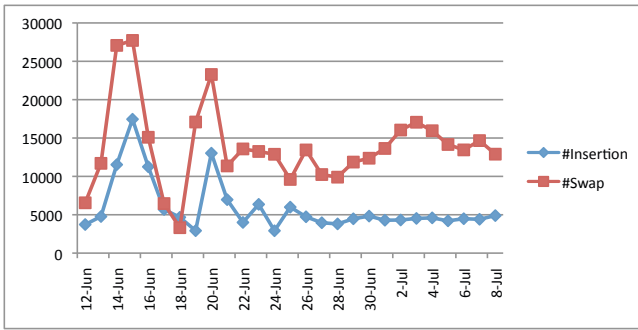
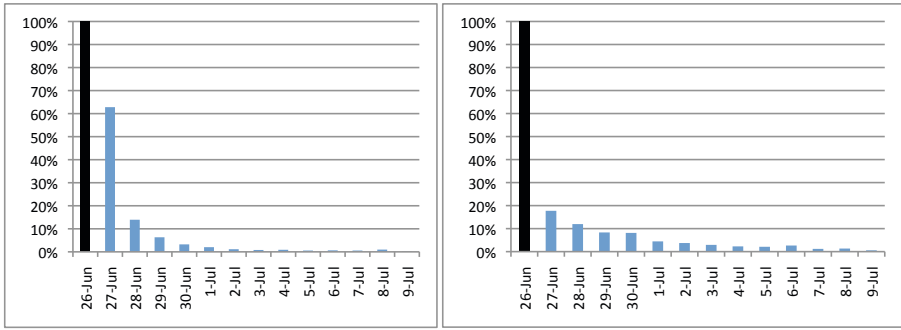


Fig. 5. Daily number of insertions and swaps at top 10

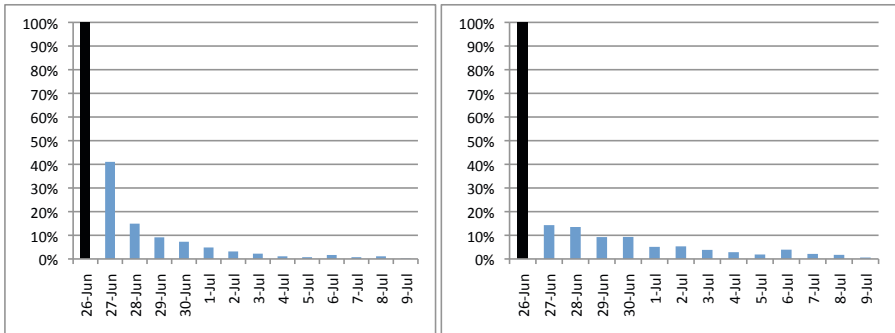
### 5.4 Duration of Change

We then analyzed the duration of change as defined in Section 3.3 using Bing data set. Figure 6 shows the number of insertions at June 26th, and the number of insertions that was revoked in the following days (that is, given all document insertions in the top 10 that happened on June 26th, the percentage of these documents that disappeared from the top 10 in the following days). Here you can see that queries with news intent (951) have larger portion of temporary insertions (90% if we use the threshold of 5 days). However, even for non-news queries (11,650), 50% of inserted documents are taken out of top 10 within 5 days.

Figure 6 shows the number of swaps at June 26th, and the number of swaps that were revoked in the following days. Here you can see that queries with news intent have still larger portion of temporary swaps (70%), yet 50% of rank swaps for regular queries are revoked within 5 days. In overall, we can see that a majority of changes that happens to top results last no longer than a few days, as was the case of the example query in Figure 1.



**Fig. 6.** The percentage of insertions on June 26th that was revoked in the following days for queries with news intent (left) and regular queries (right)



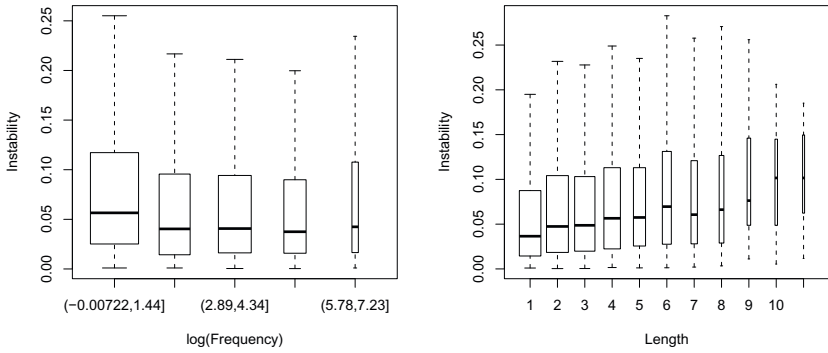
**Fig. 7.** The percentage of swaps on June 26th that was revoked in the following days for queries with news intent (left) and regular queries (right)

### 5.5 Factors that Affect Instability

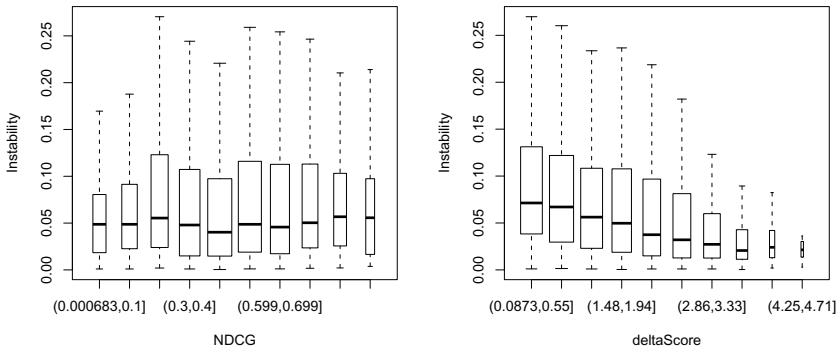
Finally, we investigate a few factors regarding queries that affect instability. We first look at how queries with different frequency and length have different levels of instability. Here, the frequency is calculated from Bing’s query logs, and the length indicates word counts. Note that we use the range of  $NDCG_5$  ( $rNDCG_5$ ) as the metric of instability henceforth.

Figure 8 shows a boxplot with the instability of queries for different frequency ranges (in log) and different lengths. The width of each boxplot is proportionate to the square root of the number of data points in each range. Although instability and query frequency does not seem have straightforward correlation, it is clear that longer queries have higher instability than shorter queries. Given that most of long queries are known to be tail queries, the trends in both plots are consistent.

Next, the left side of Figure 9 shows the degree of instability for different levels of query difficulty, as measured by  $NDCG_5$ . Although there seems to be some indication that the hardest queries as well as the easiest queries have lower



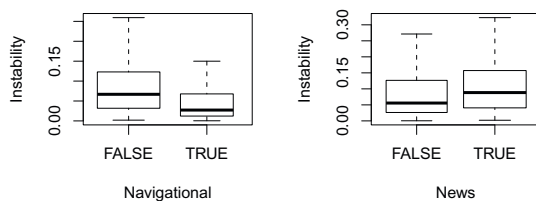
**Fig. 8.** The instability of queries with different log of frequency (left) and length (right)



**Fig. 9.** The instability of queries with varying difficulty (left) and score difference between the 1st and the 5th document (right)

instability ranges, overall query difficulty does not show a clear correlation with time instability. The right side of Figure 9 shows the clear relationship between instability and the difference in ranking scores between the top document and 5th document (deltaScore5). This makes intuitive sense in that queries with smaller differences between document scores are more likely to have rank swaps even at the small change in input feature values. Also, combined with the finding that top positions in a search result have higher differences between documents (plot omitted for space constraint), this explains the result in Section 3.2 that top positions in web search results are usually more stable than others.

We finally looked at the relationship between instability and some query classes. The left plot in Figure 10 shows that navigational queries have much less instability, which is consistent with the trend in Figure 8 since navigational queries are typically shorter and more frequent. The plot on the right confirms that queries with news intent has more changes in the top rank list, which is expected considering that search engines inject lots of news documents into those queries.



**Fig. 10.** The instability with respect to two query classes: Navigational and News

## 6 Conclusions

This paper presented the first large-scale study on the time instability of web search results. We suggested several classes of instability, and the metrics for quantifying it. Our data shows that top results in all major search engines have similar levels of day-to-day fluctuation and that many of these changes are temporary. We investigated several factors that impact time instability of each query, and we found that longer queries are more unstable than their counterparts, and certain classes of queries such as navigational present lower instability than average, while other classes show higher than average instability, such as news queries.

Being the first large-scale analysis of this kind, our study can lead to various directions as future work. While previous studies provide some evidence that the the instability causes problem with re-finding, a more careful user study should be followed to quantify the impact of instability in user's search experience. Also, given the negative impact of instability in user perception and search quality measurement, we need to find out how to control unwanted fluctuations in search results.

## Acknowledgements

The authors would like to thank Rich Caruana, Giridhar Kumaran, Manish Malik and Mohammad Makarechian for their helpful feedback during our project. This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect those of the sponsor.

## References

1. Adar, E., Teevan, J., Dumais, S.T., Elsas, J.L.: The web changes everything: understanding the dynamics of web content. In: WSDM 2009, pp. 282–291. ACM, New York (2009)

2. Fetterly, D., Manasse, M., Najork, M., Wiener, J.L.: A large-scale study of the evolution of web pages. In: WWW 2003, pp. 669–678. ACM Press, New York (2003)
3. Ntoulas, A., Cho, J., Olston, C.: What’s new on the web? the evolution of the web from a search engine perspective. In: WWW 2004, pp. 1–12. ACM Press, New York (2004)
4. Selberg, E., Etzioni, O.: On the instability of web search engines. In: Proceedings of RIAO 2000, pp. 223–235 (2000)
5. Teevan, J., Adar, E., Jones, R., Potts, M.A.S.: Information re-retrieval: repeat queries in yahoo’s logs. In: SIGIR 2007, pp. 151–158. ACM, New York (2007)



# Rules of Thumb for Information Acquisition from Large and Redundant Data

Wolfgang Gatterbauer

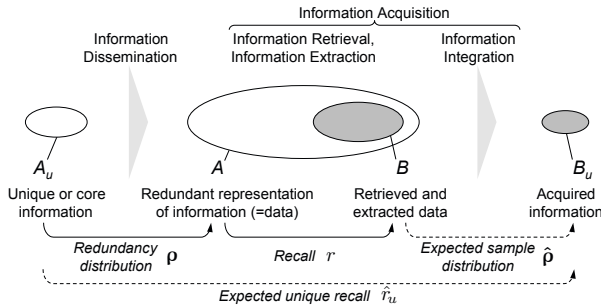
Computer Science and Engineering,  
University of Washington, Seattle  
gatter@cs.washington.edu

**Abstract.** We develop an abstract model of information acquisition from redundant data. We assume a random sampling process from data which contain information with bias and are interested in the fraction of information we expect to learn as function of (i) the sampled fraction (*recall*) and (ii) varying bias of information (*redundancy distributions*). We develop two rules of thumb with varying robustness. We first show that, when information bias follows a Zipf distribution, the 80-20 rule or Pareto principle does surprisingly not hold, and we rather *expect to learn less than 40% of the information when randomly sampling 20% of the overall data*. We then analytically prove that for large data sets, randomized sampling from power-law distributions leads to “truncated distributions” with the same power-law exponent. This second rule is very robust and also holds for distributions that deviate substantially from a strict power law. We further give one particular family of power-law functions that remain completely invariant under sampling. Finally, we validate our model with two large Web data sets: link distributions to web domains and tag distributions on delicious.com.

## 1 Introduction

The 80-20 rule (also known as Pareto principle) states that, often in life, 20% of effort can roughly achieve 80% of the desired effects. An interesting question is as to whether this rule also holds in the context of information acquisition from redundant data. Intuitively, we know that we can find more information on a given topic by gathering a larger number of data points. However, we also know that the marginal benefit of knowing additional data decreases with the size of the sampled corpus. Does the 80-20 rule hold for information acquisition from redundant data? Can we learn 80% of URLs on the Web by parsing only 20% of the web pages? Can we learn 80% of the used vocabulary by looking at only 20% of the tags? Can we learn 80% of the news by reading 20% of the newspapers? More generally, *can we learn 80% of all available information in a corpus by randomly sampling 20% of data* without replacement?

In this paper, we show that when assuming a Zipf redundancy distribution, the Pareto principle does not hold. Instead, we rather expect to see less than 40% of the available information. To show this in a principled, yet abstract



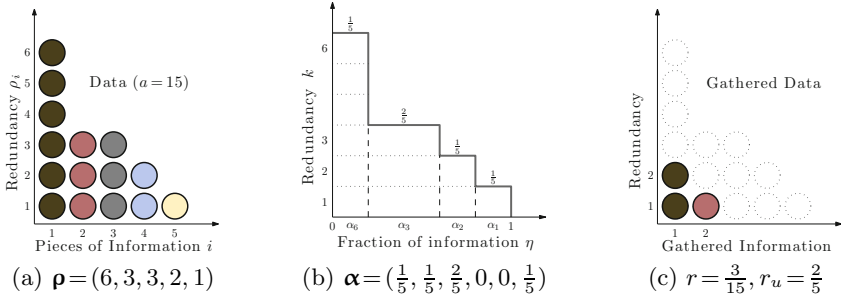
**Fig. 1.** Processes of information dissemination and information acquisition. We want to predict the fraction of information we can learn ( $\hat{r}_u$ ) as a function of recall ( $r$ ) and the bias in the data (redundancy distribution  $\rho$ ).

fashion, we develop an *analytic sampling model* of information acquisition from redundant data. We assume the *dissemination* of relevant information is biased, i.e. different pieces of information are more or less frequently represented in available sources. We refer to this bias as *redundancy distribution* in accordance with work on redundancy in information extraction [8]. Information acquisition, in turn, can be conceptually broken down into the subsequent steps of IR, IE, and II, i.e. visiting a fraction  $r$  of the available sources, extracting the information, and combining it into a unified view (see Fig. 1). Our model relies on only three simple abstractions: (1) we consider a purely *randomized sampling* process without replacement; (2) we do not model *disambiguation* of the data, which is a major topic in information extraction, but not our focus; and (3) we consider the process in the limit of *infinitely large data sets*. With these three assumptions, we estimate the success of information acquisition as function of the (i) *recall* of the retrieval process and (ii) *bias in redundancy* of the underlying data.

**Main contributions.** We develop an analytic model for the information acquisition from redundant data and (1) derive the 40-20 rule, a modification of the Pareto principle which has not been stated before. (2) While power laws do not remain invariant under sampling in general [19], we prove that one particular power law family does remain invariant. (3) While other power laws do not remain invariant in their overall shape, we further prove that the “core” of such a frequency distribution does remain invariant; this observations allows us to develop a second rule of thumb. (4) We validate our predictions by randomly sampling from two very large real-world data sets with power-law behavior. All proofs and more details are available in the full version of this paper [13].

## 2 Basic Notions Used Throughout This Paper

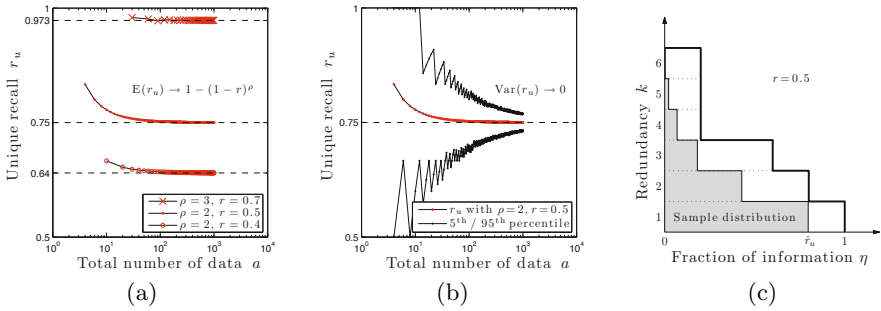
We use the term *redundancy* as synonym for frequency or multiplicity. We do so to remain consistent with the term commonly used in web information extraction, referring to the redundant nature of information on the Web. The notions of data and information are defined in various and partly contradicting ways



**Fig. 2.** (a): Redundancy distribution  $\rho$ . (b): Representation as *redundancy frequency distribution*  $\alpha$ . (c): Recall  $r = \frac{3}{15}$  and *unique recall*  $r_u = \frac{2}{5}$ .

in the information retrieval, information extraction, database and data integration literature. In general, their difference is attributed to novelty, relevance, organization, available context or interpretation. The most commonly found understanding is that of *data as representation of information* which can become information when it is interpreted as new and relevant in a given context [4]. In this work, we follow this understanding of data as “raw” information and use the term data for the partly redundant representation of information.

Let  $a$  be the total number of data items and  $a_u$  the number of unique pieces of information among them. *Average redundancy*  $\rho$  is simply their ratio  $\rho = \frac{a}{a_u}$ . Let  $\rho_i$  refer to the redundancy of the  $i$ -th most frequent piece of information. The *redundancy distribution*  $\rho$  (also known as rank-frequency distribution) is the vector  $\rho = (\rho_1, \dots, \rho_{a_u})$ . Figure 2a provides the intuition with a simple balls-and-urn model: Here, each color represents a piece of information and each ball represents a data item. As there are 3 red balls, redundancy of the information “color = red” is 3. Next, let  $\alpha_k$  be the fraction of information with redundancy equal to  $k$ ,  $k \in [k_{\max}]$ . A *redundancy frequency distribution* (also known as count-frequency plot) is the vector  $\alpha = (\alpha_1, \dots, \alpha_{k_{\max}})$ . It allows us to describe redundancy without regard to the overall number of data items  $a$  (see Fig. 2b) and, as we see later, an analytic treatment of sampling for the limit of infinitely large data sets. We further use the term *redundancy layer* (also known as complementary cumulative frequency distribution or ccf)  $\eta_k$  to describe the fraction of information that appears with redundancy  $\geq k$ :  $\eta_k = \sum_{i=k}^{k_{\max}} \alpha_i$ . For example, in Fig. 2a and Fig. 2b, the fraction of information with redundancy at least 3 is  $\eta_3 = \alpha_3 + \alpha_6 = \frac{2}{5} + \frac{1}{5} = \frac{3}{5}$ . Finally, *recall* is the well known measure for the coverage of a data gathering or selection process. Let  $b$  be a retrieved subset of the  $a$  total data items. Recall is then  $r = \frac{b}{a}$ . We define *unique recall* as is its counterpart for unique data items. Thus, it measures the coverage of information. Let  $b_u$  be the number of unique pieces of information among  $b$ , and  $a_u$  the number of unique pieces of information among  $a$ . Unique recall  $r_u$  is then  $r_u = \frac{b_u}{a_u}$ . We illustrate again with the urns model: assume that we randomly gather 3 from the 15 total balls (recall  $r = \frac{3}{15}$ ) and that, thereby, we learn 2 colors out of the 5 total available colors (Fig. 2c). Unique recall is thus  $r_u = \frac{2}{5}$  and the sample redundancy distribution is  $\hat{\rho} = (2, 1)$ .



**Fig. 3.** (a, b): Random sampling from an urn filled with  $a$  balls in  $a_u$  different colors. Each color appears on exactly  $\rho = \frac{a}{a_u}$  balls. (c): Normalized sample distribution in grey with unique recall  $\hat{r}_u \approx 0.8$  for  $r = 0.5$  from  $\alpha$  in [Fig. 2b](#).

### 3 Unique Recall

We next give an analytic description of *sampling without replacement* as function of recall and the bias of available information in the limit of very large data sets.

**Proposition 1 (Unique recall  $\hat{r}_u$ ).** *Assume randomized sampling without replacement with recall  $r \in [0, 1]$  from a data set with redundancy frequency distribution  $\alpha$ . The expected value of unique recall for large data sets is asymptotically concentrated around* 
$$\hat{r}_u = 1 - \sum_{k=1}^{k_{\max}} \alpha_k (1-r)^k.$$

The proof applies Stirling’s formula and a number of analytic transformations to a combinatorial formulation of a balls-and-urn model. The important consequence of [Prop. 1](#) is now that unique recall can be investigated without knowledge of the actual number of data items  $a$ , but by just analyzing the normalized redundancy distributions. Hence, we can draw general conclusions for families of redundancy distributions assuming very large data sets. To simplify the presentation and to remind us of this limit consideration, we will use the hat symbol and write  $\hat{r}_u$  for  $\lim_{a \rightarrow \infty} \mathbf{E}(r_u) \simeq \mathbf{E}(r_u)$ .

[Figure 3](#) illustrates this limit value with two examples. First, assume an urn filled with  $a$  balls in  $a_u$  different colors. Each color appears on exactly two balls, hence  $\rho = 2$  and  $a = 2a_u$ . Then the expected value of unique recall  $r_u$  (fraction of colors sampled) is converging towards  $1 - (1-r)^\rho$  and its variance towards 0 for increasing numbers of balls  $a$  ([Fig. 3a](#), [Fig. 3b](#)). For example, keeping  $\rho = 2$  and  $r = 0.5$  fixed, and varying only  $a = 4, 6, 8, 10, \dots$ , then unique recall varies as  $r_u = 0.83, 0.80, 0.79, 0.78, \dots$ , and converges towards  $\hat{r}_u = 0.75$ . At  $a = 1000$ ,  $r_u$  is already  $0.7503 \pm 0.02$  with 90% confidence. Second, assume that we sample 50% of balls from the distribution  $\alpha = (\frac{1}{5}, \frac{1}{5}, \frac{2}{5}, 0, 0, \frac{1}{5})$  of [Fig. 2b](#). Then we can expect to learn  $\approx 80\%$  of the colors if  $a$  is very large ([Fig. 3c](#)). In contrast, exact calculations show that if  $a = 15$  as in [Fig. 2a](#), then the actual value of  $\mathbf{E}(r_u)$  is around  $\approx 79\%$  or  $\approx 84\%$  for sampling 7 or 8 balls, respectively (note that 7.5 is 50% of 15). Thus, [Prop. 1](#) calculates the exact asymptotic value only for the limit, but already gives very good approximations for large data sets.

## 4 Unique Recall for Power Law Redundancy Distributions

Due to their well-known ubiquity, we will next study *power law redundancy distributions*. We distinguish three alternative definitions: (1) power laws in the redundancy distributions, (2) in the redundancy frequencies, and (3) in the redundancy layers. These three power laws are commonly considered to be different expressions of the identical distribution [20,16] because they have the same tail distribution [1], and they are in fact identical in a continuous regime. However, for discrete values, these three definitions of power laws actually produce different distributions and have different unique recall functions. We will show this next.

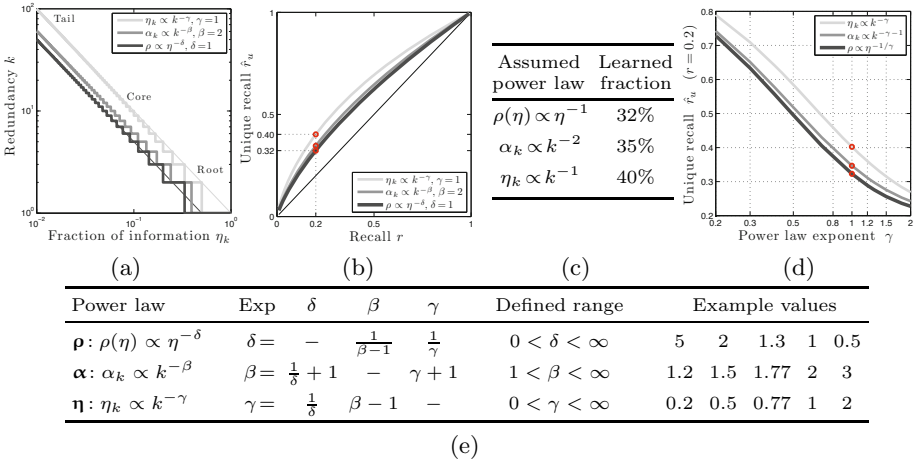
**Power laws in the redundancy distribution  $\rho$ .** This distribution arises when the frequency or redundancy  $\rho$  of an item is proportional to a power law with exponent  $\delta$  of its rank  $i$ :  $\rho(i) \propto i^{-\delta}$ ,  $i \in [a_u]$ . Two often cited examples of such power law redundancy distributions where  $\delta \approx 1$  are the frequency-rank distribution of the biggest words appearing in arbitrary corpora and the size distribution of the biggest cities for most countries. These are called “Zipf Distribution” after [20]. Using [Prop. 1] we can derive in a few steps  $\hat{r}_{u\rho}(r, \delta) = 1 - \sum_{k=1}^{\infty} ((2k-1)^{-\frac{1}{\delta}} - (2k+1)^{-\frac{1}{\delta}})(1-r)^k$ . For the particularly interesting case of  $\delta = 1$ , this infinite sum can be reduced to  $\hat{r}_{u\rho}(r, \delta=1) = \frac{r}{\sqrt{1-r}} \operatorname{artanh}(\sqrt{1-r})$ .

**Power laws in the redundancy frequency distribution  $\alpha$ .** This distribution arises when a fraction of information  $\alpha_k$  that appears exactly  $k$  times follows a power law  $\alpha_k = C \cdot k^{-\beta}$ ,  $k \in \mathbb{N}_1$ . Again, using [Prop. 1] we can derive in a few steps  $\hat{r}_{u\alpha}(r, \beta) = 1 - \frac{\operatorname{Li}_{\beta}(1-r)}{\zeta(\beta)}$ , where  $\operatorname{Li}_{\beta}(x)$  is the polylogarithm  $\operatorname{Li}_{\beta}(x) = \sum_{k=1}^{\infty} k^{-\beta} x^k$ , and  $\zeta(\beta)$  the Riemann zeta function  $\zeta(\beta) = \sum_{k=1}^{\infty} k^{-\beta}$ .

**Power laws in the redundancy layers  $\eta$ .** This distribution arises when the redundancy layers  $\eta_k \in [0, 1]$  follow a power law  $\eta_k \propto k^{-\gamma}$ . From  $\eta_1 = 1$ , we get  $\eta_k = k^{-\gamma}$  and, hence,  $\alpha_k = k^{-\gamma} - (k+1)^{-\gamma}$ . Using again [Prop. 1], we get in a few steps  $\hat{r}_{u,\eta}(r, \gamma) = \frac{r}{1-r} \operatorname{Li}_{-\gamma}(1-r)$ . For the special case of  $\gamma = 1$ , we can use the property  $\operatorname{Li}_1(x) = -\ln(1-x)$  and simplify to  $\hat{r}_{u,\eta}(r, \gamma=1) = -\frac{r \ln r}{1-r}$ .

**Comparing unique recall for power laws.** All three power laws show the typical *power law tail* in the loglog plot of the redundancy distribution (loglog rank-frequency plot), and it is easily shown that the exponents can be calculated from each other according to [Fig. 4e]. However, the distributions are actually different at the *power law root* ([Fig. 4a]) and also lead to different unique recall functions. [Figure 4b] shows their different unique recall functions for the particular power law exponent of  $\delta=1$  ( $\gamma=1, \beta=2$ ), which is assumed to be the most common redundancy distribution of words in a large corpus [20] and many other frequency distributions [16,17]. Given our normalized framework, we can now ask the interesting question: *Does the 80-20 rule hold for information acquisition assuming a Zipf distribution?* Put differently, if we sample 20% of the total

<sup>1</sup> With *tail of a distribution*, we refer to the part of a redundancy distribution for  $\eta \rightarrow 0$ , with *root* to  $\eta \rightarrow 1$ , and with *core* to the interval in between (see [Fig. 4a]).



**Fig. 4.** The three power law redundancy distributions (e) have the same *power law tail* and *power law core* but different *power law roots* in the loglog redundancy plot (a). This leads to different unique recall functions (b&d), and different fractions of information learned after sampling 20% data (c).

amount of data (e.g., read 20% of a text corpus, or look at 20% of all existing tags on the Web), what percentage of the contained information (e.g., fraction of different words in a corpus or the tagging data) can we expect to learn if redundancy follows a Zipf distribution? [Figure 4c](#) lists the results for the three power law distributions and shows that we can only expect to learn between 32% and 40% of the information, depending on which from the three definitions we choose. Note that we can apply this rule of thumb without knowing the total amount of available information. Also note that these numbers are sensitive to the power law root and, hence, to deviations from an ideal power law. This is also why unique recall diverges for our 3 variations of power law definitions in the first place ([Fig. 4a](#)). Finally, [Fig. 4d](#) shows that the power law exponent would have to be considerably different from  $\gamma = 1$  to give a 80-20 rule.

**Rule of thumb 1 (40-20 rule).** *When randomly sampling 20% of data whose redundancy distribution follows an exact Zipf distribution, we expect to learn less than 40% of the contained information.*

## 5 K-Recall and the Evolution of Redundancy Distributions

So far, we were interested in the expected fraction  $r_u$  of information we learn when we randomly sample a fraction  $r$  of the total data. We now generalize the question and derive an analytic description of the overall shape of the *expected sample redundancy distribution* ([Fig. 3c](#)). As it turns out, and what will become clear in this and the following section, the natural way to study and solve this question is again to *analyze the horizontal “evolution” of the redundancy layers  $\eta$*

during sampling. To generalize unique recall  $r_u$ , we define  $k$ -recall as the fraction  $r_{uk}$  of information that has redundancy  $\geq k$  and also appears at least  $k$  times in our sample. More formally, let  $a_{uk}$  be the number of unique pieces of information with redundancy  $\geq k$  in a data set, and let  $b_{uk}$  be the number of unique pieces of information with redundancy  $\geq k$  in a sample.  $K$ -recall  $r_{uk}$  is then the fraction of  $a_{uk}$  that has been sampled:  $r_{uk} = \frac{b_{uk}}{a_{uk}}$ . The special case  $r_{u1}$  is then simply the so far discussed unique recall  $r_u$ . We assume large data sets throughout this and all following section without always explicitly using the hat notation  $\hat{r}_{uk}$ .

$K$ -recall has its special relevance when sampling from partly unreliable data. In such circumstances, the general fall-back option is to assume a piece of information to be true when it is independently learned from at least  $k$  different sources. This approach is used in statistical polling, in many artificial intelligence applications of learning from unreliable information, and in consensus-driven decision systems: Counting the number of times a piece of information is occurring (its *support*) is used as strong indicator for its truth. As such, *to believe a piece of information only when it appears at least  $k$  times in a random sample* serves as starting point from which more complicated polling schemes can be conceived. In this context,  $r_{uk}$  gives the ratio of information that we learn *and* consider true (it appears  $\geq k$  times in our sample) to the overall information that we would consider true if known to us (it appears  $\geq k$  times in the data set) (Fig. 5a).

We also introduce a variable  $\omega_k$  for the fraction of total information we get in our sample that appears at least  $k$  times instead of just once. Note that  $\omega_k = \eta_k r_{uk} = \frac{b_{uk}}{a_u}$ . All  $\omega_k$  with  $k \in [k_{\max}]$  together form the vector  $\boldsymbol{\omega}$  representing the *sample redundancy layers* in a random sample with  $r \in [0, 1]$ . As  $r$  increases from 0 to 1, it “evolves” from the  $k_{\max}$ -dimensional null vector  $\mathbf{0}$  to the redundancy layers  $\boldsymbol{\eta}$  of the original redundancy distribution. Because of this intuitive interpretation, we call *evolution of redundancy* the transformation of a redundancy frequency distribution given by the redundancy layers  $\boldsymbol{\eta}$  to the expected distribution  $\boldsymbol{\omega}$  as a function of  $r$ :  $\boldsymbol{\eta} \xrightarrow{r} \boldsymbol{\omega}$ ,  $r \in [0, 1]$ . We further use  $\Delta_k$  to describe the fraction of information with redundancy exactly  $k$ :  $\Delta_k = \omega_k - \omega_{k+1}$ . To define this equation for all  $k \in \mathbb{N}_0$ , we make the convention  $\omega_0 = 1$  and  $\omega_k = 0$  for  $k > k_{\max}$ . We can then derive the following analytic description:

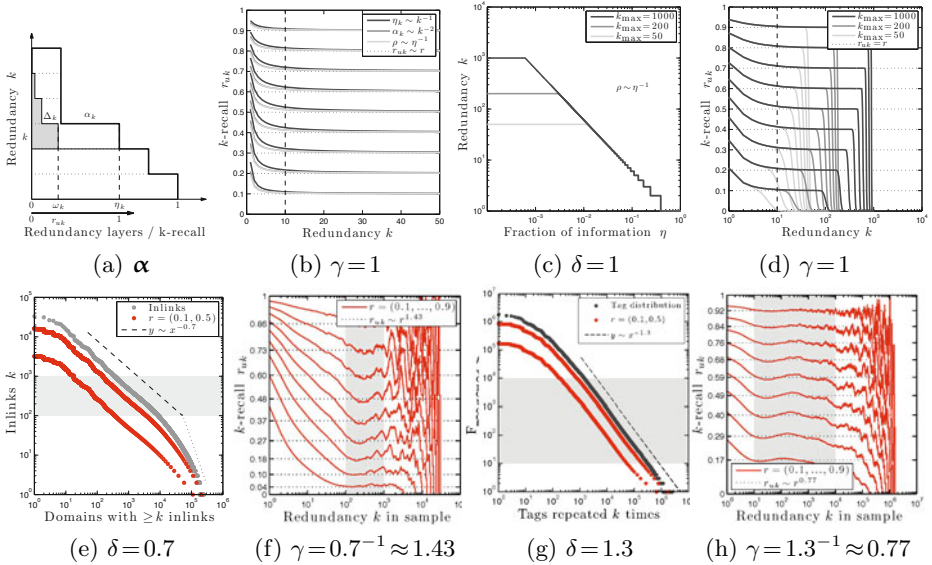
**Proposition 2 (Sample distribution  $\boldsymbol{\omega}$ ).** *The asymptotic expectation of the fraction of information  $\omega_k$  that appears with redundancy  $\geq k$  in a randomly sampled fraction  $r$  without replacement from a data set with redundancy distribution*

$\boldsymbol{\alpha}$  is  $\boxed{\hat{\omega}_k = 1 - \sum_{y=0}^{k-1} \sum_{x=y}^{\infty} \alpha_x \binom{x}{y} r^y (1-r)^{x-y}}$  for  $\lim_{a \rightarrow \infty}$ .

The first part of the proof constructs a geometric model of sampling from infinitely large data sets with homogenous redundancy and derives the binomial distribution as evolution of the redundancy layers. The second part then applies this result to stratified sampling from arbitrary redundancy distributions.

## 6 The Evolution of Power Laws

Given the complexity of Prop. 2, it seems at first sight that we have not achieved much. As it turns out, however, this equation hides a beautiful simplicity for



**Fig. 5.** Given a redundancy frequency distribution  $\alpha$  and recall  $r$ . K-recall  $r_{uk}$  describes the fraction of information appearing  $\geq k$  times that also appears  $\geq k$  times in our sample:  $r_{uk} = \frac{\omega_k}{\eta_k}$  (a). Sampling from *completely developed power laws* leads to sample distributions with the same power law tail, and  $r_{uk} \approx r^\gamma$  holds independent of  $k$  for  $k \gtrsim 10$  (b). *Truncated power laws* are cut off at some maximum value  $k_{\max}$  (c). As a consequence, the tails of the sample distributions “break in” for increasingly lower recalls (d). However, the invariant power law core with  $r_{uk} \approx r^\gamma$  is still visible. Original and sample web link distribution (e). Resulting k-recalls (f). Original and sample tag distribution on Delicious (g). Resulting k-recalls (h).

power laws: namely, their overall shape remains “almost” invariant during sampling. We will first formalize this notion, then prove it, and finally use it for another, very robust rule of thumb.

We say a redundancy distribution  $\alpha$  is *invariant under sampling* if, independent of  $r$ , the expected *normalized sample distribution*  $\Delta/\omega_1$  is the same as the *original distribution*:  $\frac{\Delta_k}{\omega_1} = \alpha_k$ . Hence, for an invariant distribution it holds that  $\omega_k = \sum_{x=k+1}^\infty \Delta_x = \omega_1 \sum_{x=k+1}^\infty \alpha_x = \omega_1 \eta_k$ , and, hence,  $r_{uk}$  is independent of  $k$ :  $r_{uk} = \omega_1$ . With this background, we can state the following lemma:

**Lemma 1 (Invariant family).** *The following family of redundancy distributions is invariant under sampling:  $\alpha_k = (-1)^{k-1} \binom{\tau}{k}$ , with  $0 < \tau \leq 1$ .*

The proof of **Lemma 1** succeeds by applying **Prop. 2** to the invariant family and deriving  $r_{uk} = r^\tau$  after application of several binomial identities. Note that the invariant family has a power law tail. We see that by calculating its asymptotic behavior with the help of the asymptotic of the binomial coefficient  $\binom{\tau}{k} = \mathcal{O}\left(\frac{1}{k^{1+\tau}}\right)$ , as  $k \rightarrow \infty$ , for  $\tau \notin \mathbb{N}$ . Therefore, we also have  $\alpha_k = \mathcal{O}\left(k^{-(1+\tau)}\right)$  for  $k \rightarrow \infty$ . Comparing this equation with the power-law in the redundancy



frequency plot,  $\alpha_k \propto k^{-\beta}$ , we get the power-law equivalent exponent as  $\beta = \tau + 1$ , with  $1 < \beta \leq 2$ . Also note that the invariant family is not “reasonable” according to the definition of [1], since the mean redundancy  $\sum_{k=1}^{\infty} \eta_k$  is not finite.

We next analyze sampling from *completely developed power laws*, i.e. distributions that have infinite layers of redundancy ( $k_{\max} \rightarrow \infty$ ). Clearly, those cannot exist in real discrete data sets, but their formal treatment allows us to also consider sampling from *truncated power laws*. The latter are real-world power law distributions which are truncated at  $k_{\max} \in \mathbb{N}$  (Fig. 5c). We prove that the power law core remains invariant for truncated power laws, and they, hence, appear as “almost” invariant over a large range, i.e. except for their tail and their root.

**Lemma 2 (Completely developed power laws).** *Randomized sampling without replacement from redundancy distributions with completely developed power law tails  $\alpha_C$  leads to sample distributions with the same power law tails.*

**Theorem 1 (Truncated power laws).** *Randomized sampling without replacement from redundancy distributions with truncated power law tails  $\alpha_T$  leads to distributions with the same power law core but further truncated power law tails.*

The proof for Lemma 2 succeeds in a number of steps by showing that  $\lim_{i,j \rightarrow \infty} \frac{r_{u,i}}{r_{u,j}} = 1$  for distributions with  $\alpha_C$ . The proof of Theorem 1 builds upon this lemma and shows that  $\lim_{k \ll k_{\max}} \Delta_k(\alpha_T, r) = \Delta_k(\alpha_C, r)$ . In other words, Theorem 1 states that sampling from real-world power law distributions leads to distributions with the same power law core but possibly different tail and root. More formally,  $r_{uk} \approx r^\gamma$  for  $k_1 < k < k_2$ , where  $k_1$  and  $k_2$  depend on the actual distribution, maximum redundancy and the power law exponent. Both, tail and root, are usually ignored when judging whether a distribution follows a power law (cf. Figure 3 in [6]), and to the best of our knowledge, this result is new. Furthermore, it is only recently that Stumpf et al. [19] have shown that sampling from power laws does not lead to power laws in the sample, in general. Our results clarify this result and shows that *only their tails and roots are subject to change*. Figure 5b, Fig. 5c and Fig. 5d illustrate our result.

**Rule of thumb 2 (Power law cores).** *When randomly sampling from a power law redundancy distribution, we can expect the sample distribution to be a power law with the same power law exponent in the core:*  $r_{uk} \approx r^\gamma$  for  $k_1 < k < k_2$ .

## 7 Large Real-World Data Sets

**Data sets.** We use two large real-world data sets that exhibit power-law characteristics to verify and illustrate our rules of thumb: the number of links to web domains and the keyword distributions in social tagging applications.

(1) The first data set is a snapshot of a top level domain in the World Wide Web. It is the result of a complete crawl of the Web and several years old. The set contains 267,415 domains with 5.422,730 links pointing between them. From Fig. 5e, we see that the redundancy distribution follows a power law with

exponent  $\delta = 0.7$  ( $\gamma \approx 1.43$ ,  $\beta \approx 2.43$ ) for  $k \gtrsim 100$ . Below 100, however, the distribution considerably diverges from this exponent, which is why we expect that rule of thumb 1 does not apply well. We now assume random sampling amongst all links in this data set (e.g. we randomly choose links and discover new domains) and ask: (i) what is the expected number of domains and their relative support (as indicated by linking to it) that we learn as function of the percentage of links seen? (ii) what is the fraction of domains with support  $\geq k$  in the original data that we learn with the same redundancy?

(2) The second data set concerns different keywords and their frequencies used on the social bookmarking web service Delicious (<http://delicious.com>). A total number of  $\approx 140$  Mio tags are recorded of which  $\approx 2.5$  Mio keywords are distinct [5]. The redundancy distribution (Fig. 5g) follows a power law with exponent  $\delta = 1.3$  ( $\gamma \approx 0.77$ ,  $\beta \approx 1.77$ ) very well except for the tail and the very root. Here we assume random sampling amongst all individual tags given by users (e.g. we do not have access to the database of Delicious, but rather crawl the website) and ask: (i) what is the expected number of different tags and their relative redundancies that we learn as function of the percentage of all tags seen? (ii) what is the fraction of important tags in the sample (tags with redundancy at least  $k$ ) that we can also identify as important by sampling a fraction  $r$ ?

**Results.** From Fig. 5f and Fig. 5h we see that after sampling 20% of links and tags respectively, we learn 60% and 40% of the domains and words, respectively. Hence, our first rule of thumb works well only for the second data set which better follows a power law (compare also with the prediction for arbitrary exponents in Fig. 4d). Our second rule of thumb, however, works well for both data sets: In Fig. 5f we see that, in accordance with our prediction, the horizontal lines for  $r_{uk} = r^\gamma$  become apparent for  $10^2 < k < 10^3$ , and in Fig. 5h, for  $10^1 < k < 10^4$  (compare with our prediction in Fig. 5d).

## 8 Related Work

Whereas the influence of redundancy of a search process has been widely analyzed [3,15,18], and randomized sampling used in other papers in this field [8,15], our approach is new in the way that we analytically characterize the behavior of the sampling process as a function of (i) the bias in redundancy of the data and (ii) recall of the used retrieval process. In particular, this approach allows us to prove a to date unknown characteristics of power laws during sampling. Achlioptas et al. [1] give a mathematical model that shows that traceroute sampling from Poisson-distributed random graphs leads to power laws. Their analysis is limited to “reasonable” power laws, which are such for which  $\alpha > 2$  and also assumes a very concrete sampling process tailored to their context. This is different from our result which proves that completely developed power law functions retain their power law tail, and truncated power laws at least their power law core during sampling. Haas et al. [14] investigate ways to estimate the number of different attribute values in a given database. This problem is related in its background, but different in its focus. We estimate the number

of unique attributes seen after sampling a fraction and later the overall sample distribution. Stump et al. [19] show that, in general, power laws do not remain invariant under sampling. In this paper, we could show that – while not in their entirety – at least the *core of power laws remains invariant under sampling*. General balls-and-urn models have been treated in detail by Gardy [11]. Gardy showed a general theorem which contains Prop. 1 as a special case. However, she does neither investigate the behavior of power laws during sampling, nor extends this result to the evolution of the overall distribution (Prop. 2). Only the latter allowed us to investigate the overall shape of redundancy distributions during sampling. Flajolet and Sedgewick [9] study the evolution of balanced, single urn models of finite dimensions under random sampling, where dimensionality refers to the number of colors. They mainly focus on urn models of dimension 2 (i.e., balls can be of either of two colors), and also solve some special cases for higher dimensions. They further note, that there is no hope to obtain general solutions for higher dimensions, however, that special cases warrant further investigation. Using a slightly different nomenclature, we also studied a special case of balanced, single urn models, however with infinite dimension (i.e., infinite number of colors), and showed that this case allows closed analytic solutions.

In [12], we gave Prop. 1 and motivated the role of different families of redundancy distributions on the effectiveness of information acquisition. However, we did not treat the case of power laws, nor the evolution of distributions during sampling (Prop. 2). To the best of our knowledge, the main results in this paper are new. Our analytic treatment of power laws during sampling, the invariant family, and the proof that sections of power laws remain invariant are not mentioned in any prior work we are aware of (cf. [7,9,10,11,16,17]).

## 9 Discussion and Outlook

Our target with this paper was to develop a general model of the information acquisition process (retrieval, extraction and integration) that allows us to estimate the overall success rate when acquiring information from redundant data. With our model, we derive the 40-20 rule of thumb, an adaptation of the Pareto principle. This is a negative result as to what can be achieved, in general. A crucial idea underlying our mathematical treatment of sampling was adopting a horizontal perspective of sampling and *thinking in layers of redundancy* (“k-recall”). Whereas our approach assumes an infinite amount of data, we have shown our approximation holds very well for large data sets (see Fig. 2b). We have focused on power laws, as they are the dominant form of biased frequency distributions. Whereas Stump et al. [19] have shown that, in general, power laws do not remain invariant under sampling, we have shown that (i) there exists one concrete family of power laws which *does* remain invariant, and (ii) while power laws do not remain invariant in their tails and root, their core does remain invariant. And we have used this observation to develop a second rule of thumb which turns out to be very robust (cp. Fig. 5d with Fig. 5h).

**Acknowledgements.** This work was partially supported by a DOC scholarship from the Austrian Academy of Sciences, by the FIT-IT program of the Austrian Federal Ministry for Transport, Innovation and Technology, and by NSF IIS-0915054. The author would like to thank one of the major search engines for the web link data, Ciro Cattuto and the TAGora project for the Delicious tag data, and Bernhard Gittenberger and the anonymous reviewers for helpful comments. More details are available on the project page: <http://uniquerecall.com>

## References

1. Achlioptas, D., Clauset, A., Kempe, D., Moore, C.: On the bias of traceroute sampling: or, power-law degree distributions in regular graphs. In: STOC, pp. 694–703 (2005)
2. Adamic, L.A.: Zipf, power-law, pareto – a ranking tutorial. Technical report, Information Dynamics Lab, HP Labs, Palo Alto, CA 94304 (October 2000)
3. Bernstein, Y., Zobel, J.: Redundant documents and search effectiveness. In: CIKM, pp. 736–743 (2005)
4. Capurro, R., Hjørland, B.: The concept of information. *Annual Review of Information Science and Technology* 37(1), 343–411 (2003)
5. Cattuto, C., Loreto, V., Pietronero, L.: Semiotic dynamics and collaborative tagging. *PNAS* 104(5), 1461–1464 (2007)
6. Chaudhuri, S., Church, K.W., König, A.C., Sui, L.: Heavy-tailed distributions and multi-keyword queries. In: SIGIR, pp. 663–670 (2007)
7. Clauset, A., Shalizi, C.R., Newman, M.: Power-law distributions in empirical data. *SIAM Review* 51(4), 661–703 (2009)
8. Downey, D., Etzioni, O., Soderland, S.: A probabilistic model of redundancy in information extraction. In: IJCAI, pp. 1034–1041 (2005)
9. Flajolet, P., Dumas, P., Puyhaubert, V.: Some exactly solvable models of urn process theory. *Discrete Math. & Theoret. Comput. Sci. AG*, 59–118 (2006)
10. Flajolet, P., Sedgewick, R.: *Analytic combinatorics*. CUP (2009)
11. Gardy, D.: Normal limiting distributions for projection and semijoin sizes. *SIAM Journal on Discrete Mathematics* 5(2), 219–248 (1992)
12. Gatterbauer, W.: Estimating Required Recall for Successful Knowledge Acquisition from the Web. In: WWW, pp. 969–970 (2006)
13. Gatterbauer, W.: Rules of thumb for information acquisition from large and redundant data. *CoRR abs/1012.3502* (2010)
14. Haas, P.J., Naughton, J.F., Seshadri, S., Stokes, L.: Sampling-based estimation of the number of distinct values of an attribute. In: VLDB, pp. 311–322 (1995)
15. Ipeirotis, P.G., Agichtein, E., Jain, P., Gravano, L.: To search or to crawl? towards a query optimizer for text-centric tasks. In: SIGMOD, pp. 265–276 (2006)
16. Mitzenmacher, M.: A brief history of generative models for power law and lognormal distributions. *Internet Mathematics* 1(2), 226–251 (2004)
17. Newman, M.E.: Power laws, pareto distributions and zipf’s law. *Contemporary Physics* 46(5), 323–351 (2005)
18. Soboroff, I., Harman, D.: Overview of the trec 2003 novelty track. In: TREC 2003. NIST, pp. 38–53 (2003)
19. Stumpf, M.P.H., Wiuf, C., May, R.M.: Subnets of scale-free networks are not scale-free: sampling properties of networks. *PNAS* 102(12), 4221–4224 (2005)
20. Zipf, G.K.: *Human Behaviour and the Principle of Least Effort: an Introduction to Human Ecology*. Addison-Wesley, Reading (1949)

# Bringing Why-QA to Web Search

Suzan Verberne, Lou Boves, and Wessel Kraaij

Centre for Language Studies / Institute for Computing and Information Sciences  
Radboud University Nijmegen  
s.verberne@let.ru.nl

**Abstract.** We investigated to what extent users could be satisfied by a web search engine for answering causal questions. We used an assessment environment in which a web search interface was simulated. For 1401 *why*-queries from a search engine log we pre-retrieved the first 10 results using Bing. 311 queries were assessed by human judges. We found that even without clicking a result, 25.2% of the *why*-questions is answered on the first result page. If we count an intended click on a result as a vote for relevance, then 74.4% of the *why*-questions gets at least one relevant answer in the top-10. 10% of *why*-queries asked to web search engines are not answerable according to human assessors.

## 1 Introduction and Background

The problem of automatically answering open-domain questions by pinpointing exact answers in a large text (web) corpus has been studied since the mid 1990s. Already in 2001, Kwok et al. [1] argued that if developers of open-domain Question Answering (QA) systems want their system to be useful for a large audience, QA should be web-based and integrated in existing search interfaces.

While approaches to QA for factoid, list and definition questions that might scale up to the web have been implemented and evaluated in the TREC (and CLEF) evaluation forums, QA for more complex questions such as *why*- and *how*-questions is considered a complex NLP task that requires advanced linguistic processing [2]. Accordingly, most research in *why*-QA (also called causal QA) is directed at extracting causal relations from text [3,4,5,6]. Some papers describe IR-based approaches to *why*-QA [7,8,9], but these use a restricted corpus of candidate answer documents (either newspaper texts or a static version of Wikipedia). Until now, no attempts have been made to investigate causal QA using web data and web search engines.

There is a huge amount of user-generated QA data available on the web [10], also for *why*-questions [11]. We think that *why*-QA, just as any other open-domain retrieval task, should be studied in the context and scale of the web. Ideally, a search engine should be able to recognize a query as a causal question, and provide the answer directly. The implementation of such a service is only possible if the answers to *why*-questions can be found by web search engines.

In the current paper, we investigate to what extent users could be satisfied by a web search engine for answering *why*-questions. We approximate user satisfaction

using an assessment environment in which a web search interface is simulated. Participants in the experiment are presented with *why*-queries from a search engine log and a ranked list of results that have been pre-retrieved with the Bing search engine<sup>1</sup>. The judges are asked to assess the relevance of the first ten results for each query.

Using this assessment environment we address the question “What proportion of *why*-questions can be answered satisfactorily by a state-of-the-art web search engine?” Other questions that we will address are: “To what extent do judges agree in their assessment of answers to causal questions?”, and “What proportion of *why*-queries that are posed to search engines are not answerable according to human assessors?”.

## 2 Web Data for Evaluating Why-QA

We obtained user-generated *why*-questions from the Microsoft RFP data set<sup>2</sup>. This collection consists of approximately 14 million queries from US users entered into the Microsoft Live search engine in the spring of 2006. In this data set, 2 879 queries (1 401 unique) start with the word ‘why’. There are a number of possible paraphrases of the question word ‘why’, but their frequency in the data is very low: ‘for what reason’ does not occur at all and ‘how come’ occurs in 11 queries. Therefore, we decided to only extract queries that start with the word *why*, assuming that they are representative for all causal queries. Since the majority of queries in our set are questions, we will use the words query and question interchangeably in the remainder of this paper.

Not all queries are syntactically complete sentences (e.g. “why so many religions”) and some are not even questions: “why you wanna lyrics”. We kept ungrammatical questions and queries with spelling errors in the set in order to reflect the noisiness of user queries. Moreover, we did not filter out questions with subjective answers (“why marriages fail”) or queries that are probably no questions (“why do fools fall in love”); we left the decision whether a query is answerable or not to the assessors. We only filtered out the queries that contain the word ‘lyrics’. The result is a set of 1 382 unique *why*-queries.

Using the Perl module LWP::Simple<sup>3</sup>, we sent all queries to the Bing search engine and extracted the 10 results from the first result page. For each of the results, we saved the title, URL and snippet. We refer to the combination of title, URL and snippet as an ‘answer’.

## 3 Relevance Assessment Set-Up

For the manual assessment of answers, we recruited subjects among colleagues and friends. We promised them a treat if they assessed at least twenty questions.

<sup>1</sup> [www.bing.com](http://www.bing.com)

<sup>2</sup> This set was distributed for the WSCD 2009 workshop.

<sup>3</sup> The module is available at <http://www.cpan.org/>

For the levels of answer relevance in our assessment task we adopt the distinction between “the passage answers the query” and the “passage does not answer the query” from [9]. In addition to this binary choice, we add the alternatives “don’t know” and “I expect to find the answer if I would click the link. The latter option was added because the snippets returned by Bing are short and users of web search engines are used to click on results they expect them to be relevant. In evaluation studies using click data, a click on a document is often considered a vote for the document’s relevance<sup>4</sup>.

The participants were allowed to skip a question, with or without providing a reason for skipping. Three different reasons are predefined: *There is no answer possible (the query is a joke or title)*, *The answer is subjective/depends on the person* and *I don’t understand the query*. If a question is skipped, all the corresponding snippets in the result list receive the label “don’t know”. In the user interface of the experiment, we imitate a web search engine’s formatting by using a larger font and blue for titles and green for the URL under the snippet. In the title and the snippet, query words are printed in bold face. Next to each answer are the radio buttons for the four assessment options.

Each time a participant logged into the assessment environment, a random query from the total set of *why*-queries is presented to him/her together with the first ten results that were returned by Bing. For the purpose of calculating agreement between the assessors, each assessor was presented at least one query that was already assessed by another assessor.

## 4 Results

Table 1 shows general statistics of the collected data. We collected answer assessments for 238 unique queries, 42 of which have been assessed by two assessors. This set may seem small but is big compared to sets of *why*-questions collected in previous work (see Section 5). When calculating the inter-judge agreement for the answers to these queries, we disregarded the answers that were labeled as “don’t know”. We measured strict agreement, where each of the three categories ‘yes’, ‘click’ and ‘no’ is considered independently, and lenient agreement, where ‘yes’ and ‘click’ are taken together as one category because they both represent relevant answers. Measured strictly, we found a fair agreement (Cohen’s  $\kappa = 0.34$ ); measured leniently, there was a moderate agreement (Cohen’s  $\kappa = 0.47$ ).

In the evaluation of the retrieval results on the basis of the user assessments, we also distinguish between strict and lenient evaluation. In Table 2, we present the results in terms of Success@10 (the percentage of questions with at least one relevant answer) and Precision@10 (the percentage of answers that is judged relevant). Figure 1 shows Success@n as a function of n.

We investigated the influence of query frequency and query length on the query success. We hypothesized that web search engines are more successful for

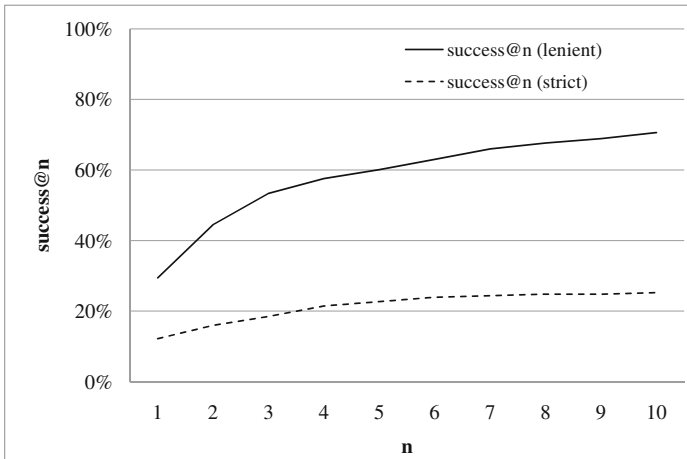
<sup>4</sup> Although it was shown by [12] that click behavior is biased by trust in the search engine and the quality of the ranking.

**Table 1.** General statistics of the collected data

Number of assessors	22
Number of questions assessed (including skipped)	311
Number of unique questions assessed (including skipped)	271
Number of unique questions skipped	33
Number of unique questions with at least one assessed answer	238
Number of answers with a judgment (other than ‘don’t know’)	2 105

**Table 2.** Evaluation of Bing (Summer 2010) for *why*-queries on the basis of collected user assessments. In addition to success@10 and precision@10, we present MRR (based on the highest ranked answer per question) because it was used in previous research on *why*-QA.

	strict	lenient
Success@10	25.2%	74.4%
Precision@10	8.1%	34.1%
Mean Reciprocal Rank (MRR)	0.163	0.500

**Fig. 1.** Success@n as a function of n (the length of the result list), over all *why*-queries

more frequent queries and for longer queries. For both predictors however, we found no correlation with average precision (Pearson’s  $\rho = 0.08$  for query frequency and  $\rho = 0.06$  for query length;  $N = 238$ ). Query length is approximately normally distributed with  $\mu = 6.2$  and  $\sigma = 2.6$  but since query frequency is relatively sparse (55% of the queries has a frequency of 1), these results may be falsified when using a larger click data set.

## 5 Comparison to Other Approaches

It is difficult to compare our current results to other work in *why*-QA because previous approaches either disregard the retrieval step of the QA task [3,4] or



address a different language than English [7]. The work described in [8,9] is the most comparable to our work because it evaluates an approach to *why*-QA for English that is based on passage retrieval, using a set of real users' questions (questions asked to [answers.com](http://answers.com)). The answer corpus used is a Wikipedia XML corpus and the evaluation set only contains *why*-questions for which the answer is present in this corpus (only 186 of the 700 Webclopedia *why*-questions). Moreover, grammatically incomplete questions and questions containing spelling errors were removed. Using TF-IDF only for ranking, MRR was 0.24 with a success@10 of 45% (precision@10 was not measured).

Since the web contains much more redundant information than Wikipedia proper, our evaluation shows a different pattern. Dependent on strict or lenient evaluation, success@10 is lower (25.2%) or much higher (74.4%) than the results in [9]. More strikingly, precision@10 is 34%, which means that one in three retrieved answers is relevant. Although we did not filter out questions for which no answer is available on the web (clearly, there is no good way to do this), both Success@10 and MRR (measured leniently) are much higher than the results reported in [9]. Apparently, the abundance of information on the web compensates heavily for the careful filtering of questions in previous work. This confirms previous findings in redundancy-based QA [13].

## 6 Discussion and Conclusion

We found that even without clicking a result, 25.2% of *why*-questions is answered by a state-of-the-art web search engine on the first result page. If we count an intended click on a result as a vote for relevance, then 74.4% of the *why*-questions gets at least one relevant answer in the top-10. The large difference between strict and lenient evaluation suggests that for *why*-queries, presenting longer snippets in the search engine output may increase user satisfaction.

We measured the difficulty of answer assessment for causal questions in terms of inter-judge agreement. Our  $\kappa$ -values suggest that if assessors are asked to judge other persons' causal questions, they quite often disagree on the relevance of the answers proposed by a search engine.

The assessors had the possibility of skipping questions, with or without ticking one of the reasons provided in the interface. 33 of the 311 *why*-queries were skipped. For most of these, the assessor provided a reason. The reason '*There is no answer possible (the query is a joke or title)*' was chosen for 5 queries such as "why did the chicken cross the road". The reason '*The answer is subjective/depends on the person*' was selected for 10 questions containing subjective judgements such as "why Disney Is Bad" and personal questions such as "why am I the best hockey coach for the position". The last reason '*I don't understand the query*' was selected 13 times, for complex queries such as "why circumscribing the role and behavior in the biblical community according to ephesian 6 text" but also for underspecified queries such as "why photography". Overall, we can conclude that 10% of *why*-queries asked to web search engines are not answerable according to human assessors.

In future work, we aim to investigate how a dedicated approach to *why*-QA can be implemented for web search, combining the output of Bing with knowledge from previous research in the computational linguistics community.

## References

1. Kwok, C., Etzioni, O., Weld, D.: Scaling question answering to the Web. *ACM Transactions on Information Systems (TOIS)* 19(3), 242–262 (2001)
2. Maybury, M.: Toward a Question Answering Roadmap. In: *New Directions in Question Answering*, pp. 8–11 (2003)
3. Girju, R.: Automatic detection of causal relations for question answering. In: *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*, vol. 12, p. 83. Association for Computational Linguistics (2003)
4. Pechsiri, C., Kawtrakul, A.: Mining Causality from Texts for Question Answering System. *IEICE Transactions on Information and Systems* 90(10), 1523–1533 (2007)
5. Verberne, S., Boves, L., Oostdijk, N., Coppens, P.: Discourse-based Answering of Why-Questions. *Traitement Automatique des Langues (TAL)*, special issue on Discours et document: traitements automatiques 47(2), 21–41 (2007)
6. Vazquez-Reyes, S., Black, W.: Evaluating Causal Questions for Question Answering. In: *Mexican International Conference on Computer Science, ENC 2008*, pp. 132–142 (2008)
7. Higashinaka, R., Isozaki, H.: Corpus-based Question Answering for Why-Questions. In: *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, pp. 418–425 (2008)
8. Verberne, S., Boves, L., Oostdijk, N., Coppens, P.: What is not in the Bag of Words for Why-QA? *Computational Linguistics* 32(2), 229–245 (2010)
9. Verberne, S., Van Halteren, H., Theijssen, D., Raaijmakers, S., Boves, L.: Learning to Rank for Why-Question Answering. In: *Information Retrieval (2010)*, doi:10.1007/s10791-010-9136-6
10. Adamic, L., Zhang, J., Bakshy, E., Ackerman, M.: Knowledge sharing and yahoo answers: everyone knows something. In: *Proceeding of the 17th International Conference on World Wide Web*, pp. 665–674. ACM, New York (2008)
11. Ignatova, K., Toprak, C., Bernhard, D., Gurevych, I.: Annotating Question Types in Social Q&A Sites. In: *GSCL Symposium” Language Technology and eHumanities*. Citeseer (2009)
12. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Gay, G.: Accurately interpreting clickthrough data as implicit feedback. In: *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 154–161. ACM, New York (2005)
13. Dumais, S., Banko, M., Brill, E., Lin, J., Ng, A.: Web question answering: Is more always better? In: *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 291–298. ACM, New York (2002)

# The Power of Peers

Nick Craswell<sup>1</sup>, Dennis Fetterly<sup>2</sup>, and Marc Najork<sup>2</sup>

<sup>1</sup> Microsoft, Bellevue, WA, USA  
nickcr@microsoft.com

<sup>2</sup> Microsoft Research Silicon Valley, Mountain View, CA, USA  
{fetterly,najork}@microsoft.com

**Abstract.** We present a study of the contributions of three classes of ranking signals: BM25F, a retrieval function that is based on words in the content of web pages and the anchors that link to them; SALSA, a link-based feature that takes all or part of the result set to a query as input; and matching-anchor count (MAC), a feature that measures precise matches between queries and anchors pointing to result pages. All three features incorporate both link and textual features, but in varying degrees. BM25F is the state-of-the-art exponent of Salton’s term-vector model, and is based on a solid theoretical foundation; the two other features are somewhat more ad-hoc. We studied the impact of two factors that go into the formation of SALSA’s “base” set: whether to use conjunctive or disjunctive query semantics, and how many results to include into the base set. We found that the choice of query semantics has little impact on the effectiveness of SALSA (with conjunctive semantics having a slight edge); more surprisingly, we found that limiting the size of the base set to a few hundred results of high expected quality maximizes performance. Furthermore, we experimented with various linear combinations of BM25F, MAC and SALSA. In doing so, we made a remarkable observation: adding BM25F to a two-way weighted linear combination of MAC and SALSA does not increase performance in any statistically significant way.

## 1 Introduction

In this work, we compare the ranking performance of linear combinations of three features: BM25F, SALSA-SETR, and MAC. BM25F [8] is a ranking function in the tradition of Salton’s term-vector model that is based on a solid theoretical model and that correlates query terms with terms in the title and body of a web page as well as in its URL and any HTML anchors pointing to it. SALSA-SETR [7] is a variant of SALSA [6] which in turn was derived from Kleinberg’s HITS [5] algorithm. It projects (some of) the results of a query onto the web graph, extracts (some of) the distance-one neighborhood graph, and computes “authority scores” on that graph. MAC (“matching-anchor count”) is a simple heuristic that measures how many anchors pointing to a given result precisely match the query [4]. More precisely, it counts the number of IP subnets containing hosts that serve pages containing one or more matching anchors.

Each of the three features incorporates both text and link information: BM25F is predominantly text-based, but it incorporates link information by considering anchor text. SALSA is predominantly link-based, but the “base set” of vertices that is the input to the SALSA algorithm is based on textual matching between query and corpus documents, and in our implementation, is furthermore biased toward documents that have high BM25F scores. MAC incorporates text and link features in equal (and quite simplistic ways), by looking for anchors (which imply links) whose text precisely matches the query.

We conducted our experiments using three data sets: the ClueWeb09 corpus [2], a collection of slightly over one billion web pages (we only use the 503 million English-language pages); the test set used in the TREC 2009 Web Track [1], which contains 50 queries and 27,964 results paired with binary judgments; and an additional test set comprised of the same 50 queries as the previous test set as well as 4,298 binary judgments completed by the authors.

Our evaluation measures are the ones that were used in the diversity task of the TREC 2009 Web Track. In addition we use the measure “IA-P@20 (judged)” where the denominator is the number of judged documents in the top-20 instead of all 20. This addresses the fact that the TREC test set is only partially judged, and that SALSA in particular surfaces a high number of unjudged results.

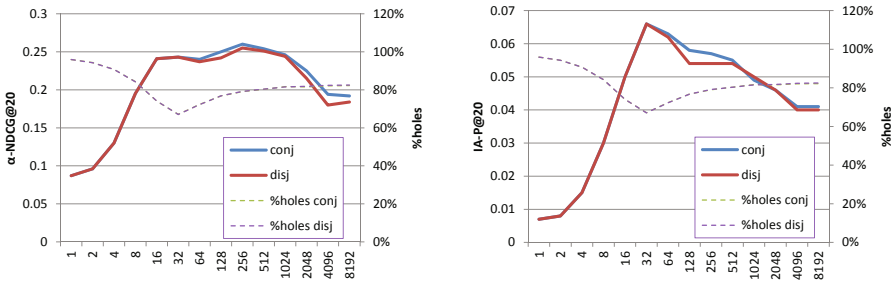
In order to quantify the impact of unjudged documents on system ranking, we completely judged the result sets for one weighted linear combination of features described in Section 3. We chose a document cutoff value of 20, which is consistent with other evaluations in this paper, which yielded 3,053 query-results pairs. The query-result pairs were grouped by query and then divided roughly into thirds, which were each evaluated by a single judge so that any particular judge would judge all of the results for a single query. Judging was performed using a tool that displayed information relevant to the query, such as the description of the information need, the type of facet of the query, and the information need for that specific facet, as well as the content of the page itself. A proxy server was employed so that stored content from the ClueWeb collection could be returned for document requests, but images, style sheets, and other page content would be requested from the original web site. We were unable to assess 37 query-result pairs, either because the page could not be rendered or because it contained malware. These judgments are available to the research community; please contact the authors to obtain them. The following table relates the TREC judgments (horizontal) to ours (vertical):

	NR	U	R
NR	995	873	154
U	20	13	4
R	261	320	413

For example, we considered 873 of the unjudged TREC results to be relevant. We computed Cohen’s Kappa statistic between the TREC 2009 judgments and ours. After flattening the subtopics, we have  $\kappa = 0.49$  and a 77% agreement rate.

## 2 Impact of Base Set on SALSA’s Performance

Our first set of experiments studied the impact of the selection and size of the “base set” that serves as the input to SALSA-SETR [7]. For a given query, we produced a “candidate result set” using either disjunctive ( $t_1$  OR  $t_2$ ) or conjunctive ( $t_1$  AND  $t_2$ ) semantics for multi-term queries, computed BM25F scores for each candidate, and admitted the  $k$  highest-scoring candidates into the “base set” that is the input to any HITS-like ranking algorithm, including SALSA-SETR. Figure 1 shows the results. The two graphs show two different effectiveness measures ( $\alpha$ -nDCG@20 and IA-P@20); the horizontal axis plots  $k$ ; the vertical axis plots effectiveness; and the two curves in each graph show the performance of the two query semantics we consider. We can see that the choice of query semantics does not have a great impact on effectiveness, although AND slightly outperforms OR. More interestingly, we find that SALSA is most effective when given a base set of ten to a few hundred results.



**Fig. 1.** Impact of the choice and size of the base set on SALSA-SETR’s performance. These results use the TREC judgments.

## 3 Combination of Features

For the remainder of this work, SALSA-SETR is parameterized to use conjunctive query semantics, form the base set out of the 5,000 top candidate results according to BM25F, and to vertex and edge sampling parameters  $a = 5$ ,  $b = 6$ ,  $c = 6200$ , and  $d = 2900$  (see [7] for a detailed description of the algorithm). We studied the performance of the three features in isolation, as well as the three possible pairwise and the one three-wise combination [4]. We applied a log-based transform function to MAC and SALSA when combining them with other features, and we weighted MAC by a factor of 1 and SALSA by a factor of 500. In our experience, increasing SALSA’s weight beyond 500 decreases  $\alpha$ -nDCG and IA-P, since it substantially increases the number of highly-ranked unlabeled results.

<sup>1</sup> Commercial search engines typically combine hundreds of features, e.g. see [3].

**Table 1.** Performance of individual features and feature combinations evaluated using the TREC 2009 judgments. Each ranker is a combination of the BM25F score  $B$ , the MAC score  $A$ , and/or the SALSA-SETR score  $S$ . \* indicates a significant difference from  $AS$ , in a one-tailed t-test, with  $p < 0.01$ .

Ranker	$\alpha$ -nDCG			IA-P			IA-P (judged)		
	@5	@10	@20	@5	@10	@20	@5	@10	@20
BAS	0.280	0.319	0.365	0.138	0.115	0.109	0.141	0.119	0.113
AS	0.284	0.325	0.363	0.135	0.117	0.094*	0.143	0.135	0.122
BA	0.282	0.311	0.361	0.138	0.113	0.108	0.141	0.116	0.112
A	0.275	0.300	0.350	0.131	0.104	0.090*	0.143	0.121	0.119
BS	0.208*	0.243*	0.281*	0.107	0.098	0.086*	0.129	0.121	0.113
B	0.180*	0.221*	0.253*	0.084*	0.082*	0.072*	0.092*	0.094	0.091*
S	0.143*	0.165*	0.193*	0.058*	0.051*	0.041*	0.109	0.114	0.114

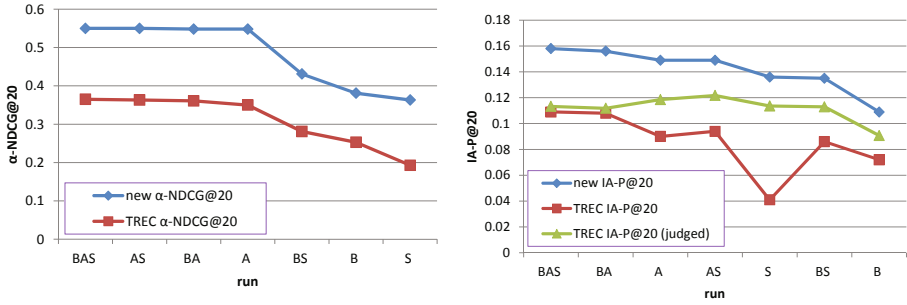
**Table 2.** Performance of individual features and feature combinations evaluated using the complete judgments performed by the authors. Each ranker is a combination of the BM25F score  $B$ , the MAC score  $A$ , and/or the SALSA-SETR score  $S$ . \* indicates a significant difference from  $AS$ , in a one-tailed t-test, with  $p < 0.01$ .

Ranker	$\alpha$ -nDCG			IA-P		
	@5	@10	@20	@5	@10	@20
BAS	0.443	0.491	0.550	0.201	0.174	0.158
AS	0.440	0.502	0.550	0.190	0.170	0.149
BA	0.443	0.487	0.548	0.198	0.171	0.156
A	0.433	0.482	0.548	0.197	0.167	0.149
BS	0.318*	0.370*	0.431*	0.163	0.155	0.135
B	0.262*	0.320*	0.381*	0.121*	0.116*	0.109*
S	0.256*	0.303*	0.363*	0.141	0.141	0.136

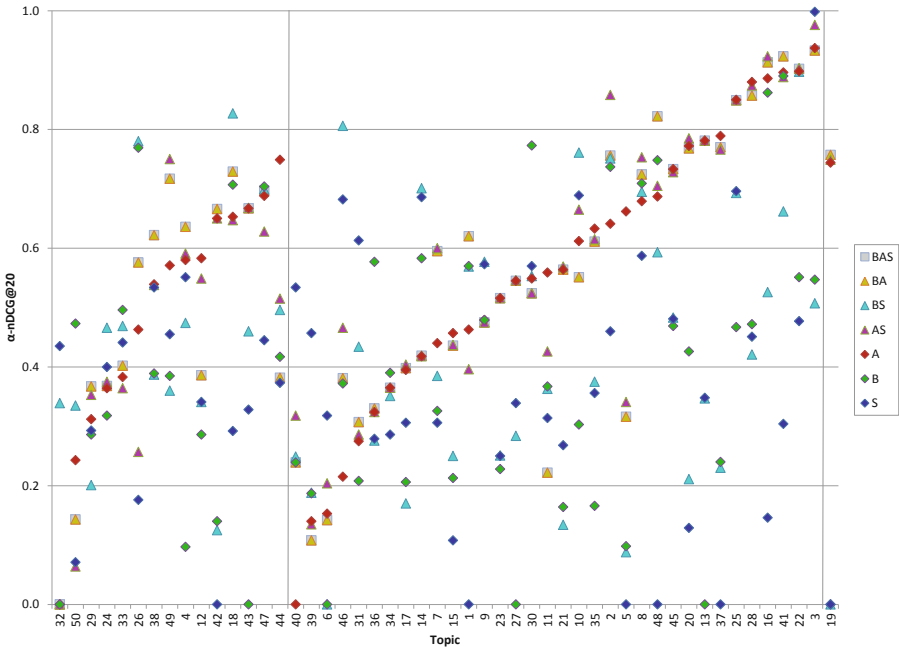
We combined the three features (BM25F, MAC, and SALSA) into all seven possible combinations using near-optimal weights, and used each resulting ranker to rank the result sets of each query. Table 1 shows the performance of each of these systems. We can observe that SALSA-SETR performs substantially better under IA-P (judged) than under IA-P or  $\alpha$ -nDCG. This is due to the fact that rankers using SALSA surface more unjudged results in the top twenty.

In order to quantify the performance of these systems on a completely judged result set, we pooled the top twenty results of each ranker and judged them using the methodology in Section 1. Table 2 shows the performance of these systems evaluated using the new judgments. MAC is the most efficient single feature, and furthermore, any combination involving MAC does not differ from any other such combination in a statistically significant fashion.

Figure 2 shows the performance of the seven rankers evaluated both the TREC judgments as well as the new judgments. The size of the gap between the curves for TREC IA-P@20 and TREC IA-P@20 (judged) illustrates the fraction of unjudged top 20 results for each ranker. This gap is very small for BAS, indicating



**Fig. 2.** The system ranking for  $\alpha$ -NDCG@20 is similar whether we use TREC judgments or our new judgments. The IA-P@20 system ranking under TREC judgments differs from our fully judged results, and in particular is worse for SALSA.



**Fig. 3.** Performance of the seven rankers in terms of  $\alpha$ -nDCG, broken down by topic and grouped by query intent (purely informational, informational+ navigational, purely navigational)

that the TREC judgments contain virtually no holes for the top-20 results returned by this ranker; conversely it is very large for S. The gap between TREC IA-P@20 (judged) and new IA-P@20 for the BAS ranker indicates that we considered more results to be relevant than the TREC assessors did. For the other rankers, a smaller gap indicates that more of the results that were unjudged in the TREC collection were considered non-relevant by us.

Finally, Figure 3 shows the performance of the seven rankers in terms of  $\alpha$ -nDCG, broken down by topic. The left 15 topics are purely informational, the rightmost topic is purely navigational, and the remaining 34 topics have both informational and navigational subtopics.

## 4 Conclusion

In this work, we studied the effectiveness – in isolation and in combination – of three ranking features that each incorporate both text and link information: BM25F, MAC, and SALSA. We used publicly available data sets and standard measures of retrieval effectiveness to investigate which and how many candidate results to incorporate into the base set of results, and how to combine these features to maximize effectiveness. To our surprise, we found that MAC, a fairly ad-hoc feature, performed better than any other feature, and that any combination of features involving MAC performed equally well in a statistically significant sense.

## References

1. Clarke, C., Craswell, N., Soboroff, I.: Report on the TREC 2009 Web Track. In: 18th Text Retrieval Conference (2009)
2. The ClueWeb 09 Dataset, <http://boston.lti.cs.cmu.edu/Data/clueweb09/>
3. Hansell, S.: Google keeps tweaking its search engine. New York Times (2007), <http://www.nytimes.com/2007/06/03/business/yourmoney/03google.html>
4. Craswell, N., Fetterly, D., Najork, M., Robertson, S., Yilmaz, E.: Microsoft Research at TREC 2009: Web and relevance feedback tracks. In: 18th Text Retrieval Conference (2009)
5. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: Proc. 9th Annual ACM-SIAM Symposium on Discrete Algorithms (1998)
6. Lempel, R., Moran, S.: The stochastic approach for link-structure analysis (SALSA) and the TKC effect. In: 9th International World Wide Web Conference (2000)
7. Najork, M., Gollapudi, S., Panigrahy, R.: Less is More: Sampling the neighborhood graph makes SALSA better and faster. In: 2nd ACM International Conference on Web Search and Data Mining (2009)
8. Zaragoza, H., Craswell, N., Taylor, M., Saria, S., Robertson, S.: Microsoft Cambridge at TREC-13: Web and HARD tracks. In: 13th Text Retrieval Conference (2004)



# Introducing the User-over-Ranking Hypothesis

Benno Stein and Matthias Hagen

Bauhaus-Universität Weimar  
first name.last name@uni-weimar.de

**Abstract.** The User-over-Ranking hypothesis states that rather the user herself than a web search engine’s ranking algorithm can help to improve retrieval performance. The means are longer queries that provide additional keywords.

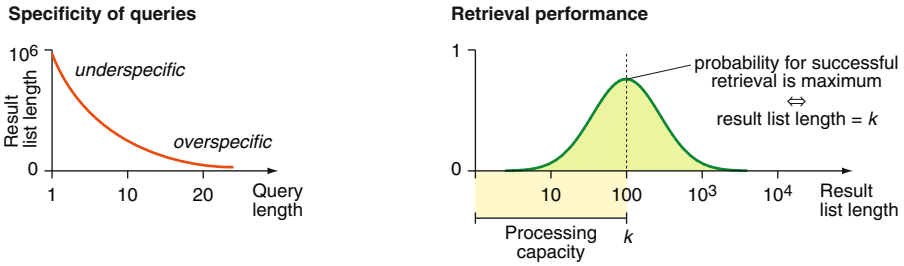
Readers who take this hypothesis for granted should recall the fact that virtually no user and none of the search index providers consider its implications. For readers who feel insecure about the claim, our paper gives empirical evidence.

## 1 Introduction

Web search is a standard information retrieval use case: information needs are satisfied with an indexed document collection by submitting keyword queries to a search engine and getting back ranked result lists. Typically, the user is neither given arbitrary access to the index nor has she knowledge about the engine’s underlying retrieval model, its implementation details, and the like: the search engine appears as a black box.

For many easy queries like `google` or `ebay` a user does not have any effort besides typing the query—current web search engines work so well that the desired item will pop up in the top results. However, the scenario we are considering here is that of an experienced user who has a more intricate information need. We assume that the user knows a whole bunch of keywords that, in her opinion, all tell something about her information need. However, if this entire set is submitted as a single query, it is likely that the returned result list contains very few or even no elements. On the other hand, if single-word queries are submitted, the obtained result list lengths will be in index size order of magnitude—a fact termed as the “million or none problem” [1].

We argue that very promising queries are the ones that return just a “reasonable” number of results. This is inspired by the observation that the number of results a user will consider from the entire result list is constrained by her processing capacity  $k$ , determined by the user’s reading time, her patience in browsing result lists, the available processing time, etc. *Underspecific* queries entail over-length result lists from which only a fraction—typically the top-ranked results—are processed at user site, whereas *overspecific* queries with only a handful of results usually do not help either. We argue that the probability to satisfy a user’s information need by exploring the top- $k$  results becomes maximum if the result list length is in the order of magnitude of the user’s processing capacity. The user then is still able to check all the results and can avoid any search engine ranking issues that she cannot influence. We term this argument *the-user-knows-better hypothesis* or, more formally, *User-over-Ranking hypothesis*. See Figure 1 for an idealized illustration of the outlined connections. Empirical justification for both parts of Figure 1 is provided in Sections 2 and 3. Potential application areas for the User-over-Ranking hypothesis and related work is presented in Section 4.



**Fig. 1.** The User-over-Ranking hypothesis consists of two parts: specificity and retrieval performance. The specificity part (left) takes up the folklore assumption that short (underspecific) queries return exponentially more results than long (overspecific) queries. The retrieval performance part (right) assumes that a user’s processing capacity  $k$  constrains the number of documents she will consider (we assume  $k \leq 100$ ). For longer result lists, the user typically just scans the top- $k$  documents. Shorter result lists often contain no relevant document at all. Under the User-over-Ranking hypothesis a result list length of  $k$  maximizes the probability to satisfy a given information need.

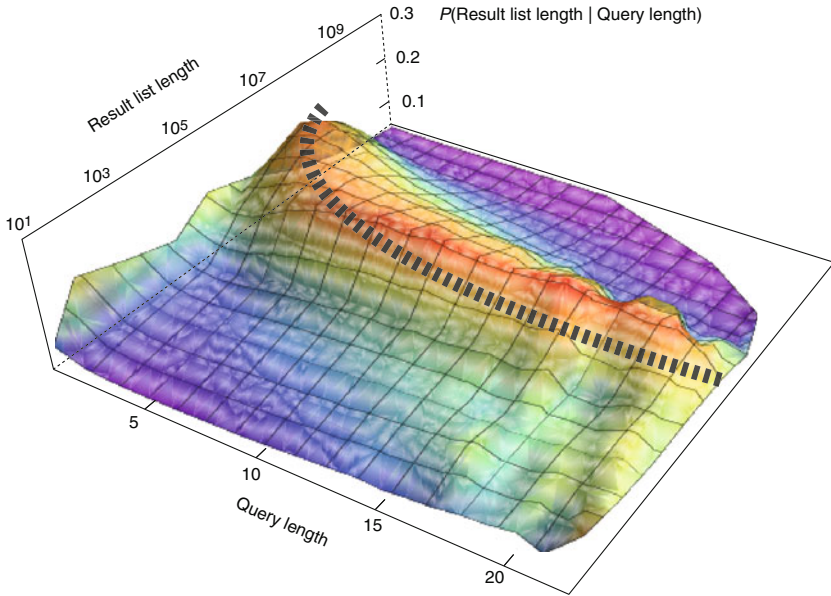
## 2 Specificity of Queries

In this section we empirically examine the specificity part of the User-over-Ranking hypothesis with real users’ web queries from a large search engine query log.

**Experimental Setup.** We use the AOL query log [9] that contains about 21 million web queries collected from about 650 000 AOL users over three months in 2006. A preprocessing removed the few users that could be considered as automatic bots (i.e., that submitted very many queries within very short time periods) as well as queries that just contain a URL. Such URL queries were probably submitted by users confusing the search box and the address bar of their browser. Finally, we eliminated query duplicates, and we restricted the query length to 22 keywords: there are too few queries with 23 or more keywords to draw reasonable conclusions. It remained 4 424 198 unique queries with an average length of 3.53 keywords. The distribution is as follows: about 300 000 single keyword queries, about 1 million each with 2-4 keywords, 0.5 million with 5 keywords, and then a steady decrease to about 100 queries with 22 keywords. All these queries were submitted to the Bing API during May 21-25, 2010, and Bing’s reported estimations on the number of results were stored for each query.

**Evaluation.** Note that the Bing API’s estimations are not unbounded but the largest result list length is  $2^{31}$  (the maximum `long`-value). As  $10^9 < 2^{31} < 10^{10}$ , we decided to use  $\log_{10}$ -discretization in order to have 10 bins for queries returning between  $10^{n-1}$  and  $10^n$  results for  $n = 1, \dots, 10$ . In Figure 2 we show the resulting distribution of query length and result list length. For every query length  $q$  we depict the relative portion of queries with length  $q$  that fall in the same  $\log_{10}$ -scaled result list bin.

One can observe a “ridge” in Figure 2 whose characteristic implies that on average queries with more keywords have an exponential decrease in the estimated result count. We further analyze this decrease by computing, for every query length  $q$ , the median



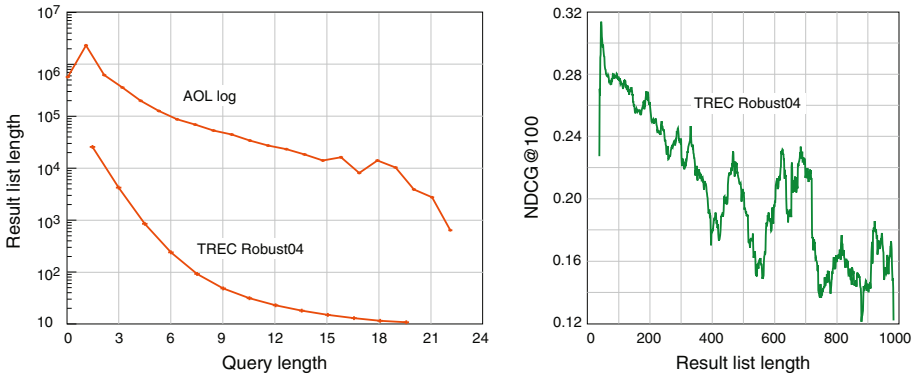
**Fig. 2.** 3D distribution of query length (in keywords) over result list length ( $\log_{10}$ -scaled) in the AOL query log sample. For each query length  $q = 1, \dots, 22$  the plot shows the relative portion of queries having length  $q$  that return a specific result list length ( $\log_{10}$ -scaled). The dashed line further highlights the location of the plot's “ridge”.

result list length of all queries of length  $q$ . The resulting medians clearly show an exponential decrease for larger  $q$  (see the AOL log plot in the left part of Figure 3 for a visualization). Using the Eureka tool<sup>1</sup> and excluding the “outlier” for single keyword queries we obtain  $f(q) = 8\,730\,000 \cdot q^{-2.29}$  as the corresponding decrease function. This gives experimental evidence for the specificity part of the User-over-ranking hypothesis and justifies the folklore assumption that short (underspecific) queries return exponentially longer result lists than long (overspecific) queries.

Finally, we briefly address the none-case of the million or none problem. For a closer consideration we treated all queries returning at most 10 results as queries representing the none-case. The reason is that some AOL log mirror pages exist on the web that just contain all the AOL log queries. Hence, hardly any query from the AOL log returns no results at all and result counts below 10 can be supposed to have had an empty result list during AOL log recording in 2006.

The relative portion of “no result”-queries we observe per query length is a little surprising. Although the portion increases for longer queries, it stays below 10% up to 21-keywords queries; increasing to 20% for 22-keywords queries. This observation is also supported by the median result list length that stays above 2 000 for queries of at most 21 keywords and drops to about 600 for 22 keywords. One reason for the

<sup>1</sup> <http://ccsl.mae.cornell.edu/eureka>



**Fig. 3. Left:** The median result list length over query length (in keywords) in the AOL log (cf. Section 2) and the average result list length over query length (in *keyphrases*) for the constructed phrase queries against TREC Robust04 (cf. Section 3). **Right:** Retrieval performance over result list length for the constructed phrase queries against TREC Robust04 (cf. Section 3). The NDCG@100-values are averaged over all queries that return a specific number of results. For smoothing purposes each data point is averaged with its 32 next neighbors to the left and to the right.

surprising behavior is that often the longer queries are verbose parts of song lyrics that still return a significant number of results.

### 3 Retrieval Performance

In this section we empirically examine how a query’s result list length influences retrieval performance. For this purpose we conduct a TREC style experiment. Our results show that queries with short but not too short result lists have a better retrieval performance than queries with longer result lists. This gives empirical evidence for the retrieval performance part of the User-over-Ranking hypothesis: the idea that a user should strive for queries that return about as many results as the user’s capacity is.

**Experimental Setup.** For our experiments we choose the TREC Robust04 corpus as it is used in several studies [3, 7, 8] evaluating long query reduction—an interesting potential application area for the User-over-Ranking hypothesis (cf. Section 4). The Robust04 corpus contains 528 155 newswire documents from Federal Register, LA Times, Foreign Broadcast Information Service, and Financial Times. There are the TREC topics 301–450 and 601–700 associated with this corpus. On average these 250 topics contain 2.66 words in the title field, 14.5 words in the description field, and 38.3 words in the narrative field; and on average have 1 246 associated Qrels (documents from the corpus with relevance judgments).

For our experiments we use the BM25 retrieval model and querying should be possible with complete phrases and not just words. Therefore, we indexed the Robust04 corpus with Terrier 3.0 using block-indexing with a block size of 1. To produce queries for each topic we segmented the topic titles into phrases using the naïve query segmentation method [5]. From the description and narrative fields we extracted keyphrases

using an implementation of the head noun extractor described in [2]. Especially for the narrative part we performed some manual cleaning after the automatic keyphrase extraction in order to remove some non-relevant phrases like “relevant document” etc. For each topic we introduced an ordering of the phrases according to their first appearance in the original topic (i.e., the first phrase from the title, the second phrase from the title, ..., the first phrase from the description, ..., and finally the phrases from the narrative part). From the phrase ordering we only use the first 15 phrases per topic when available. Using the described technique we obtain on average 9.5 phrases per topic. For example we get “whale watching california” “californian site” “whales” “habitat” “guide services” for topic 660.

The queries are built as follows. Phrases are combined to all possible sequences with respect to the phrase ordering. Hence, there is no sequence where the  $i$ -th phrase comes before a phrase  $j$  for  $j < i$ , but phrases could be left out (e.g., a sequence with the first and third phrase but not the second is possible). Each sequence is a distinct query such that there are  $2^{15} - 1$  possible queries for a topic with 15 phrases. All these queries were submitted to Terrier with highlighted phrases. For each query we stored the number of returned results and the first 1 000 documents when available. Queries returning at most 10 results were removed.

**Evaluation.** Our first observation supports the findings of the AOL experiments (cf. Section 2), namely, that there is an exponential decrease in the result list length for longer queries. The TREC Robust04 plot in the left part of Figure 3 shows the average result list length for our constructed phrase queries at each query length level.

However, the main focus is on the retrieval performance. As stated before, our assumption is that a typical user has a processing capacity  $k$  that constraints the number of results she will consider from the whole result list. Typically, these considered results will be the top ranked documents. We fix  $k = 100$  (i.e., assuming that a user will not check more than the first 100 results). As for evaluating retrieval performance we use NDCG to account for the ranking of the relevant documents. Hence, we measure NDCG@100 for the obtained Terrier result lists with the notion that for lists longer than 100 results the user will not skim further. The right part of Figure 3 shows the averaged NDCG@100 for all queries having a specific result list length for all topics together. Note that the right part of Figure 3 does not completely reflect the idealized retrieval performance plot of Figure 1. Nevertheless, it clearly supports the retrieval performance part of the User-over-Ranking hypothesis: the best performing queries are the ones that return about as many results as the assumed user’s capacity of checking at most 100 results.

## 4 Applicability and Related Work

One field where the User-over-Ranking hypothesis can be applied is that of long query reduction (i.e., handling queries with many keywords or even verbose text queries, similar to the description parts of TREC topics or queries to medical search engines). Research on long queries is on the rise [1, 3, 6, 7, 8], as a typical web query nowadays is becoming longer and longer, or, “more verbose.” Existing approaches reduce the long user query to a shorter keyword query by applying sophisticated strategies that try to

only focus on promising candidates of reduced queries. The User-over-Ranking hypothesis opens an interesting perspective from the user's site: promising candidates are the ones that return about as many results as the user can consider.

Using the User-over-Ranking hypothesis to identify promising queries cannot only be useful in reducing long queries but also for helping users that enter short and under-specific queries. Stein and Hagen [10] describe an automatic approach for helping users during search sessions. Their system combines a user's keywords from the short (under-specific) queries of a search session to a longer and more specific query. The User-over-Ranking hypothesis explains why the approach can improve user experience.

Also fully automatic query formulation systems can benefit from the hypothesis' insights. For example, the plagiarism detection system of [4] constructs web queries against commercial search engines from keywords extracted from a suspicious document. The aim is to retrieve web documents with similar content from which the suspicious document's author might have plagiarized. The User-over-Ranking hypothesis states that the constructed queries should be enlarged with additional keywords till the result list length is short enough for the detection system to handle it completely.

## 5 Conclusion and Outlook

In this paper we postulate the User-over-Ranking hypothesis and give empirical evidence for it. The hypothesis is split into two parts. The specificity part picks up the folklore assumption that longer (and thus more specific) queries return exponentially fewer results. However, the actual crux of the User-over-Ranking hypothesis is the second part on retrieval performance. It states that queries that return about as many results as the user can consider will maximize the probability of satisfying the user's information need. The straightforward conclusion from both parts then is that sufficiently long and specific queries are the best choice to query a search engine. Hence, the User-over-Ranking hypothesis provides an explanation for the plausible fact that users can help search engines by adding additional keywords to underspecific queries in order to obtain fewer and better results. Of course, the hypothesis is not meant to be applied for "easy" queries but rather as a model to explain the success of refined querying strategies in more elaborate information need scenarios.

A potential application area for the User-over-Ranking hypothesis is long query reduction. The conclusion from the User-over-Ranking hypothesis for a bad performing long verbose query is to reduce it to a keyword query that returns a reasonable number of hits. Developing a corresponding long query reduction approach is an interesting task for future work.

## References

- [1] Balasubramanian, N., Kumaran, G., Carvalho, V.R.: Exploring Reductions for Long Web Queries. In: Proceedings of SIGIR 2010, pp. 571–578 (2010)
- [2] Barker, K., Cornacchia, N.: Using Noun Phrase Heads to Extract Document Keyphrases. In: Proceedings of AI 2000, pp. 40–52 (2000)
- [3] Bendersky, M., Croft, W.B.: Discovering Key Concepts in Verbose Queries. In: Proceedings of SIGIR 2008, pp. 491–498 (2008)

- [4] Hagen, M., Stein, B.: Capacity-constrained Query Formulation. In: Proceedings of ECDL 2010, pp. 384–388 (2010)
- [5] Hagen, M., Potthast, M., Stein, B., Bräutigam, C.: The Power of Naïve Query Segmentation. In: Proceedings of SIGIR 2010, pp. 797–798 (2010)
- [6] Huston, S., Croft, W.B.: Evaluating Verbose Query Processing Techniques. In: Proceedings of SIGIR 2010, pp. 291–298 (2010)
- [7] Kumaran, G., Allan, J.: Adapting Information Retrieval Systems to User Queries. *Information Processing and Management* 44(6), 1838–1862 (2008)
- [8] Lease, M., Allan, J., Croft, W.B.: Regression Rank: Learning to Meet the Opportunity of Descriptive Queries. In: Proceedings of ECIR 2009, pp. 90–101 (2009)
- [9] Pass, G., Chowdhury, A., Torgeson, C.: A Picture of Search. In: Proceedings of Infoscale 2006, article number: 1 (2006)
- [10] Stein, B., Hagen, M.: Making the Most of a Web Search Session. In: Proceedings of WI-IAT 2010, pp. 90–97 (2010)
- [11] Tunkelang, D.: *Faceted Search*. Morgan & Claypool Publishers (2009)

# Second Chance: A Hybrid Approach for Dynamic Result Caching in Search Engines

I. Sengor Altingovde<sup>1</sup>, Rifat Ozcan<sup>1</sup>,  
B. Barla Cambazoglu<sup>2</sup>, and Özgür Ulusoy<sup>1</sup>

<sup>1</sup> Department of Computer Engineering, Bilkent University, Ankara, Turkey  
{ismaila,rozcan,oulusoy}@cs.bilkent.edu.tr

<sup>2</sup> Yahoo! Research, Barcelona, Spain  
barla@yahoo-inc.com

**Abstract.** Result caches are vital for efficiency of search engines. In this work, we propose a novel caching strategy in which a dynamic result cache is split into two layers: an HTML cache and a docID cache. The HTML cache in the first layer stores the result pages computed for queries. The docID cache in the second layer stores ids of documents in search results. Experiments under various scenarios show that, in terms of average query processing time, this hybrid caching approach outperforms the traditional approach, which relies only on the HTML cache.

**Keywords:** Search engines, query processing, result cache.

## 1 Introduction

Result caching is a crucial mechanism employed in search engines to satisfy low response time and high throughput requirements under high query workloads [2]. Usually, a static result cache is filled by the result pages of queries that were frequent in the past. Additionally, a dynamic result cache is maintained to handle the burst in query traffic. The content of the result cache changes dynamically depending on the query stream. Each time the cache is full, an entry is evicted from the cache based on a certain replacement policy (e.g., LRU). A real life search engine might either split the available cache capacity between static and dynamic caches [3], or involve a sufficiently large dynamic cache that would almost never evict frequent queries, as if they were kept in a static cache.

In design and evaluation of caching strategies, a traditionally used measure is the cache hit rate. Recently, some works have also taken into account the fact that the cost of a cache miss depends on the query, i.e., some queries require more computational resources to be answered [14]. These works have shown that it is better to tune a caching strategy according to the query processing cost incurred on the backend system, instead of the achieved hit rate alone.

A typical entry in a dynamic result cache stores the HTML result page<sup>1</sup> generated as an answer to a query. A storage-wise profitable alternative to this

---

<sup>1</sup> By HTML result page, we mean the textual content such as the URLs and snippets of the documents in the result page [3], but not the visual content in the page.



---

**Algorithm 1.** The second chance caching algorithm

---

**Require:**  $q$ : query,  $H$ : HTML cache,  $D$ : docID cache

```

1:  $R_q \leftarrow \emptyset$  ▷ initialize the result set of  $q$ 
2: if  $q \notin H$  and  $q \notin D$  then
3:   evaluate  $q$  over the backend and obtain  $R_q \triangleright C_q = C_q^{\text{list}} + C_q^{\text{rank}} + C_q^{\text{doc}} + C_q^{\text{snip}}$ 
4:   insert  $R_q$  into  $H$  and  $D$ 
5: else if  $q \in H$  then
6:   get  $R_q$  from  $H$ 
7:   update statistics of  $q$  in both  $H$  and  $D \triangleright C_q = 0$ 
8: else if  $q \in D$  then
9:   get doc ids from  $D$  and compute snippets to obtain  $R_q \triangleright C_q = C_q^{\text{doc}} + C_q^{\text{snip}}$ 
10:  insert  $R_q$  into  $H$ 
11:  update statistics of  $q$  in  $D$ 
12: end if
13: return  $R_q$ 

```

---

is to store only the ids of the documents in the result page (possibly, together with their scores) [3]. Given the same amount of space, a docID cache can store entries for a larger number of queries than the HTML cache, and hence it can yield a higher hit rate. However, since the snippets had to be computed for the matching documents, average query processing times are higher relative to the HTML cache. In this respect, the information provided by the HTML cache is complete and ready-to-serve, whereas it is incomplete and requires further computation (i.e., snippet computations) in case of the docID cache.

In this study, we propose to split a dynamic result cache into two layers, namely HTML and docID caches, and introduce the so-called second chance caching strategy. The basic intuition behind our strategy is that, since a docID result for a query takes significantly less storage space than an HTML result, even if the HTML result is evicted from the cache, the docID result may still remain. The trade-off is simple: we reserve a relatively small space for complete HTML results and store incomplete results for a large number of queries that would, otherwise, be evicted. For those queries stored in the docID cache, we introduce snippet computation overhead. However, for potentially many queries, we avoid the more expensive scoring cost, completely.

The rest of the paper is as follows. Our caching strategy is presented in Sec. 2. In Sec. 3, we describe a detailed cost model for evaluating this strategy. Section 4 provides experimental results, showing performance improvements under various scenarios. The paper ends with the concluding discussion in Sec. 5.

## 2 Second Chance Caching Strategy

The main idea of our strategy is to divide the cache into two layers as HTML and docID caches. In Algorithm 1, we provide the basic outline of our caching strategy. Whenever a query  $q$  leads to a miss in both the HTML and docID caches, its result is computed at the backend search system and inserted into both caches (lines 3–4). As long as  $q$  is found in the HTML cache, its results are

served by this cache and its statistics are updated (lines 6–7). At some point in time,  $q$  may become the LRU item. In this case, it is discarded from the HTML cache, but its (incomplete) results still reside in the docID cache. In some sense, this approach gives a second chance to  $q$ . If the query is ever repeated soon, it becomes a hit in the docID cache. In this case, the backend system computes only the snippets to create the HTML result page, which is both sent to the user and inserted into the HTML cache (lines 9–11). Note that each case in the algorithm is associated with a cost ( $C_q$ ), which we will discuss next.

### 3 Cost Model and Scenarios

Processing of a query  $q$  in a search engine has four main steps: (i) fetching posting lists for query terms from disk ( $C_q^{\text{list}}$ ), (ii) decompressing lists and computing the top  $k$  results ( $C_q^{\text{rank}}$ ), (iii) fetching  $k$  result documents from disk ( $C_q^{\text{doc}}$ ), and (iv) computing snippets for fetched documents ( $C_q^{\text{snip}}$ ). Costs incurred in case of a miss, an HTML cache hit, and a docID cache hit are given in Algorithm 1.

In practice, a search cluster in a commercial search engine is made up of hundreds of nodes that store a part of the inverted index and document collection.<sup>2</sup> This means that steps (i) and (ii) are executed on all nodes in the cluster. Then, the partial results for the query are sent to the broker node, which merges them and computes the final top  $k$  document ids. Finally, these documents are accessed to compute snippets (steps (iii) and (iv)) and generate the HTML result page. The cost of transferring and merging partial results is mostly negligible.

Under the above cost model, we further consider two key issues: 1) caching of posting lists as well as documents and 2) assignment of documents to search nodes. Regarding the first issue, it is known that search engines cache posting lists and documents, in addition to search results. In case of a cache miss, steps (i) and (iii) require disk accesses. To this end, we consider two different scenarios for caching the latter two types of data: A “full caching” scenario, where all lists and documents are kept in the memory, and a more conservative “moderate caching” scenario, where only 50% of each item type is cached.

The second issue we consider is the distribution of the snippet computation overhead on the nodes of a search cluster with  $K$  nodes. This distribution affects how the costs in steps (iii) and (iv) are computed. We again consider two basic scenarios: In the “random assignment” scenario, we assume that documents in the collection are randomly assigned to cluster nodes, as usual. For  $K \gg k$ , it is very likely that every document in the top  $k$  result resides on a different node. Then, document access and snippet generation take place on each node in parallel and, in effect, the total processing cost is almost equal to the cost of executing steps (iii) and (iv) for only one document (i.e., as if  $k=1$ ). In the “clustered assignment” case, we assume that documents are assigned to nodes based on, say, topic. In the worst case, all top  $k$  documents reside in the same node. Hence, the costs of steps (iii) and (iv) are incurred for all  $k$  documents.

<sup>2</sup> The result cache is maintained in the broker node.

## 4 Experiments

We use a collection of 2.2 million web pages obtained from the ODP web directory (<http://www.dmoz.org>). The index file includes only document ids and term frequencies, and it is compressed with the Elias- $\delta$  encoding scheme. We sample from the AOL query log [5] 16.8 million queries, whose all terms appear in our collection. Queries are processed in timestamp order. First 8M queries are used as the training set and the remaining 8.8M queries are used as the test set. Training and test sets include 3.5M and 3.8M unique queries, respectively.

Training queries are used for two purposes. First, frequent queries are used to warm up the HTML and docID caches. Second, for the “moderate caching” scenario, we use these queries to populate the posting list and document caches. To decide on the terms to be cached, we use the popularity/size metric [2], where the former is the term’s frequency in the training query log and the latter is the size of its posting list. While caching documents, we process training queries over the collection and select the documents that are most frequent in top 10 search results. In both cases, the items are cached up to the 50% capacity limit.

In Tables 1 and 2, we provide the parameters and cost formulas used, respectively. Decompression and scoring times are experimentally obtained over our dataset. For snippet computation time, we assume a simple method (each query term in the document along with a few neighboring terms are added to the snippet) and set this value to a fraction of the query processing time.

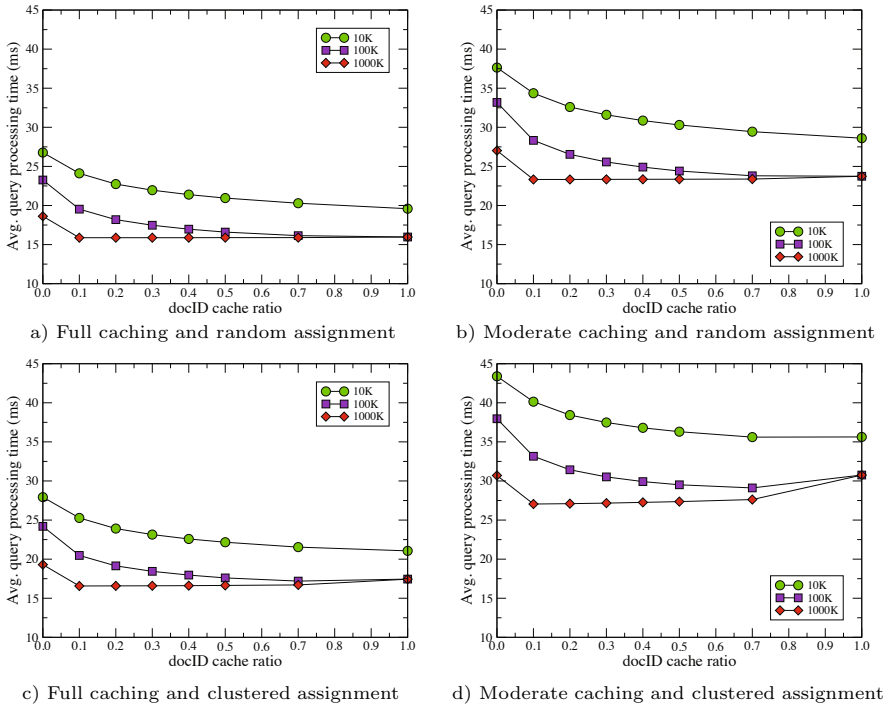
In Table 1, we specify the ratio ( $S$ ) between the sizes of a result item in the HTML cache and that in the docID cache because all experiments reported below specify the cache capacity in terms of the number of HTML result pages that can fit into the cache. Hence, our findings are valid as long as the ratio among the result item sizes is preserved, regardless of the actual values. Assuming that

**Table 1.** Parameters

Parameter	Value	Parameter	Value
Size ratio of cache items ( $S$ )	64	Number of results per query ( $k$ )	10
Disk latency ( $D_\ell$ )	12.7 ms	Decompression per posting ( $P_d$ )	100 ns
Disk block read ( $D_{br}$ )	4.9 $\mu$ s	Ranking per posting ( $P_r$ )	200 ns
Disk block size ( $D_{bs}$ )	512 bytes	Snippet computation per byte ( $P_s$ )	10 ns

**Table 2.** Cost formulas ( $t$  denotes a term in  $q$ ,  $I_t$  denotes the inverted list of  $t$ , and  $d$  denotes a document in the query result set  $R_q$ )

Cost	Formula	Description
$C_q^{\text{list}}$	$\sum_{t \in q} (D_\ell + (D_{br} \times  I_t  / D_{bs}))$	Fetching of posting lists from disk
$C_q^{\text{rank}}$	$\sum_{t \in q} ( I_t  \times (P_d + P_r))$	Score computations during ranking
$C_q^{\text{doc}}$	$\sum_{d \in R_q} (D_\ell + (D_{br} \times  d  / D_{bs}))$	Fetching of documents from disk
$C_q^{\text{snip}}$	$\sum_{d \in R_q} ( d  \times P_s)$	Snippet computations



**Fig. 1.** Performance of the hybrid result cache under different scenarios. In all figures, the docID cache ratio of 0 corresponds to the pure HTML cache, which is our baseline.

a single document id may take around 4 bytes and a result URL and snippet (of 20 terms) may take around 256 bytes, we set this ratio to 64.

We compare the performance of our strategy for varying cache split ratios, which range from the pure HTML cache (the ratio is 0) to the pure docID cache (the ratio is 1) over the test query set. We experiment with three different cache capacities that are representatives of small (10K), medium (100K), and large (1000K) caches for our query log. We express the cache size in terms of the number of HTML result pages that can fit into the cache, e.g., the 1000K cache can store 1000K pages. The largest cache capacity corresponds to about 11% of all test queries and 26% of the unique test queries. As mentioned before, for the random and clustered assignment scenarios, we set  $k$  to 1 and 10, respectively.

In Fig. 1, we consider four different combinations of our scenarios. Since our strategy (by definition) improves the hit rate of the baseline HTML cache and the trade-off is in terms of the incurred system costs, we report only query processing times. Our findings are as follows: (i) For all cases, we can identify docID cache ratios that yield better performance than the baseline, i.e., the pure HTML cache. In other words, using a hybrid cache always reduces the average query processing time with respect to the baseline. (ii) For cache sizes of 10K and 100K, it is more efficient to devote the majority (i.e., greater than 70%) or even all of the cache space to the docID cache. This implies that for these

cases, most of the frequent queries cannot stay long enough in the pure HTML cache. For these cache sizes, reductions in average query processing time reach up to 31%. (iii) For the largest cache size, we observe that reserving 10% of the capacity to the docID cache yields the best performance. This indicates that, as the available cache size increases, the gain provided by the docID cache decreases, as it would be possible to store more results in the HTML cache. Still, the relative reductions provided by the second chance caching strategy are 15% and 14% for the full and moderate caching cases with random assignment of documents, respectively. Achieved reductions are slightly lower for the clustered assignment of documents (i.e., 14% and 12% for full and moderate caching, respectively). The setup with the highest query processing cost (i.e., moderate caching with clustered document assignment) yields the lowest reduction (12%).

## 5 Concluding Discussion

Our strategy provides significant advantages when the result cache capacity is limited. In the experiments, we showed that our strategy yields 15% reduction in query processing time even when the cache is large enough to store 26% of all unique queries in the test set. Clearly, as the result cache gets larger, benefits of our strategy diminish. At one extreme, if the search system has enough resources to cache the results of all non-singleton queries for a reasonably long time, the hybrid cache is rendered useless. However, given the current query workloads of search engines, such a solution may require large amounts of storage, summing up to an infeasible financial value. Our solution, on the other hand, is a compromise for providing better utilization on a limited-memory result cache.

Our caching strategy exploits the fact that the snippet generation cost is a fraction of the total query processing cost. To best of our knowledge, there is no work that explicitly compares the cost of snippet generation to other costs, such as decompression, list intersection, and ranking. Nevertheless, to investigate the sensitivity of our caching strategy to snippet generation cost, we repeated our experiments with higher  $P_s$  values (100 ns and 1000 ns per byte). For the full and moderate caching scenarios with random document assignment, the best docID cache ratios still reduce query processing times by 14% and 12%, for the 100 ns case, and by 10% and 5%, for the 1000 ns case, respectively.

This work is a first step for investigating the performance of a hybrid dynamic caching strategy for search engines in a cost-based framework. In the future, we plan to integrate result pre-fetching into our strategy.

## References

1. Altıngövdü, I., Özcan, R., Ulusoy, O.: A cost-aware strategy for query result caching in web search engines. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 628–636. Springer, Heidelberg (2009)

2. Baeza-Yates, R., Gionis, A., Junqueira, F., Murdock, V., Plachouras, V., Silvestri, F.: The impact of caching on search engines. In: Proc. 30th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 183–190 (2007)
3. Fagni, T., Perego, R., Silvestri, F., Orlando, S.: Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data. *ACM Trans. Inf. Syst.* 24(1), 51–78 (2006)
4. Gan, Q., Suel, T.: Improved techniques for result caching in web search engines. In: Proc. 18th Int'l Conf. on World Wide Web, pp. 431–440 (2009)
5. Pass, G., Chowdhury, A., Torgeson, C.: A picture of search. In: Proc. 1st Int'l Conf. on Scalable Information Systems, p. 1 (2006)

# Learning Models for Ranking Aggregates

Craig Macdonald and Iadh Ounis

School of Computing Science, University of Glasgow,  
Glasgow G12 8QQ, UK

{craig.macdonald,iadh.ounis}@glasgow.ac.uk

**Abstract.** Aggregate ranking tasks are those where documents are not the final ranking outcome, but instead an intermediary component. For instance, in expert search, a ranking of candidate persons with relevant expertise to a query is generated after consideration of a document ranking. Many models exist for aggregate ranking tasks, however obtaining an effective and robust setting for different aggregate ranking tasks is difficult to achieve. In this work, we propose a novel learned approach to aggregate ranking, which combines different document ranking features as well as aggregate ranking approaches. We experiment with our proposed approach using two TREC test collections for expert and blog search. Our experimental results attest the effectiveness and robustness of a learned model for aggregate ranking across different settings.

## 1 Introduction

Identifying expert persons in an organisation, key bloggers for a topic on the blogosphere and finding related entities on the Web are all examples of aggregate ranking tasks, where objects - e.g. people - are represented by sets of documents that must be ranked in response to a query. Various models have been proposed for aggregate ranking [1,2,3]. However, while each model might perform well in a particular setting, it might not adapt well to another aggregate ranking task. For instance, the Model 2 approach [1] performs well for expert search, but less so for identifying key bloggers [4]. In this work, we investigate how to learn effective and robust aggregate ranking models using the learning to rank paradigm [5].

In learning to rank, features are normally defined on the objects being evaluated, i.e. document features which are then evaluated directly. However, in aggregate ranking tasks, the usefulness of document features is difficult to assess in a learning to rank framework, as relevance assessments are only defined on the aggregates. We propose a novel methodology for applying learning to rank to the ranking of document aggregates. In this methodology, features are defined in terms of three independent variables, namely the document ranking, the rank at which the document ranking is truncated, and the aggregate ranking strategy used to convert the document ranking into a ranking of aggregates. We evaluate the proposed methodology using standard TREC test collections for two aggregate ranking tasks, namely expert and blog search. Our experiments analyse the impact of each of the independent variables on the effectiveness of the learned

models. The results show that effective and robust learned models for aggregate ranking can be obtained using our proposed methodology.

To the best of our knowledge, our work is the first framework showing how to apply learning to rank techniques to aggregate ranking. In particular, we conduct an in-depth study into learning models for two aggregate ranking task. The remainder of this paper is structured as follows: Section 2 reviews existing approaches for ranking aggregates, and discusses their limitations, before introducing the learning to rank paradigm for information retrieval; Section 3 defines our methodology for learning to rank in aggregate ranking tasks; Section 4 details our research questions and experimental setup, while the experimental results and analysis follow in Section 5; Finally, Section 6 compares our work to other recent learned approaches for aggregate ranking, and Section 7 provides concluding remarks.

## 2 Aggregate Ranking towards Learning to Rank

### 2.1 Aggregate Ranking

There are several well-known approaches for aggregate ranking spawned by research in the expert and blog search tasks. Typically, all approaches use profiles of documents to represent each *candidate object*. However, they differ in the way in which the profiles are ranked in response to a query. For instance, Balog et al. [1] proposed the Model 1 and Model 2 language modelling approaches for expert search. Similarly, Elsas et al. [3] proposed the “large document” and “small document” language models in the context of blog search, which correspond to Models 1 & 2, respectively. Macdonald & Ounis took a different approach to expert ranking - in their Voting Model [2], a document ranking provides votes to associated objects (e.g. experts) to be retrieved for the query. Twelve different voting techniques were proposed that define different ways in which the votes can be aggregated. In particular, the CombSUM voting technique is similar to Model 2, but agnostic to the particular document ranking technique applied. Different voting techniques have since been shown to be effective at finding key bloggers [6] and related entities from the Web [7].

While these various aggregate ranking approaches may be suitable for various tasks or datasets, an effective setting for one aggregate ranking task may not perform well on another task. For instance, while Balog et al. found Model 2 to be more effective than Model 1 for expert search [1], the opposite was found to be true for finding key bloggers [4]. In contrast, Elsas et al. [3] found the small document model (which corresponds to Model 2 of Balog et al.) to be more effective than the large document model (c.f. Model 1) for key blog identification. By proposing various different voting techniques, Macdonald & Ounis [2] acknowledged that different voting techniques may be suitable for different tasks or settings. For example, while the CombMAX voting technique can be effective for some expert search tasks [8], it was found to be less effective for finding key blogs [6]. Moreover, the voting techniques can consider any number of top-ranked



documents from the underlying document ranking. However, the most effective rank cutoff can vary between tasks and collections [8].

Due to these difficulties in finding consistently effective and robust settings for aggregate ranking approaches, in this work, we propose instead to learn an aggregate ranking model, which is robust and effective across different tasks and settings. In particular, the voting techniques of the Voting Model provide various different aggregate ranking strategies that are effective for different tasks or collections. We hypothesise that it is possible to learn an appropriate and effective combination of voting technique strategies using learning to rank techniques. In the remainder of this section, we provide background on learning to rank techniques, and the challenges incurred in their application to aggregate ranking. This is followed in Section 3 by our proposed methodology for learning an effective aggregate ranking model.

## 2.2 Learning to Rank

Learning to rank describes the application of machine learning techniques to select weights for different document features in an information retrieval (IR) system [5]. For instance, learning to rank techniques are often applied by Web search engines, to combine various document weighting models and other query-independent features [9]. The various learning to rank approaches in the literature fall into one of three categories, namely pointwise (learn relevance independently of other documents), pairwise (optimise the number of pairs of documents correctly ranked) and listwise (optimise an information retrieval evaluation measure that considers the entire ranking list). In this work, we consider two listwise approaches that directly evaluate with respect to the target IR evaluation measure, instead of the evaluation approximations that are used by pointwise or pairwise approaches. Moreover, listwise techniques have been shown to learn more effective models [5]. To examine the impact of different learning to rank techniques on the effectiveness of the learned models, we deploy two listwise techniques, namely Metzler's Automatic Feature Selection algorithm (AFS) [10], and AdaRank [11]. Both AFS and AdaRank take a greedy approach to feature selection, by iteratively selecting the feature that most improves retrieval performance in combination with the previously selected features. Features that are not beneficial to the retrieval performance on the training set will not be selected. However, while AFS finds the optimal weight for each feature, AdaRank calculates feature weights based on their boosted performance on the training queries [11]. In practice, this makes AdaRank considerably faster than AFS. The general steps to learn a ranking *model* are as follows [5]:

1. Generate a sample of training documents using an initial retrieval approach.
2. Extract all features for all of the documents in the sample. A feature is a numerical indicator thought to be of use in a learned model.
3. Learn a ranking model through the application of a learning to rank approach.
4. Apply the learned model on a sample of test documents with the same features.

It is of note that the strategy used to create the sample of documents to re-rank impacts on the effectiveness of the learned model. In [5], Liu states that for the LETOR learning to rank datasets, the top 1000 documents are sampled using BM25 on the content alone. However, Liu notes that while this method of producing the sample is sufficient, it may not be the best [5]. Indeed, if the sample has insufficient recall, then the scope for the learning to rank approach to generate a quality ranking will be hindered. In Section 4, we describe how an appropriate sample of aggregate objects is created for our learning approach.

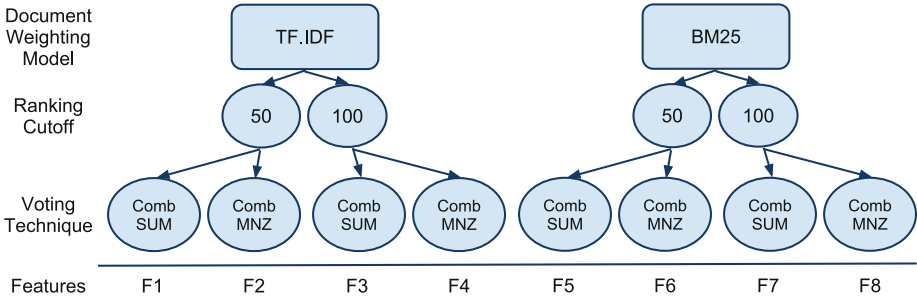
A salient point of learning to rank is that the features are defined on the objects being evaluated, i.e. document features which are then evaluated directly. However, in aggregate ranking tasks, the usefulness of document features is difficult to assess in a learning to rank framework, since such features are not directly defined on the aggregates, and relevance assessments are only defined on the aggregates. To tackle this problem, in the next section, we propose a novel methodology for applying learning to rank for ranking document aggregates.

### 3 Learning to Rank Aggregates

In aggregate search tasks, the goal is to rank objects such as people, entities or blogs, where each aggregate object is defined as a set of documents. However, as mentioned above, a key complication of learning to rank aggregates is that many features (e.g. uni- and bi-gram weighting models, PageRank, to name a few) are defined at the level of documents, rather than at the level of aggregate objects. However, when using learning to rank, the features must be defined at the same level as the relevance assessments, i.e. at the aggregate level.

To take such document features into account, it could be intuitive to use a learned document ranking as input to an aggregate ranking strategy. However, two factors combine to make such an approach not viable. Firstly, in aggregate ranking tasks, the relevance assessments are defined at the aggregate level, therefore there is no easy way to learn an effective ranking. Secondly, even when there are document level relevance assessments for the same queries as the aggregate ranking task (e.g. TREC 2007 and 2008 Enterprise tracks, for the document search and expert search tasks [12,13]), it has been shown that increasing the quality of the document ranking does not always result in increasing the effectiveness of the resulting ranking of candidate experts [14]. Moreover, Macdonald & Ounis [15] showed that when perfect document rankings (e.g. MAP 1.0) are applied, the resulting candidate rankings were further degraded. Such counter-intuitive results can be explained in that the document relevance assessments measure different properties of the document ranking than those desirable for an effective ranking of candidates [14], and suggest that the direct learning of document ranking features for use to rank aggregates is not a viable strategy.

Instead of trying to directly learn a document ranking, we propose to work with features directly defined at the level of the aggregate objects. Firstly, a sample of aggregate objects to be re-ranked is defined, using a single effective aggregate ranking strategy (c.f. BM25 used by LETOR [5]). Then, we propose



**Fig. 1.** Eight features obtained from two document ranking models (TF.IDF & BM25), two rank cutoffs (50 and 100) and two voting techniques (CombSUM & CombMNZ)

that each possible aggregate ranking strategy is a feature, defined on the objects in the sample. In particular, in this work, as mentioned in Section 2.1, we apply different instantiations of the Voting Model. Indeed, the Voting Model defines many aggregate ranking strategies as voting techniques, each of which can use various document rankings with different rank cutoffs. Hence, each different voting technique that is applied to the same document ranking represents a different feature. Similarly, if the document ranking is changed, or truncated at a different rank cutoff, then a new feature is defined. Figure 1 shows an example feature set, where two document ranking models (TF.IDF and BM25), truncated at two rank cutoffs (50 and 100) are combined with two voting techniques from 2 (CombSUM and CombMNZ), to make a total of eight features. From Figure 1, it is clear that a single feature represents a path through the different levels of the tree, where the document ranking model, the cutoff and the voting technique are three independent variables of the feature. More formally, a single feature  $f$  for an object  $O$  for query  $Q$  is defined as:

$$f(O, Q, DM, \theta, VT) = VT(O, Tr(DM(Q), \theta)) \quad (1)$$

where  $VT(O, Tr(DM(Q), \theta))$  is the score for object  $O$  according to a particular voting technique, operating on a ranking of documents returned by a ranking model  $DM$  for query  $Q$ .  $Tr(DM(Q), \theta)$  truncates document ranking  $DM(Q)$  to the top  $\theta$  ranked documents. In this formulation of aggregate learning, there are three levels. Each level corresponds to one of the independent variables in Equation (1), namely  $DM$ ,  $\theta$  or  $VT$ , and the particular values for each variable define the exact feature generated. For example, in Figure 1, eight features are generated by  $DM = \{DPH, BM25\}$ ,  $\theta = \{50, 100\}$ , and two voting techniques  $VT = \{\text{CombSUM}, \text{CombMNZ}\}$ .

Once many features have been extracted for the sampled candidates, a learning to rank technique can be applied. The outcome of the learning to rank technique is a weighted linear combination of features, such that a new aggregate ranking strategy is created, consisting of an *ensemble* of various voting techniques using different document rankings and cutoffs. Moreover, as with learning to rank applied on documents, the success of a learned approach depends on the number

and usefulness of the features. As will be shown in the next section, we vary the three independent variables, to generate a large number of features for learning to rank aggregates. In this regard, the Voting Model is particularly suitable, as it defines many voting techniques for the independent variable  $VT$ . Moreover, it is agnostic to the choice of the document ranking  $DM$  and the ranking cutoff  $\theta$ . In essence, this allows a combinatorial approach to feature generation by varying the instantiations of each independent variable.

Finally, in this work, our experiments only use features based on query-dependent document ranking models. However, query-independent features could also be handled within our proposed methodology, by defining them on a sample of documents selected by a query-dependent document ranking model. e.g. using PageRank as a document ranking, but defined on documents ranked by BM25.

## 4 Experimental Setup

In the proposed methodology for learning aggregate models using many features, each feature is generated by different instantiations of the  $DM$ ,  $\theta$  and  $VT$  independent variables. In our experiments, we investigate the importance of each independent variable, by varying one while holding all other variables constant, to create *groups* of features. We then ascertain the importance of each of these feature groups, by addressing the following research questions:

- (1). Does using more than one voting technique benefit retrieval performance?
- (2). Does using more than one document cutoff benefit retrieval performance?
- (3). Does using more than one document ranking benefit retrieval performance?
- (4). Finally, what are the most important features for each of the investigated retrieval tasks?

The remainder of this section defines the experimental setup to address these research questions. In particular, Section 4.1 details the selected test collections and the adopted training regime, while Section 4.2 details the generated features.

### 4.1 Tasks and Training

We address the above research questions using two aggregate ranking search tasks, namely expert search and blog distillation. In the expert search task, candidate experts within an enterprise organisation are aggregate objects that must be ranked in response to a query. In particular, the expertise of each candidate is represented by a profile of intranet documents containing their name or email address. We use the TREC Enterprise track 2007 and 2008 expert search task test collections [12,13] - denoted EX:07 and EX:08, respectively. Both tasks are based on the CERC corpus of intranet documents. In the blog distillation task, key blog(ger)s that have a recurring interest in a query topic should be identified. In this task, each blog is an aggregate, consisting of all of the blog's postings. In particular, we use the TREC Blog track 2007 and 2008 blog distillation test collections [16,17], denoted BD:07 and BD:08, respectively, both of which use the Blogs06 corpus. Statistics of the used test collections are presented in Table 1.

**Table 1.** Tasks and test collections used

	EX:07	EX:08	BD:07	BD:08
Corpus	CERC		Blogs06	
Number of Documents	370K		3M	
Number of Candidates	3.4K		100K	
Mean Profile Size	68.2		31.94	
Number of Topics	50	55	45	50

We deploy the two listwise learning to rank techniques described in Section 2.2, namely AFS and AdaRank. Moreover, applying a learning to rank technique requires enough training data to successfully learn the weights of the features, as well as sufficient test data to adequately evaluate the learned models. The selected test collections are the only aggregate ranking test collections currently available with more than 50 topics, sufficient for both training and testing. In contrast, the LETOR datasets for evaluating learning to rank approaches [5] do not address aggregate ranking tasks. While the TREC 2009 Blog track faceted blog distillation task and the TREC 2009 Entity track related entity finding task are also aggregate ranking tasks, they do not have enough topics (39 and 20, respectively) for successfully applying learning to rank techniques.

In our experiments, we apply an appropriate training regime whereby results are reported on different topics from the training topics, but within the same corpus. Hence, a clear separation between training and testing topics is enforced. For instance, we train on EX:07 topics and test on EX:08, and vice versa. Note that this training regime makes our results perfectly comparable to the participating systems of the EX:08 and BD:08 TREC tasks only. However, for the EX:07 and BD:07 topics, we are using training data that was not available to the TREC participants of that year. Mean Average Precision (MAP) is used as both the training measure during learning, and the measure reported in the results.

## 4.2 Feature Generation

For both the expert search and blog distillation tasks, we use various instances for the variables  $DM$ ,  $\theta$  and  $VT$  to generate different features. In general, to permit an impartial cross-comparison of selected features across expert search and blog distillation tasks as per research question (4), we adopt a uniform setting between both tasks, in that only techniques that are applicable to both tasks are applied. Table 2 details the instantiations of the  $DM$ ,  $\theta$  and  $VT$  independent variables. In particular, the DPH document weighting model [18] is applied, with and without proximity [19] or collection enrichment [20], on either document content or the corresponding anchor text of the incoming hyperlinks. Nine different document ranking cutoffs are applied, along with ten different voting techniques from the Voting Model [2]. Five of these voting techniques apply profile length normalisation, which often increases effectiveness by preventing aggregate objects with many associated documents from being over emphasised in the final ranking [8]. The product of all of the above possible instantiations

**Table 2.** Applied instantiations of each independent variable. A total of 540 aggregate level features are generated.

Variable	#	Description
$DM$	6	DPH document weighting model [18] on either content or anchor text, with and without query term proximity [19]. Collection enrichment using Wikipedia as per [20], applied on either content or anchor text.
$\theta$	9	Ranking cutoffs: $\theta = \{50, 100, 250, 500, 1000, 2000, 3000, 4000, 5000\}$ .
$VT$	10	CombSUM, CombMNZ, CombMAX, expCombSUM, expCombMNZ from the Voting Model, as per [26]. CombSUMNorm1D, CombMNZNorm1D, CombMAXNorm1D, expCombSUMNorm1D, expCombMNZNorm1D adds profile length normalisation [8].

of variables  $DM$ ,  $\theta$  and  $VT$  is a total of  $6 \times 9 \times 10 = 540$  features that can be considered by the two learning to rank techniques used in our experiments.

Lastly, we consider the generation of the sample of objects to re-rank. As discussed in Section 2.2, for the LETOR datasets, Liu suggests that selecting the top 1000 ranked BM25 documents produces a sample of sufficient recall [5]. Similarly, in this work, our sample consists of the top 200 aggregate objects ranked by the CombSUM voting technique [2], using a DPH document ranking cutoff at rank 1000. In particular, DPH represents an effective parameter-free document weighting model [18], while CombSUM was shown to be effective for both expert search and blog distillation [28], and is similar to the Model 2 language modelling approach [1]. By using a sample with depth 200, we have a good recall from which to re-rank objects, as the testing evaluation is limited to rank 100 as per the TREC setting of the expert search and blog distillation tasks.

## 5 Experimental Results

In this section, we experiment to address each of the research questions described in Section 4 in turn. In particular, in Sections 5.1, 5.2, and 5.3, we investigate the impact of each specific feature group on the learning process, i.e. by varying  $VT$ ,  $\theta$ , and  $DM$  one at a time, while holding the other two variables constant. Each section analyses the results of Table 3. In this table, the independent variables  $DM$ ,  $\theta$  and  $VT$  are varied in turn - each can take a single instantiation, namely DPH on content (denoted C), 1000 and CombSUM, respectively, or all of the listed variable instantiations in Table 2 (denoted \*). For example, C 1000 CombSUM (rows 1 & 9) denotes our baseline approach that created the sample of aggregate objects to re-rank (c.f. BM25 used by Liu for LETOR [5]). We test for significant differences from the baseline using the Wilcoxon Signed Rank test, denoted by  $\gg$  ( $p < 0.01$ ),  $>$  ( $0.01 \leq p \leq 0.05$ ) and  $=$  ( $p > 0.05$ ). Finally, Section 5.4 analyses the most important features for each task, while additional discussion follows in Section 5.5.

**Table 3.** MAP performances of various feature sets, trained using two learning to rank techniques. Significant differences from C 1000 CombSUM are denoted with  $\gg$  ( $p < 0.01$ ),  $>$  ( $0.01 \leq p \leq 0.05$ ) and  $=$  ( $p > 0.05$ ). \* \* \* denotes when all features from Table 2 are applied. The best learned model for each task is emphasised.

	<i>DM</i>	$\theta$	<i>VT</i>	# Features	EX:07	EX:08	BD:07	BD:08
AFS								
1	C	1000	CombSUM	1	0.2651	0.2532	0.2468	0.1909
2	C	1000	*	10	<b>0.4184</b> $\gg$	<b>0.4128</b> $\gg$	0.2870 $\gg$	0.2493 $\gg$
3	C	*	CombSUM	9	0.3978 $\gg$	0.3180 $>$	0.2514 $=$	0.2067 $=$
4	C	*	*	90	0.4134 $\gg$	0.4043 $\gg$	0.3148 $\gg$	0.2422 $\gg$
5	*	1000	CombSUM	6	0.2776 $=$	0.2578 $=$	0.2705 $\gg$	0.2022 $>$
6	*	1000	*	60	0.4153 $\gg$	0.4103 $\gg$	0.3264 $\gg$	0.2547 $\gg$
7	*	*	CombSUM	54	0.4076 $\gg$	0.3133 $>$	0.2653 $=$	0.2170 $\gg$
8	*	*	*	540	0.4107 $\gg$	0.4041 $\gg$	<b>0.3480</b> $\gg$	<b>0.2710</b> $\gg$
AdaRank								
9	C	1000	CombSUM	1	0.2651	0.2532	0.2468	0.1909
10	C	1000	*	10	0.4141 $\gg$	<b>0.4211</b> $\gg$	0.2925 $\gg$	0.2338 $\gg$
11	C	*	CombSUM	9	0.3802 $\gg$	0.3670 $\gg$	0.2439 $=$	0.1893 $=$
12	C	*	*	90	<b>0.4199</b> $\gg$	0.3857 $\gg$	0.3189 $\gg$	<b>0.2493</b> $\gg$
13	*	1000	CombSUM	6	0.2896 $>$	0.2581 $=$	0.2698 $\gg$	0.2044 $\gg$
14	*	1000	*	60	0.4010 $\gg$	0.4035 $\gg$	<b>0.3204</b> $\gg$	0.2327 $\gg$
15	*	*	CombSUM	54	0.3823 $\gg$	0.3688 $\gg$	0.2700 $>$	0.2075 $\gg$
16	*	*	*	540	0.3973 $\gg$	0.3791 $\gg$	0.2817 $=$	0.2284 $\gg$

### 5.1 Feature Group: Voting Techniques

Firstly, we examine the impact of the research question (1), namely whether applying more than a single voting technique increases the effectiveness of the learned aggregate ranking model. To analyse the influence of adding additional voting techniques on the effectiveness of the learned model, we compare C 1000 CombSUM (rows 1 & 9) with C 1000 \* (rows 2 & 10) in Table 3. We observe up to 70% relative improvements over the baseline. Indeed, these improvements are statistically significant for both learning to rank techniques, and for all topic sets. We conclude that building a ranking model with multiple voting techniques can massively benefit retrieval performance. Moreover, if we examine other settings, e.g. where multiple cutoffs or multiple ranking feature groups have already been applied, we see further improvements (see each setting in rows 3 vs 4, 5 vs 6, 7 vs 8, 11 vs 12, 13 vs 14, or 15 vs 16). Overall, these positive results show that applying multiple voting techniques and learning a suitable combination results in an effective model that robustly generalises to other topic sets on the same corpus.

### 5.2 Feature Group: Ranking Cutoffs

Next, we investigate research question (2), addressing whether adding more document ranking cutoffs increases retrieval effectiveness. Comparing C 1000 CombSUM (rows 1 & 9) with C \* CombSUM (rows 3 & 11), we observe significant

increases for the expert search task. However, on the blog distillation task, only AFS can identify models that improve over the baseline sample. This suggests that having multiple cutoffs are very useful for expert search, where there are highly on-topic documents retrieved early in the system ranking that bring valuable expertise evidence. On the other hand, for blog distillation, adding additional cutoffs brings less benefit, suggesting that for this task, the top ranked documents are, in general, less useful. We conclude that the multiple cutoffs feature group is useful for the expert search task only.

We also examine the impact of the multiple cutoffs feature group when the multiple document rankings or multiple voting techniques feature groups have already been applied. In these cases, we note that, in general, adding the multiple cutoffs feature group is beneficial for improving the effectiveness of the multiple document ranking feature group alone (rows 5 vs 7 and 13 vs 15). However when multiple voting techniques have been applied, multiple document ranking cutoffs have little or no positive impact on retrieval performance (rows 2 vs 4, 6 vs 8, 10 vs 12 and 14 vs 16). This suggests that the sources of evidence from multiple voting techniques and document ranking cutoffs are correlated. However, in a learning to rank setting, this is not a disadvantage, as the learning process will only select one of two similar features.

### 5.3 Feature Group: Document Rankings

We now examine the impact on effectiveness of using features based on multiple document rankings, as per the research question (3). Comparing with C 1000 CombSUM (rows 1 vs 5 and 9 vs 13), we note improvements for all settings. However, these are only statistically significant in 5 out of 8 settings. Overall, we conclude that while adding the multiple document rankings feature group does positively impact retrieval effectiveness, it does not have as large an effect as the multiple cutoffs or multiple voting techniques feature groups (e.g. rows 1 vs 5, compared to 1 vs 2, and 1 vs 3).

Nevertheless, the effectiveness of multiple document rankings can be improved by further adding the multiple cutoffs or multiple voting techniques feature groups. In fact, the best settings for the BD:07 and BD:08 tasks can be found when applying the 540 features of \* \* \* (see row 8 for AFS - we compare AFS and AdaRank performances in Section 5.5 below). We conclude that some aspects of the extra document rankings (collection enrichment, proximity or anchor text) do bring some useful additional evidence, particularly for blog distillation.

### 5.4 Task Analysis

We now address research question (4), by analysing the most important features identified when training for each task. Table 4 reports the top 4 features by weight as identified by AFS in the C \* \* feature set of 90 features, as well as the total number of features selected (out of 90). From this table 4, we observe that 12-19 features were typically chosen - this suggests the similarity

<sup>1</sup> For space reasons, we only report the best features from C \* \*. However, all of the conclusions are equally applicable to the most important features in \* \* \*.



**Table 4.** Strongest four features in C \*\*, and the number of features selected by AFS (out of 90)

EX:07	EX:08	BD:07	BD:08
C 50 CombMNZ-Norm1D	C 2000 expCombMNZ-Norm1D	C 250 expCombMNZ-Norm1D	C 5000 CombSUM-Norm1D
C 50 expCombMNZ	C 100 expCombSUM	C 100 expCombMNZ-Norm1D	C 4000 expCombSUM
C 2000 CombSUM-Norm1D	C 100 expCombMNZ-Norm1D	C 100 CombSUM	C 1000 CombMAX
C 1000 expCombMNZ	C 4000 expCombSUM	C 1000 CombMAX	C 500 CombMAX
(12)	(19)	(19)	(14)

of many of the features derived from the C document ranking (i.e. DPH). Of the first ranked chosen features, all apply voting techniques with profile length normalisation, suggesting that this is an important attribute of effective voting techniques. However, it is of note that the chosen voting techniques and cutoffs vary for different corpora and tasks. This attests the usefulness of our approach to learn effective models for different aggregate ranking corpora and tasks, since the various effective features for each task can be automatically selected and weighted.

## 5.5 Discussion

Having examined each of the feature groups in turn, we now analyse the combination of all feature groups. Looking across all feature groups, we note that once multiple voting techniques have been applied, applying multiple cutoffs or multiple document rankings has in general no marked benefit in retrieval performance across all search tasks. However, the multiple voting techniques feature group is robust across all tasks and learning to rank techniques (i.e. 27 significant increases out of 28 across all even numbered rows in Table 3). Indeed, for blog distillation, \*\* learned by AFS exhibits the highest performance, suggesting that by allowing the learner to choose from all 540 features, an effective and robust model can be learned. It is also of note that we performed additional experiments using 5-fold cross validation across all ~100 topics for each task (These results are omitted for reasons of brevity). While the obtained retrieval performances were naturally improved by more training, promisingly, all of the experimental conclusions were unchanged.

Comparing the learning to rank techniques, we note that higher quality results are generally found by AdaRank on expert search, however, on blog distillation, AFS is more successful. This contrasts with the results obtained on the training topics, where AFS always identifies a model that is significantly better than that by AdaRank. We leave a study on the attributes of features sets that make each learning to rank technique amenable to different tasks to future work.

Finally, we compare our results to the TREC best runs for each task. In particular, for both EX:08 and BD:08, we note that our best results are comparable to the top ranked group. For the expert search task, the highest performing TREC run deployed models encompassing the proximity of candidate name occurrences to the query terms. However, as this source of evidence is specific to the expert

search task, we chose not to deploy it to facilitate the inter-task cross-comparison of research question (4). Nevertheless, even without this expert search-specific evidence, our learned approach exhibits comparable results.

Overall, the experimental results allow us to conclude that our methodology for learning aggregate ranking models is effective and robust across two aggregate search tasks and four topic sets. Using a thorough analysis, we identified that the most effective features originated from the *VT* independent variable. However, both *DM* and  $\theta$  also bring valuable additional sources of evidence that can successfully be integrated into the learned model.

## 6 Related Work

Learned approaches for aggregate ranking tasks such as expert search have seen very little published work. In this section, we review the very recent existing literature, comparing and contrasting with our own approach. In particular, in [21], Cummins et al. used genetic programming to learn a formula for the weights of document associations within the candidate profiles. This is orthogonal to our approach, where features and weights are defined and learned at the aggregate level. While the genetic programming approach appears promising, our results on the expert search tasks are 15-19% higher than the best results obtained by their learned approach, showing the superiority of our proposed approach.

Fang et al. [22] recently introduced a discriminative approach for expert search. In this approach, the importance of candidate features and association features are automatically learned from the training data. However, the mathematical machinery to the discriminative approach is complex, requiring partial derivatives to be empirically evaluated via nonlinear optimisations (the authors in [22] used a BFGS Quasi-Newton optimisation). In contrast, by defining all features at the aggregate level, our approach can easily use existing learning to rank approaches without complex derivations, while also being agnostic to various aggregation strategies (e.g. voting techniques), rather than the two used in [22].

## 7 Conclusions

In this work, we proposed a novel yet natural approach to learn rankings for aggregate search tasks such as expert search and blog distillation. In this approach, each feature is defined at the aggregate object level (e.g. persons or blogs), and generated by instantiations of three independent variables, namely the document ranking weighting model, its rank cutoff, and the voting technique. From these features, we showed that effective and robust ensemble models can be learned using existing learning to rank techniques. Moreover, our experimental results showed that the inclusion of multiple voting techniques - each with different ways of ranking aggregates - results in the most marked and significant increases in retrieval performance. In the future, we will continue to develop novel and effective features within our learning methodology for aggregate ranking.

## References

1. Balog, K., Azzopardi, L., de Rijke, M.: Formal models for expert finding in enterprise corpora. In: Proceedings of SIGIR 2006, pp. 43–50 (2006)
2. Macdonald, C., Ounis, I.: Voting for candidates: Adapting data fusion techniques for an expert search task. In: Proceedings of CIKM 2006, pp. 387–396 (2006)
3. Elsas, J.L., Arguello, J., Callan, J., Carbonell, J.G.: Retrieval and feedback models for blog feed search. In: Proceedings of SIGIR 2008, pp. 347–354 (2008)
4. Balog, K., de Rijke, M., Weerkamp, W.: Bloggers as experts: feed distillation using expert retrieval models. In: Proceedings of SIGIR 2008, pp. 753–754 (2008)
5. Liu, T.Y.: Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3(3), 225–331 (2009)
6. Macdonald, C., Ounis, I.: Key blog distillation: ranking aggregates. In: Proceedings of CIKM 2008, pp. 1043–1052 (2008)
7. Santos, R.L.T., Macdonald, C., Ounis, I.: Voting for related entities. In: Proceedings of RIAO 2010 (2010)
8. Macdonald, C.: The Voting Model for People Search. PhD thesis, Univ. of Glasgow (2009)
9. Pederson, J.: The Machine Learned Ranking Story (2008), [http://jopedersen.com/Pre-presentations/The\\_MLR\\_Story.pdf](http://jopedersen.com/Pre-presentations/The_MLR_Story.pdf)
10. Metzler, D.A.: Automatic feature selection in the Markov random field model for information retrieval. In: Proceedings of CIKM 2007, pp. 253–262 (2007)
11. Xu, J., Li, H.: AdaRank: a boosting algorithm for information retrieval. In: Proceedings of SIGIR 2007, pp. 391–398 (2007)
12. Bailey, P., Craswell, N., de Vries, A.P., Soboroff, I.: Overview of the TREC-2007 Enterprise track. In: Proceedings of TREC 2007 (2007)
13. Balog, K., Soboroff, I., Thomas, P., Bailey, P., Craswell, N., de Vries, A.P.: Overview of the TREC 2008 Enterprise track. In: Proceedings of TREC 2008 (2008)
14. Macdonald, C., Ounis, I.: The influence of the document ranking in expert search. In: Proceedings of CIKM 2009, pp. 1983–1986 (2009)
15. Macdonald, C., Ounis, I.: On perfect document rankings for expert search. In: Proceedings of SIGIR 2009, pp. 740–741 (2009)
16. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the TREC-2007 Blog track. In: Proceedings of TREC 2007 (2007)
17. Ounis, I., Macdonald, C., Soboroff, I.: Overview of the TREC-2008 Blog track. In: Proceedings of TREC 2008 (2008)
18. Amati, G., Ambrosi, E., Bianchi, M., Gaibisso, C., Gambosi, G.: FUB, IASI-CNR and Univ. of Tor Vergata at TREC 2007 Blog track. In: Proceedings of TREC 2007 (2007)
19. Peng, J., Macdonald, C., He, B., Plachouras, V., Ounis, I.: Incorporating term dependency in the DFR framework. In: Proceedings of SIGIR 2007, pp. 843–844 (2007)
20. Peng, J., He, B., Ounis, I.: Predicting the usefulness of collection enrichment for enterprise search. In: Azzopardi, L., Kazai, G., Robertson, S., R uger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) ICTIR 2009. LNCS, vol. 5766, pp. 366–370. Springer, Heidelberg (2009)
21. Cummins, R., Lalmas, M., O’Riordan, C.: Learned aggregation functions for expert search. In: Proceedings of ECAI 2010, pp. 535–540 (2010)
22. Fang, Y., Si, L., Mathur, A.P.: Discriminative models of integrating document evidence and document-candidate associations for expert search. In: Proceedings of SIGIR 2010, pp. 683–690 (2010)

# Efficient Compressed Inverted Index Skipping for Disjunctive Text-Queries

Simon Jonassen and Svein Erik Bratsberg

Department of Computer and Information Science,  
Norwegian University of Science and Technology,  
Sem Sælands vei 7-9, NO-7491 Trondheim, Norway  
{simonj,sveinbra}@idi.ntnu.no

**Abstract.** In this paper we look at a combination of bulk-compression, partial query processing and skipping for document-ordered inverted indexes. We propose a new inverted index organization, and provide an updated version of the MaxScore method by Turtle and Flood and a skipping-adapted version of the space-limited adaptive pruning method by Lester et al. Both our methods significantly reduce the number of processed elements and reduce the average query latency by more than three times. Our experiments with a real implementation and a large document collection are valuable for a further research within inverted index skipping and query processing optimizations.

## 1 Introduction

The large and continuously increasing size of document collections requires search engines to process more and more data for each single query. Even with up-to-date inverted index partitioning, distribution and load-balancing approaches the performance of a single node remains important.

A large number of methods aimed to reduce the amount of data to be fetched from disk or processed on CPU have been proposed. Among these we find static and dynamic pruning, impact-ordered lists, compression, caching and skipping. In this paper we look at document-ordered inverted lists with a combination of bulk-compression methods, partial query processing and skipping.

One of the main challenges associated with processing of disjunctive (OR) queries is that documents matching *any* of the query terms might be returned as a result. In contrast, conjunctive (AND) queries require to return only those documents that match *all* of the terms. In the latter case, the shortest posting list can be used to efficiently *skip* through the longer posting lists and thus reduce the amount of data to be processed. An approach proposed by Broder et al. [2] was therefore to process a query as an AND-query, and only if the number of results is too low, process it once again as the original query. Instead, we look at two heuristics, the *MaxScore* method by Turtle and Flood [15] and the space-limited adaptive pruning method by Lester et al. [10], with a purpose to apply skipping in a combination with OR-queries.

The recent publications by Suel et al. [8] [16] [17] have demonstrated superior efficiency of the PForDelta [19] and its variants compared to the alternative index compression methods. For 32 bit words PForDelta compresses data in chunks of 128 entries and does fast decompression of data with unrolled loops. However, to our knowledge, the only skipping alternative considered by Suel et al. was storing the first element of each *chunk* in main memory. On the other hand, the skipping methods published so far optimize the number of *entries* to be fetched and/or decompressed, with no consideration of disk buffering optimizations, internal CPU caches and bulk-decompression.

The main motivation behind this paper is to process disjunctive queries just as efficiently as conjunctive. We expect that a proper combination of inverted index compression, skipping and query optimization techniques is sufficient to do so. The contribution of our work is as follows. (a) We present a novel and efficient skipping organization designed specifically for a bulk-compressed disk-stored inverted index. (b) We revise the MaxScore-heuristics and present a complete matching algorithm. (c) We present a modification of the pruning method by Lester in order to enable skipping. (d) We evaluate the performance of the inverted index and skipping methods against state-of-the-art methods with the GOV2 document collection and a large TREC query set on a real implementation, and provide important experimental results. Our methods significantly improve query processing efficiency and remove the performance gap between disjunctive and conjunctive queries. Finally, we show that, due to disk-access overhead, skipping more data does not necessary reduce the query latency.

This paper is organized as follows. Section 2 gives a short overview of related work. Section 3 presents the structure of our compressed self-skipping index. Section 4 revises the MaxScore method and presents an improved version of Lester’s algorithm. The experimental framework and results are given in Section 5. The final conclusions and directions for further work follow in Section 6.

## 2 Related Work

**Query optimizations.** Early query optimization strategies for inverted indexes have been considered by Buckley and Lewit [3]. Turtle and Flood [15] have discussed and evaluated a number of techniques to reduce query evaluation costs. We use **MaxScore** to refer to a document-at-a-time (DAAT) partial ranking optimization based on the maximum achievable score of a posting list and the score of the currently lowest ranked document mentioned in the paper.

However, the original description of MaxScore [15] omits some important details, and it differs from a later description by Strohman, Turtle and Croft [14]. Also the recited description of the method provided by Lacour et al. [9] is closer to the original rather than the later description of the method. We believe that both descriptions [14] [15] are correct and explain two different heuristics that can be combined. We find this combination to be highly efficient, but lacking a clear, unified explanation. For this reason we present a complete algorithmic implementation of MaxScore and explain how skipping can be done.

Moffat and Zobel [12] have presented the original *Quit/Continue* strategies for term-at-a-time processing (TAAT) and explained how these can be combined with efficient skipping. The paper also explains the choice of skipping distance for single- and multiple-level skipping. In a later comparison paper, Lacour et al. [9] have found these optimizations most efficient, while MaxScore was only slightly better than full TAAT evaluation. However, as we understand, the MaxScore implementation by Lacour et al. was limited only to partial ranking [15] with no skipping. We show that, compared to this method, skipping improves the average query latency with by a factor of 3.5.

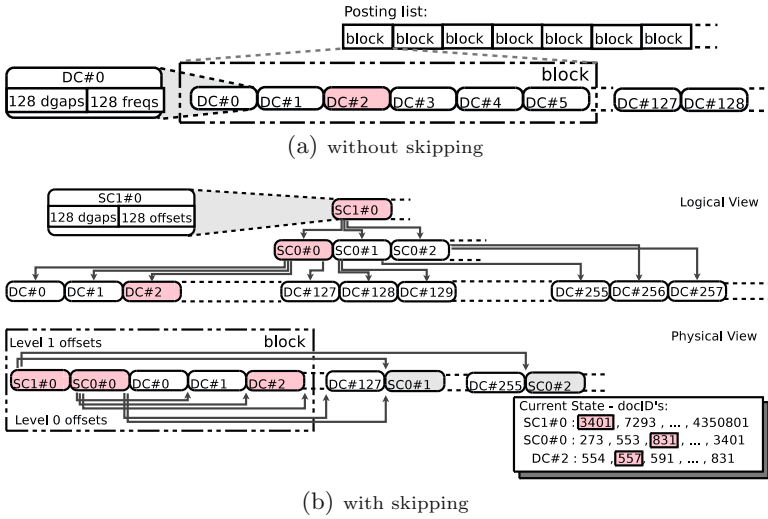
Lester et al. [10] introduced an efficient space-limited TAAT query evaluation method, which provides a trade-off between the number of maintained *accumulators* (ie. partially scored candidates) and the result quality. However, no considerations of skipping have been made by the authors. In our work, we present a modification of the method that does inverted list skipping and demonstrate a significant performance improvement.

**Compression.** PForDelta compression was originally proposed by Zukowski [19]. Suel et al. [17] have demonstrated the efficiency of this method compared to the other methods and suggested a number of improvements [16]. Skipping has been mentioned in most of the PForDelta related papers by Suel et al., but the only implementation considered so far was to store the first document ID from each chunk of each posting list, uncompressed, in main memory [8]. Opposite to this, we suggest a self-skipping compressed inverted index.

**Skipping.** Moffat and Zobel [12] wrote one of the first papers applying inverted index skipping and presented a method to choose optimal skip-lengths for single- and multiple-level skipping with respect to disk-access and decompression time. The model behind the method assumes to fetch and decompress an *element* at a time, and the optimization is done on the total number of fetched and decompressed entries. Instead, we assume data to be compressed in *chunks* and stored in *blocks* with its implications to the processing model.

Other methods to estimate optimal skipping distances were presented by Strohman and Croft [13], Chierichetti et al. [6] and Boldi and Vigna [1]. The first paper looks at skipping with impact-ordered inverted files and the second one looks at spaghetti skips in doctored dictionaries which are not related to our focus. The skipping structure described by Boldi and Vigna [1] has a certain similarity with ours. The smallest number of pointers to be skipped is a group of 32 or 64 entries, each compressed on its own, and skipping pointers are stored in towers. Our skipping structure, as we explain in the next section, compresses groups of 128 index postings or 128 skipping pointers in chunks, while the pointers corresponding to different *skipping-levels* are stored in different chunks.

Büttcher and Clarke [4] have presented an efficient I/O optimized random-access structure for random inverted index access. While having an insignificant similarity with our skipping organization, their method operates with nodes of constant byte size, such as an L2 cache line or a memory page. Our skipping structure operates with nodes of 128 elements and we separate physical block-size (used for disk-fetching) from the index layout itself. Finally, we optimize query



**Fig. 1.** Compressed inverted index organization

processing rather than random access, and we look at PForDelta compression which was considered by the authors only as promising further work.

### 3 Index Organization and Skipping

We look at processing of disjunctive queries with a disk-stored document-ordered inverted index [18]. Each posting list entry represents a document ID and a *term frequency*  $f_{d,t}$  denoting the number of occurrences. Additionally to the inverted index that we describe below, we store a *lexicon file*, a *document dictionary* and additional statistics such as the total number of unique terms, documents, postings and tokens. The lexicon file stores a posting list pointer and the corresponding *collection frequency*  $F_t$  and *document frequency*  $f_t$  for each indexed term. The document dictionary stores mapping between document IDs, original document naming and document lengths.

#### 3.1 Basic Inverted Index

Without skipping, we split posting lists in chunks of 128 entries. Each chunk consists of 128 document IDs and 128 frequencies, where the last chunk contains a maximum of 128 entries. We use *d-gaps* instead of the original IDs. Further, each group of d-gaps and frequencies is compressed on its own, but using the same compression method. Chunks with more than 100 entries are compressed using NewPFD [16], a variant of PForDelta which stores highest order bytes of exceptions and exception offsets as two Simple9 encoded arrays. Chunks with less than 100 entries are VByte compressed. We illustrate this in Figure 1(a).

Posting lists are processed using *iterators*. In this case, a list is fetched one *block* at a time. Blocks are zero-indexed, have a constant size  $B$ , but contain a

varied number of chunks, and the block number  $i$  has a start-offset of  $B * i$  bytes. The choice of the block size itself is transparent from the physical organization. Once a block is fetched, chunks are decoded one at a time. All d-gaps and frequencies contained in a single chunk are decoded at once.

### 3.2 Self-skipping Index

To support inverted index skipping we extract the last document ID and the *end-offset* from each chunk. These result in the lowest level of skipping hierarchy. Similar to the posting data itself, we divide these into groups of 128 elements, *skip-chunks*. The last document ID and the end-offset of each skip-chunk are recursively stored in a skip-chunk in the level above. As we illustrate in the upper part of Figure 1(b), the logical structure reminds of a B-tree.

The physical layout of our self-skipping index resembles a prefix traverse of the logical tree. For this reason, the offsets stored in the lowest level skip-chunks represent the length of a corresponding *data-chunk*. For the levels above, an offset represents the length of a referred skip-chunk plus the sum of its offsets. This way, each chunk stores *real* offsets between the chunks in a lower level, and the last offset corresponds to the offset to the next chunk at the same level. We illustrate this in the lower part of Figure 1(b).

Document IDs within each skip-chunk are also gap-coded. Both d-gaps and offsets are compressed using the same compression methods as data-chunks, NewPFD for chunks with more than 100 elements and VByte otherwise. Additionally to using gaps instead of full IDs and relative instead of absolute offsets, NewPFD itself stores differences from a frame of reference, which rewards the regularity in document ID distribution and compression ratio. Finally, we avoid use of bit-level pointers, which improves both the index size and querying performance.

**Inverted Index Skipping.** As with the original index, we separate the choice of the block-size from the index organization itself. We suggest to choose the size so that the first block of a list is likely to contain the first chunks of each skipping level and the first data chunk. Further we decompress each first skip-chunk from each level and the first data-chunk and use it as *state information* of the list iterator. We refer to these chunks as to *active* chunks (see Figure 1(b)). We also convert d-gaps into document ID's and relative offsets into absolute *pointers*.

Now, a *skip(d)*-operation can be done by comparing  $d$  to the last document ID of the active data-chunk. If  $d$  is greater, we compare  $d$  to the last document IDs of the active skip-chunks from the lowest to the highest level. We proceed climbing up until we get a last document ID greater or equal  $d$ . For each chunk we mark also the entry corresponding to the currently active chunk level under. At this point we compare  $d$  to the entries between the active and the last one until we get an entry with the document ID greater or equal  $d$ . Further, we fetch and decompress the chunk referred by the offset pointer. If the corresponding chunk is a data-chunk, we quickly find the required posting. Otherwise, we climb downwards until we get to a data-chunk. The worst-case number of decompressed chunks and fetched blocks in a random skip operation is therefore equal to the



**Algorithm 1.** Skipping modification of the Lester’s algorithm

---

**Data:** inverted index iterators  $\{I_{t_1}, \dots, I_{t_q}\}$  sorted by ascending collection frequency  $F_t$   
**Result:** top  $K$  query results sorted by descending score

```

1  $A \leftarrow \emptyset; v_t \leftarrow 0.0; h_t \leftarrow 0;$ 
2 foreach iterator  $I_t$  do
3    $skipmode \leftarrow false;$ 
4   if  $F_t < L$  then  $p \leftarrow F_t + 1;$ 
5   else if  $v_t = 0.0$  then calculate values of  $v_t, h_t, p$  and  $s$  according to Lester;
6   else
7     calculate new values of  $h_t$  and  $v_t$  from the old value of  $v_t;$ 
8     if  $h_t \geq f_{max}$  then  $p \leftarrow F_t + 1; skipmode \leftarrow true;$ 
9   if  $skipmode$  then
10    foreach accumulator  $A_d \in A$  do
11      if  $I_t.docID < d$  then
12        if  $I_t.skipTo(d) = false$  then  $A \leftarrow A - \{A_* | A_* < v_t\};$  proceed to line 2;
13      if  $I_t.docID = d$  then  $A_d \leftarrow A_d + s(I_t);$ 
14      if  $A_d < v_t$  then  $A \leftarrow A - A_d;$ 
15    else
16      foreach document  $d \in I_t \cup A$  do
17        recalculate values of  $v_t, h_t, p$  and  $s$  when necessary;
18        if  $d \notin I_t$  then
19          if  $A_d < v_t$  then  $A \leftarrow A - A_d;$ 
20        else if  $d \notin A$  then
21          if  $I_t.freq \geq h_t$  then  $A_{I_t.docID} \leftarrow s(I_t); A \leftarrow A + A_{I_t.docID};$ 
22        else
23           $A_d \leftarrow A_d + s(I_t);$ 
24          if  $A_d < v_t$  then  $A \leftarrow A - A_d;$ 
25 return  $resHeap.decrSortResults(A);$ 

```

---

number of skip-levels,  $O(\log(F_t))$ , where  $F_t$  is the collection frequency of the term  $t$ .

## 4 Query Processing Methods

Query processing is done by stemming and stop-word processing query tokens, looking-up the vocabulary, fetching and processing the corresponding posting list, followed by extraction, sorting and post-processing of the  $K$  best results.

With term-at-a-time (TAAT) query processing, each posting list is processed at once, which allows certain speed-up, but requires to maintain a set of partial results, *accumulators*. Alternatively, with document-at-a-time (DAAT) query processing, all posting lists are processed in parallel and documents are scored one at a time. While DAAT processing has been considered more efficient in combination with skipping, the methods for term-partitioned distributed inverted files [11] apply TAAT processing. As we consider index skipping to be useful also for distributed query processing, we look at both methods.

### 4.1 Term-At-A-Time Processing

The advantage of the Lester’s method compared to the Continue approach was demonstrated in the original paper [10]. We prefer to look at the Lester’s method instead of TAAT MaxScore [15] since the latter is a special case of the Continue

**Algorithm 2.** MaxScore

---

**Data:** inverted index iterators  $\{I_{t_1}, \dots, I_{t_q}\}$  sorted by descending maximum score  $\hat{s}(I_t)$   
**Result:** top  $K$  query results sorted by descending score

```

1  $q_{req} = q$ ;  $resHeap \leftarrow \emptyset$ ;
2 calculate a set of cumulative maximum scores  $\hat{a}$ ,  $\hat{a}(I_{t_i}) = \sum_{i \leq j \leq q} \hat{s}(I_{t_j})$ ;
3 while  $q > 0$  and  $q_{req} > 0$  do
4    $score \leftarrow 0$ ;  $d_{cand} \leftarrow \min_{i \leq q_{req}} (I_{t_i}.doc)$ ;
5   for  $i = 1$ ;  $i \leq q$ ;  $i \leftarrow i + 1$  do
6     if  $score + \hat{a}(I_{t_i}) < resHeap.minScore$  then proceed to line 14;
7     if  $i > q_{req}$  then
8        $\lfloor$  if  $I_{t_i}.skipTo(d_{cand}) = false$  then remove  $I_{t_i}$ ; update  $\hat{a}$ ,  $q$ ,  $q_{req}$ ; continue;
9     if  $I_{t_i}.doc = d_{cand}$  then  $score \leftarrow score + s(I_{t_i})$ ;
10  if  $score > resHeap.minScore$  then
11     $resHeap.insert(d_{cand}, score)$ ;
12    for  $i = q_{req}$ ;  $i > 1$ ;  $i \leftarrow i - 1$  do
13       $\lfloor$  if  $\hat{a}(I_{t_i}) < resHeap.minScore$  then  $q_{req} \leftarrow q_{req} - 1$ ;
14  increment used and remove empty iterators  $I_{t_i \leq q_{req}}$ , update  $\hat{a}$ ,  $q$ ,  $q_{req}$  if necessary;
15 return  $resHeap.decrSortResults()$ ;
```

---

approach. TAAT MaxScore stops creating new accumulators when the maximum achievable score of the next term falls below the current score of the  $K$ th candidate. Thus the method is equivalent to the Continue approach with an additional requirement to track the first  $K$  top-scored candidates.

The modified version of the pruning method is given in Algorithm 1. The algorithm is equivalent to the original [10] except from the lines 3, 8 and 9-15. The choice and usage semantics of the threshold variables  $v_t$  and  $h_t$  and adjustment variables  $p$  and  $t$  are explained in the original paper.  $L$  is the target value for the number of maintained accumulators. Additionally, we introduce a constraint,  $f_{max}$ , which can be chosen either statically or dynamically adjusted based on the maximum document frequency of the current term.

When  $f_{max} < h_t$ , the frequency required for a posting to be evaluated by the scoring function,  $s(I_t)$ , query processing switches into *skipmode* (line 9-15), where no new accumulators will be created and therefore skipping can be performed. In skipmode, we use existing accumulators to skip through a posting list. With a low *accumulator target value*  $L$ , skipmode is achieved shortly after processing the first term and longer posting lists are efficiently skipped.

## 4.2 Document-At-A-Time Processing

Our interpretation of MaxScore heuristics is given in Algorithm 2. Prior to query processing we order the iterators by descending maximum score and calculate their cumulative maximum scores from last to first. Further, we say that terms from  $t_1$  and up to  $t_{q_{req}}$  are in the *requirement set*. We say that a term is in the requirement set if its cumulative maximum score is greater than the score of currently least ranked candidate,  $resHeap.minScore$ . From this definition, terms that are *not* in the requirement set can be skipped (line 7-9). Our algorithm begins with all terms in the requirement set,  $q_{req} = q$ . As the algorithm proceeds beyond the first  $K$  documents, it reduces the requirement set (line 10-11) until

there is only one term left. This idea is similar to the description given by Strohman, Turtle and Croft [14].

Each iteration of the algorithm begins by finding the lowest document ID within the requirement set (line 4). Further, we check every iterator from first to last and, if the current score plus the cumulative maximum score of the remaining terms is less than *resHeap.minScore*, the algorithm stops further evaluation for this candidate and proceeds to the next one (line 6). This idea is similar to the description given by Turtle and Flood [15]. Finally, we terminate processing when the requirement set becomes empty or all postings have been processed.

## 5 Evaluation

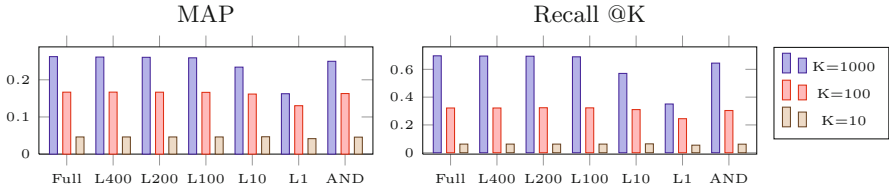
In order to evaluate index skipping and query processing algorithms we use the 426GB TREC GOV2 document corpus. We perform both stemming using the Snowball algorithm and stop-word removal on the document collection. The resulting index contains 15.4 million unique terms, 25.2 million documents, 4.7 billion pointers and 16.3 billion tokens. The total size of the compressed inverted index without skipping is 5.977GB. Additionally, we build another index with skipping, which adds 87.1MB to the index size, that is a 1.42% increase. The self-skipped inverted index contains 15.1 million posting lists with zero skip-levels, 279647 posting lists with one skip level, 15201 with two levels and only 377 with three levels.

For result quality evaluation we use the TREC Adhoc Retrieval Topics and Relevance Judgements 801-850 [5] without any modifications. For performance evaluation we use the Terabyte Track 05 Efficiency Topics [7]. As we look at optimizations for multi-keyword queries we remove any query with less than two terms matching in the index lexicon. From the original 50000 queries (having query length of 1 to 18 terms and an average length of 2.79 terms; matching 1-10, avg. 2.41 terms in the inverted index) we get 37132 queries (2-18, avg. 3.35; matching 2-10, avg. 2.91 terms), from which we extract a subset of the first 10000 queries.

All algorithms and data structures were implemented in Java. All the experiments were executed on a single workstation having an Intel Core 2 Quad 2.66GHz processor, 8GB RAM and a 1TB 7200RPM SATA2 hard-drive and running GNU/Linux. No caching optimizations have been done, OS disk cache was dropped before each run. For all experiments except the block size evaluation itself we operate with 16KB blocks. The query model used is the Okapi BM-25.

### 5.1 Term-At-A-Time Processing

We compare our modification of the Lester's method using self-skipping index to the original method and full OR evaluation using a non-skipping index. Additionally, we compare it also to a self-skipping implementation of AND-processing. Our implementation of the AND method uses shortest posting list to skip through the longer ones. Any entries not matching in the later posting lists are removed from the accumulator set. Our implementation of the Lester's



**Fig. 2.** Mean average precision and average recall comparison of the TAAT methods

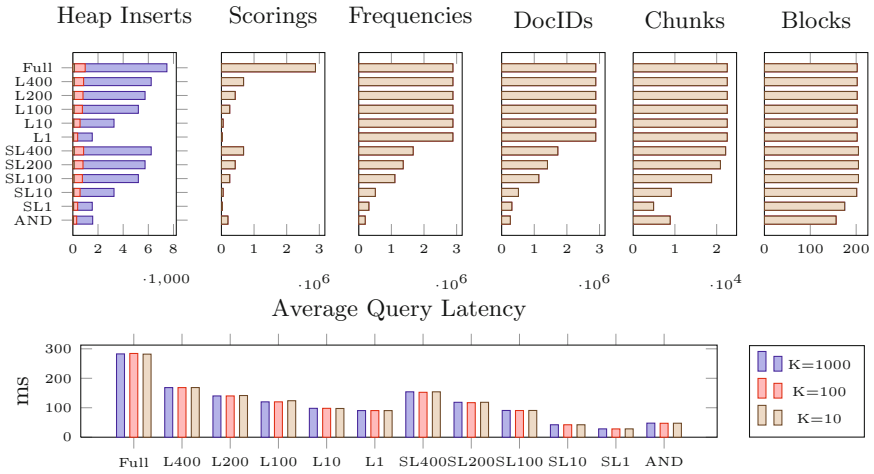
method is similar to the one used in the Zettair Search Engine<sup>1</sup>. It also avoids recalculating threshold variables if  $h_t \geq f_{max}$  at the beginning of a posting list. For both the Lester’s method itself and the modification we use a statically defined  $f_{max} = 2000$ .

The result set with our modification is always equivalent to the one returned by the Lester’s method itself. In Figure 2 we illustrate the average mean of precision scores after each relevant document (MAP) and the total recall for the query retrieval topics. We evaluate Lester’s method with accumulator set target values ( $L$ ) 400000 (L400), 200000 (L200), 100000 (L100), 10000 (L10) and 1000 (L1). The results show that there is no significant difference in the result quality as long the result set size ( $K$ ) is large enough. For  $K = 1000$ , the target value  $L$  can be chosen as low as 100000 without a significant impact on the result quality. For  $K = 100$  and 10,  $L$  can be chosen as low as 10000.

The performance evaluation results are given in Figure 3. For each method configuration we execute two runs. The first one, illustrated in the upper half of the figure, measures the total number of candidates inserted into the result heap, calls to the scoring function, frequencies and document IDs evaluated, decompressed chunks and fetched blocks. The second one, illustrated in the lower half, measures the resulting query latency (without profiling overhead). As the results show, our method significantly reduces the number of evaluated document IDs and frequencies compared to the other methods. The number of candidates inserted into the result heap and posting scorings is reduced similar to the original method. We observe also a decrease in the number of decompressed chunks due to inverted index skipping. However, the number of fetched blocks with a target value larger than 10000 is actually larger than with the other methods, due to storage overhead from skip-chunks. A significant reduction in the number of fetched blocks is observed only for  $L = 1000$  (SL1), but as we will demonstrate in Section 5.3, with a smaller block size, the number of fetched blocks will be reduced also for larger target values. Finally, our results show that the size of the result set  $K$  influences only the number of heap inserts and does not affect any other counts nor the resulting query latency.

The measured average query latency is illustrated in the lower half of Figure 3. For  $L = 1000$ , our modification is 25% faster than the original method and 69%, or more than three times, faster than a full evaluation. For a lower target value the performance of our method is comparable to the AND-only evaluation. With

<sup>1</sup> <http://www.seg.rmit.edu.au/zettair/>

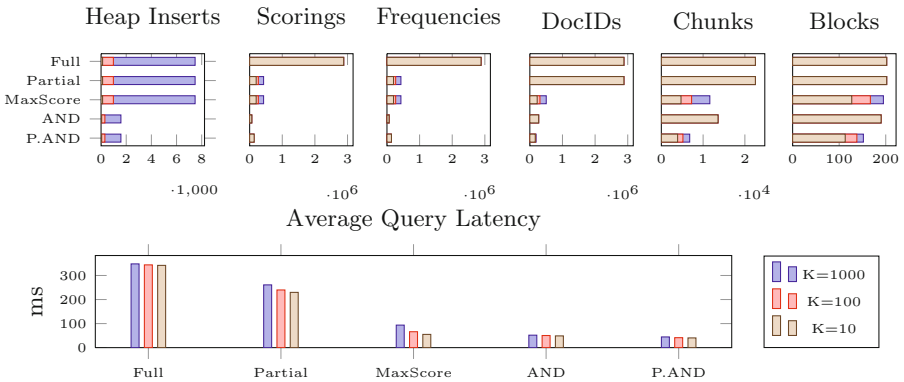


**Fig. 3.** Average number of processed entities (top) and query latency (bottom) of the TAAT methods. Our method is marked with "SL" (skipping-Lester).

$K = 100$  and  $L = 10000$  our method outperforms the AND method, while the result quality is not affected.

### 5.2 Document-At-A-Time Processing

The performance evaluation results of the DAAT methods are illustrated in Figure 4. In these experiments we compare the full and partial evaluation methods without skipping, our interpretation of the MaxScore method with skipping, and AND and Partial AND methods with skipping. The partial evaluation method is similar to the MaxScore interpretation by Lacour et al. [9], the method avoids scoring the rest of the postings for a document if the current score plus the cumulative maximum score of the remaining terms falls below the score of



**Fig. 4.** Average number of processed entities and query latency of the DAAT methods

the currently least ranked candidate. Partial AND (P.AND) combines skipping AND and partial evaluation. As the result set returned by the MaxScore method is identical to the full evaluation, we do not provide result quality evaluation for the DAAT methods.

Our result shows that the partial evaluation alone can significantly reduce the number of posting scorings and number of evaluated frequencies, which results in a 25% reduction in the average query latency. Our interpretation of MaxScore provides a significant further reduction to the number of evaluated document IDs. For  $K = 1000$  it halves the number of decompressed chunks and reduces the number of fetched blocks. The average query latency with the MaxScore is 3.7 times shorter than with a full evaluation, or 2.8 times compared to the partial evaluation method alone. With  $K$  less than 1000, the reduction in the number of decompressed chunks and fetched blocks is even more significant, and the performance of our method is close to the full AND (less than 31% difference). The improvement can be explained by increased *resHeap.minScore*, which allows to skip more data, and decreased overhead from changes in the candidate result set during processing.

### 5.3 Physical Block Size

Figure 5 illustrates the average latency versus fetched data volume per query for the first 1000 queries in the query log. The average number of fetched blocks can be obtained by dividing the data volume by the block size. As the results show, a small block size reduces the total data volume, most significantly for the MaxScore. However, the total latency improves gradually as the block size becomes larger. Decrease in the measured latency between 1 and 64KB blocks is 8.6% for Full DAAT, 9.8% for Full TAAT, 12.1% for skipping-Lester with  $L = 100000$  and 13.7% for MaxScore. This can be explained by a decrease of random disk-seeks and the fact that, when a chunk is split between two blocks, it requires two block fetches to fetch the chunk. While not illustrated, the number of fetched blocks decreases as the block size increases. In summary, our results show that, without caching, small block sizes reduce only the total data volume, but not the resulting query latency.

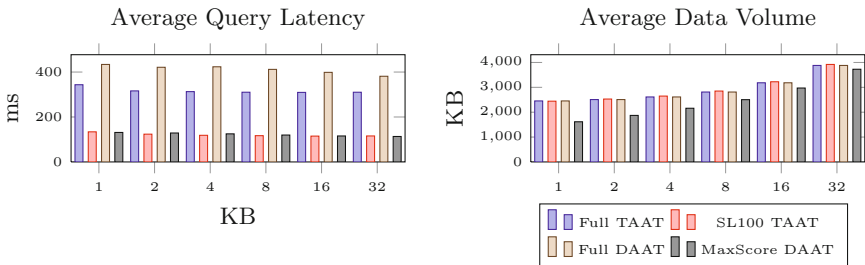


Fig. 5. Average query latency and data volume with varied block size,  $K = 1000$

## 6 Conclusions and Further Work

In this paper we have presented an efficient self-skipping organization for a bulk-compressed document-ordered inverted index. From the experimental results, our query processing optimizations achieve more than three times speed-up compared to a full, non-skipping, evaluation and remove the performance gap between OR and AND queries. Further improvements can be done by postponed/lazy decompression of posting frequencies, extension of compression methods to 64 bit words and chunk-wise caching of posting lists.

**Acknowledgment.** This work was supported by the iAd Project funded by the Research Council of Norway and the Norwegian University of Science and Technology. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect the views of the funding agencies.

## References

1. Boldi, P., Vigna, S.: Compressed perfect embedded skip lists for quick inverted-index lookups. In: Consens, M.P., Navarro, G. (eds.) SPIRE 2005. LNCS, vol. 3772, pp. 25–28. Springer, Heidelberg (2005)
2. Broder, A., Carmel, D., Herscovici, M., Soffer, A., Zien, J.: Efficient query evaluation using a two-level retrieval process. In: Proc. CIKM. ACM, New York (2003)
3. Buckley, C., Lewit, A.: Optimization of inverted vector searches. In: Proc. SIGIR, pp. 97–110. ACM, New York (1985)
4. Büttcher, S., Clarke, C.: Index compression is good, especially for random access. In: Proc. CIKM, pp. 761–770. ACM, New York (2007)
5. Büttcher, S., Clarke, C., Soboroff, I.: The trec 2006 terabyte track. In: Proc. TREC (2006)
6. Chierichetti, F., Lattanzi, S., Mari, F., Panconesi, A.: On placing skips optimally in expectation. In: Proc. WSDM, pp. 15–24. ACM, New York (2008)
7. Clarke, C., Soboroff, I.: The trec 2005 terabyte track. In: Proc. TREC (2005)
8. Ding, S., He, J., Yan, H., Suel, T.: Using graphics processors for high-performance ir query processing. In: Proc. WWW, pp. 1213–1214. ACM, New York (2008)
9. Lacour, P., Macdonald, C., Ounis, I.: Efficiency comparison of document matching techniques. In: Proc. ECIR (2008)
10. Lester, N., Moffat, A., Webber, W., Zobel, J.: Space-limited ranked query evaluation using adaptive pruning. In: Ngu, A.H.H., Kitsuregawa, M., Neuhold, E.J., Chung, J.-Y., Sheng, Q.Z. (eds.) WISE 2005. LNCS, vol. 3806, pp. 470–477. Springer, Heidelberg (2005)
11. Moffat, A., Webber, W., Zobel, J., Baeza-Yates, R.: A pipelined architecture for distributed text query evaluation. *Inf. Retr.* 10(3), 205–231 (2007)
12. Moffat, A., Zobel, J.: Self-indexing inverted files for fast text retrieval. *ACM Trans. Inf. Syst.* 14(4), 349–379 (1996)
13. Strohman, T., Croft, W.: Efficient document retrieval in main memory. In: Proc. SIGIR, pp. 175–182. ACM, New York (2007)
14. Strohman, T., Turtle, H., Croft, W.: Optimization strategies for complex queries. In: Proc. SIGIR, pp. 219–225. ACM, New York (2005)

15. Turtle, H., Flood, J.: Query evaluation: strategies and optimizations. *Inf. Process. Manage.* 31(6), 831–850 (1995)
16. Yan, H., Ding, S., Suel, T.: Inverted index compression and query processing with optimized document ordering. In: *Proc. WWW*, pp. 401–410. ACM, New York (2009)
17. Zhang, J., Long, X., Suel, T.: Performance of compressed inverted list caching in search engines. In: *Proc. WWW*, pp. 387–396. ACM, New York (2008)
18. Zobel, J., Moffat, A.: Inverted files for text search engines. *ACM Comput. Surv.* 38(2) (2006)
19. Zukowski, M., Heman, S., Nes, N., Boncz, P.: Super-scalar ram-cpu cache compression. In: *Proc. ICDE*, p. 59. IEEE Computer Society, Los Alamitos (2006)



# Within-Document Term-Based Index Pruning with Statistical Hypothesis Testing

Sree Lekha Thota and Ben Carterette

Department of Computer and Information Sciences,  
University of Delaware, Newark, DE, USA  
lekhat@gmail.com, carteret@cis.udel.edu

**Abstract.** Document-centric static index pruning methods provide smaller indexes and faster query times by dropping some within-document term information from inverted lists. We present a method of pruning inverted lists derived from the formulation of unigram language models for retrieval. Our method is based on the statistical significance of term frequency ratios: using the two-sample two-proportion (2P2N) test, we statistically compare the frequency of occurrence of a word within a given document to the frequency of its occurrence in the collection to decide whether to prune it. Experimental results show that this technique can be used to significantly decrease the size of the index and querying speed with less compromise to retrieval effectiveness than similar heuristic methods. Furthermore, we give a formal statistical justification for such methods.

## 1 Introduction

*Index pruning* is a family of methods for deciding whether to store certain information about term occurrences in documents. It is useful for decreasing index size and increasing query speed, assuming the information lost does not substantially affect retrieval results. Dynamic pruning methods are commonly used to make decisions about cache storage, while static pruning reduces disk storage needs. Static pruning can be either *term-centric*, in which term information is dropped from inverted lists independently of the documents they occur in, or *document-centric*, in which term information is dropped from within documents.

Regardless of the type of pruning, decisions about what to prune are usually made in an ad hoc manner using heuristics. This work presents a method for document-centric pruning derived from the widely-used unigram language modeling approach to retrieval. Like other approaches, we attempt to remove only the terms that are not informative for computing the language model score of a document. Our decisions are based on formal statistical methods that operate under the same modeling assumptions as language modeling: that documents have been sampled term-by-term from some underlying population. Treating the frequency of a term occurrence within a document as an estimate of the proportion of the underlying space that term represents, we can test whether that proportion is equivalent to the proportion in a general background model. If it

is, the term presumably contains no information about the document’s relevance to queries including that term.

Specifically, we use the two-sample two-proportion (2P2N) test for statistical significance to determine whether the term frequency within a document is different from its frequency in the background. If we cannot detect significance, we prune the term—we assume it is not informative. The advantage of using statistical significance is not only that it follows from the same modeling assumptions as language models, but also that its errors can be anticipated and controlled in a statistical sense. Thus we hypothesize that we can substantially reduce index size while maintaining greater retrieval effectiveness than heuristic approaches.

## 2 Previous Work

When reducing index size, we can distinguish between *lossless* techniques and *lossy* techniques. Many lossless methods have been proposed [18,11,2,20]; these are generally compression algorithms that reduce the space required to store an inverted list. These methods are highly effective and are now used in almost all retrieval systems.

Index pruning is a lossy technique: information about the terms and documents is not stored at all. While this can have a large positive effect on space, it can also negatively affect retrieval effectiveness. Thus they should be applied judiciously. Dynamic index pruning techniques are applied during the query time in order to reduce computational cost of query processing. Moffat and Zobel [14] proposed an evaluation technique that uses early recognition of which documents are likely to be highly ranked to reduce costs without degradation in the retrieval effectiveness. Tsegay et al. [19] investigate caching only the pieces of the inverted list that are actually used to answer the query during dynamic pruning. These techniques reduce memory usage, but not disk usage.

Carmel et al. [8] introduced the concept of static index pruning technique to the information retrieval systems. They present a term-centric approach in which for each term in the index only the top  $k$  postings are retained. The main idea behind this method is to use the search engine’s ranking in order to evaluate the importance of each inverted list and determine which entries can be removed from the index. Each term in the index is submitted as a query to the search engine and from the resulting document set for pruning. The term is removed from the document  $D$  if it is not present in the top  $k$  portion of the ranked result set from the search engine.

Büttcher and Clarke [5,6] presented a document centric approach: the decision about whether the term’s posting should be present or not depends on its rank by a score computed within a document rather than the posting’s rank within its term posting list. For each document  $D$  in the corpus, only the postings for the top  $k$  terms in the document are kept in the index. The terms are ranked based on their contribution to the document’s Kullback-Leibler divergence from the rest of the collection.

In 2005, de Moura et al. [13] proposed a locality based static pruning method which is a variation of Carmel’s method that aims at predicting what set of terms

may occur together in queries and using this information to preserve common documents in the inverted lists of these term. A boosting technique for Carmel’s static index pruning has been proposed by Blanco and Barreiro [4] in which they use the probabilistic-based scoring function (BM25) instead of the tf-idf method and address some features like updating of the document lengths and the average document length in the pruned inverted file which are not considered in the original model. More recently, Nguyen [15] presented a posting based approach which is a generalization of both document-centric and term-centric approaches.

### 3 Pruning Using the Two-Sample Two-Proportion Test

As described in Section 1, our pruning method is derived from the unigram language model. We start by showing the derivation, then describing the statistical method we will use. We then refine the method to account for possible errors.

#### 3.1 Language Modeling

The basic idea for our method is derived from the query-likelihood retrieval model. Language modeling [21,12] is one of the most effective and widely-used retrieval models. In the unigram language model, documents are modeled as term-by-term samples from some underlying population. They are ranked by the probability of sampling the query  $Q$  from the multinomial “bag of words” representing a document  $D$ , i.e. by the value of  $P(Q|D)$ . This is estimated as:

$$P(Q|D) = \prod_{q_i \in Q} P(q_i|D) \simeq \sum_{q_i \in Q} \log(P(q_i|D))$$

where

$$P(q_i|D) = \frac{tf_{q_i,D}}{|D|}$$

and  $tf_{q_i,D}$  is the number of times term  $q_i$  occurs in the document  $D$ , and  $|D|$  is the total number of terms in the document. Since this probability could be zero if just one query term fails to occur in the document, the model is smoothed with a background model based on the full collection of documents [21], which is also modeled as a sample from an underlying space:

$$P(q_i|D) = \lambda \frac{tf_{q_i,D}}{|D|} + (1 - \lambda) \frac{ctf_{q_i}}{|C|}$$

where  $\lambda$  is a smoothing parameter,  $ctf_{q_i}$  is the total number of occurrences of  $q_i$  in the entire collection  $C$ , and  $|C|$  is the total number of terms in the collection. We are agnostic about the modeling assumptions that lead to a particular choice of form or value of  $\lambda$ ; the Dirichlet prior is a common approach that has been shown to work well in practice [21].

From the above equation, we can see that when the ratio of document term count to document length is exactly equal to the ratio of collection term frequency to the total collection term count, the two  $\lambda$ -scaled ratios cancel out and the score of the document depends only on the collection term frequency:

$$\frac{tf_{q_i,D}}{|D|} = \frac{ctf_{q_i}}{|C|} \Rightarrow P(q_i|D) = \frac{ctf_{q_i}}{|C|}$$

Since the document's final score does not depend on the frequency of such a term, that information can be pruned with no penalty to retrieval effectiveness.

In general, we cannot expect that the two ratios computed from data will be exactly equal, even if they actually are equivalent in the underlying term populations from which  $D$  and  $C$  have been sampled. The nature of sampling means that the ratios are only estimates of the “true” underlying values, and may be higher or lower randomly but within well-defined ranges. Thus we need a way to test whether the two ratios are equivalent in the *statistical* sense of falling within a given confidence interval.

### 3.2 The Two-Sample Two-Proportion Test

The two-sample two-proportion (2N2P) test is a statistical procedure for testing the hypothesis that two proportions are equal given two estimates of those proportions calculated from two different samples [11, chapter 6]. We start by computing the difference between two proportions. Because those proportions are based on samples, they have some variance. When their difference is not exactly zero, the variance may still be high enough that we can consider them effectively equivalent. Dividing the difference between the two proportions by a standard error produces a normally-distributed test statistic  $Z$  that we can use to make a decision about whether to consider the two proportions different.

The value of the  $Z$  statistic is calculated using the formula

$$Z = \frac{\frac{x_1}{n_1} - \frac{x_2}{n_2}}{E}$$

where  $n_1, n_2$  are the sample sizes,  $x_1, x_2$  are the number of observed occurrences, and  $E$  is the standard error of the difference in proportions. The standard error is calculated as:

$$E = \sqrt{P(1-P) \left( \frac{1}{n_1} + \frac{1}{n_2} \right)}$$

where

$$P = \frac{x_1 + x_2}{n_1 + n_2}$$

$Z$  has an approximately standard normal distribution, and thus to determine whether the difference in proportions is significant we check the probability of

observing a value of  $Z$  and higher (and/or  $-Z$  and lower, depending on the type of test) in a normal distribution with mean 0 and standard deviation 1. If that probability is less than some pre-selected value  $\alpha$ , we reject the hypothesis that the proportions are the same.  $\alpha$  is frequently chosen to be 0.05, which corresponds to  $|Z| \approx 2$  in a two-sided test or  $|Z| \approx 1.65$  in a one-sided test. In Figure 1, for instance, we would reject the hypothesis that the two proportions are equal (or that  $x_2/n_2$  is greater) if we calculate  $Z > 1.65$ .

### 3.3 Static Index Pruning Using the 2N2P Test

We will use the above described statistical method to make pruning decisions. In our method, we calculate the value of the  $Z$  statistic of each term in a document. This value is calculated by using the document length and the collection length as the sample sizes and the ratios of frequency of the word in the document to the document length and the frequency of the word in the entire collection to the collection length as the proportions. Based on the value of the term’s  $Z$  statistic, we decide whether to keep the word in the index or to drop it. The value of the  $Z$  statistic gives us the significance of the term to the document.

$$Z = \frac{tf_{q_i,D} - \frac{ctf_{q_i}}{|C|}}{E}$$

where  $tf_{q_i,D}$  is the frequency of the term in the document,  $|D|$  is the length of the document,  $ctf_{q_i}$  is the frequency of the term in the entire collection,  $|C|$  is the total number of terms in the entire collection and  $E$  is the standard error. The standard error is calculated using the following formula,

$$E = \sqrt{P(1 - P) \left( \frac{1}{|D|} + \frac{1}{|C|} \right)}$$

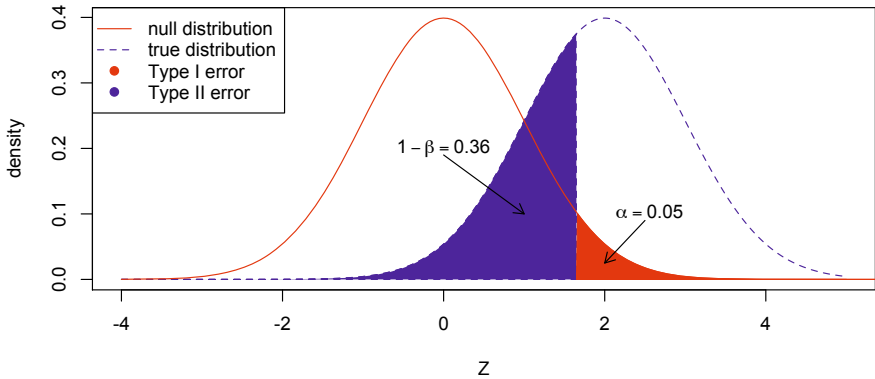
where

$$P = \frac{tf_{q_i,D} + \frac{ctf_{q_i}}{|C|}}{|D| + |C|}$$

Note that we are using the same assumptions as the unigram language model: that document and collection are sampled term-by-term from an underlying space, and the term proportions are thus estimates of their true occurrence.

We next choose a threshold value of  $Z$  to denote the significance level needed to keep information about a term in the index, i.e. we choose a value for  $Z$  *a priori* and store only those terms whose calculated value is greater than this value. Note that choosing different thresholds is equivalent to choosing different significance levels  $\alpha$ ; in Figure 1 we have chosen a threshold of 1.65, corresponding to  $\alpha = 0.05$  in a one-sided test. As the threshold increases (significance level decreases), the size of the pruned index decreases.

Therefore, the value of  $Z$  for a term gives us the level of importance of the term to the meaning of the document. Only the terms that are meaningful to the



**Fig. 1.** Illustration of statistical power. If the null hypothesis is true (red normal density curve), there is a 5% probability of a Type I error of not pruning a non-informative term (red shaded region for a one-sided test). If the null hypothesis is *not* true and the true value of the  $Z$ -statistic is 2 (blue normal density curve), there is a 36% probability of a Type II error of pruning an informative term (blue shaded region) and consequently only 64% probability of correctly keeping that term.

document are added to the index; the remaining terms are discarded. Note also that the number of terms pruned from each document is different and depends on their informative content rather than the length of the document. The resulting size of the index depends on the number of postings that are significant enough, based on the  $Z$  value we specify, to prune from each document.

Note that the stored values of the document lengths and collection statistics must not be modified for our method to work. If the test tells us to prune a term from a document, only its document-level  $tf$  value is pruned from the index. All other information about the document and collection remains unchanged, including document length  $|D|$ , collection frequency  $ctf$ , and collection length  $|C|$ . Even the fact that a pruned term appears in a document must still be stored (to handle the difference between a term with  $tf = 0$  and one with  $tf/|D| = ctf/|C|$ ), though this can be done with a minimum of overhead. If any of these values changed, the derivation in Section 3.1 above would no longer work.

### 3.4 Statistical Power of the 2N2P Test

Using the results of a statistical hypothesis test to make a decision always has some chance of resulting in an incorrect action. In our case, we may incorrectly decide to keep a term that is not meaningful to the document (a Type I error of finding a significant difference when one does not exist), or we may incorrectly decide to prune a term that is meaningful to the document (a Type II error of failing to find a significant difference when one exists). Using different thresholds for  $Z$  controls the Type I error: the lower  $Z$  is, the more likely we are to prune terms, and therefore Type I errors become more likely.

Our method is meant to determine when term counts do not need to be stored to maintain retrieval effectiveness, as we showed in Section 3.1. We can continue to store them if we are willing to accept the cost of the disk space and query processing time. This means that Type I errors are relatively cheap. Type II errors are substantially more expensive: once we’ve decided to prune a term, we cannot use any information about it in calculating document scores. If we were wrong to prune it, it may significantly and negatively impact retrieval performance. Therefore we would like to be able to control the probability of Type II errors as well as Type I errors when pruning terms from documents.

Type II error rates are inversely related to *statistical power*. Power is usually denoted  $\beta$ , and the expected Type II error rate is  $1 - \beta$ . Power analysis [11] allows us to use known quantities such as document length and collection size along with a desired Type I error rate and effect size (described below) to determine when it is best to prune a term.

Figure 1 illustrates Type I and Type II errors. If the null hypothesis is true, the  $Z$ -statistic will be drawn from the normal density function centered at zero (colored red). If the threshold for rejection is  $\alpha = 0.05$  ( $Z \approx 1.65$ ), then there is a 5% chance of a Type I error. But if the null hypothesis is *not* true, the  $Z$ -statistic will be drawn from some other distribution. In this example, we suppose that the “true” value is 2, and the observed value will be sampled from a variance-1 normal distribution centered at 2 (colored blue). The probability of a Type II error, then, is the probability that the observed value is *less than* the threshold. If it is, we would fail to reject the null hypothesis, even though it is not true.

To implement power analysis, we first define an estimated *effect size* that we consider large enough to be meaningful. Effect size, denoted  $h$ , is a dimensionless quantity computed from the proportions and  $P$ :

$$h = \frac{\frac{tf_{q_i,D}}{|D|} - \frac{ctf_{q_i}}{|C|}}{\sqrt{P(1 - P)}}$$

A loose guide to interpretation of effect size is that an effect size between 0 and 0.2 is considered “small”, between 0.2 and 0.4 is “moderate” and greater than 0.4 is “strong”. We could choose to keep terms only if the effect size is strong, i.e. only if the estimated ratios are substantially different. Or we could choose to keep terms with small effect sizes on the assumption that Type I errors are “cheap” and it takes a lot of evidence for us to decide to prune a term.

Once we have chosen an effect size, we calculate the value of  $\alpha$  (equivalently, the  $Z$  statistic threshold) that would result in finding a significant difference with probability  $\beta$ . This is done by solving the following equation for  $\alpha_D$ .

$$\Phi \left( \Phi^{-1}(\alpha_D) - h \sqrt{\frac{1}{|D|} + \frac{1}{|C|}} \right) - \beta = 0$$

To understand this, consider each component in turn:  $\Phi(Z)$  is the standard normal cumulative density function, i.e. the area under the standard normal density curve from  $Z$  to  $\infty$ . It always has a value between 0 and 1. Its inverse

$\Phi^{-1}(\alpha_D)$  is therefore the threshold for  $Z$  that would result in a Type I error rate of  $\alpha_D$ . We shift that value by effect size  $h$  scaled by a function of the total evidence we have (measured by  $|D|$  and  $|C|$ ), then calculate the probability of observing a  $Z$  of that value or greater in a standard normal distribution. In Figure 1,  $\alpha_D = 0.05$ ,  $\Phi^{-1}(\alpha_D) \approx 1.65$ ,  $h\sqrt{\frac{1}{|D|} + \frac{1}{|C|}} \approx 2$ , and  $\Phi(1.64 - 2) \approx 0.64$ . This is the power achieved when  $\alpha_D = 0.05$ .

There is no closed-form solution for  $\alpha_D$ , so we solve it with linear search. Once we have the value of  $\alpha_D$ , the corresponding  $Z_D$  can be found using normal distribution tables or by another application of the quantile function. We then apply pruning exactly as in Section 3.3: when the  $Z_D$  statistic is greater than that computed by power analysis, the term is kept; otherwise it is pruned.

The practical effect of this is essentially that each document has its own threshold for pruning, and that threshold is based on two parameters: desired effect size  $h$  and desired power  $\beta$  to detect that effect size. So we trade one parameter (a global  $Z$  threshold) for two that give us a local threshold for each document  $Z_D$ . Furthermore, since effect size and Type II error rate are monotonically related, we can effectively reduce the parameter space to a single parameter—desired power  $\beta$ . Increasing the power parameter results in lower local  $Z_D$  thresholds, which in turn results in fewer terms being pruned.

## 4 Experimental Results

### 4.1 Data

For empirical analysis, we used the GOV2 collection of 25,183,256 documents as our corpus to index. GOV2 has been used to experiment on efficiency as part of the TREC Terabyte track [9,10,7]. The queries we used to evaluate effectiveness are title queries from topic numbers 701 through 750 developed for the 2006 TREC Terabyte track.

### 4.2 Building the Index

We used the Indri retrieval engine [17] for indexing and query processing. We implemented all pruning methods in Indri. We used the Krovetz stemmer and a stopword list of 420 words that is included in the Lemur source distribution.

For calculating the value of  $Z$  at index time, we refer to a complete unpruned index of the dataset in order to obtain the term frequency in the document, the term frequency in the entire collection, document lengths, and the collection size. The Indri code is modified such that before each term is added to the index, this calculated value of  $Z$  is compared to the desired value (submitted as a parameter at runtime) and is added to the index only if it is higher compared to the desired value. We did not alter stored document lengths, collection lengths, and collection term frequencies.



**Table 1.** Pruning using KL-Divergence

Index size	$k$	MAP	Prec@10
100%	-	0.2642	0.5106
40.85%	100	0.1437	0.4601
58.54%	200	0.1675	0.4786
65.84%	250	0.1817	0.4800
76.89%	350	0.2130	0.4917
90.39%	1000	0.2519	0.5107

**Table 2.** Pruning using the 2N2P test

Index size	$Z$	MAP	Prec@10
100%	0	0.2642	0.5106
42.61%	50	0.1531	0.4578
56.61%	30	0.1662	0.4745
68.98%	10	0.1978	0.4900
75.32%	5	0.2136	0.4978
92.1%	1.69	0.2527	0.5106

### 4.3 Baseline

We compare to Büttcher and Clarke’s document-centric KL-divergence method [6] described in Section 2. We calculate the KL-divergence score of each of the terms in the document, and the top  $k$  terms are retained in the document while the others are pruned. The following formula is used to calculate the KL-divergence scores of the terms in the document:

$$Score_{DCP}(t_i) = P(t_i|D) \log \left( \frac{P(t_i|D)}{P(t_i|C)} \right)$$

where  $P(t_i|D)$  and  $P(t_i|C)$  are calculated as in Section 3.1 above. Again, Indri is modified such that only the top  $k$  terms in each document are stored in the index and the rest are pruned. For different values of  $k$ , different index sizes are obtained.

### 4.4 Evaluation

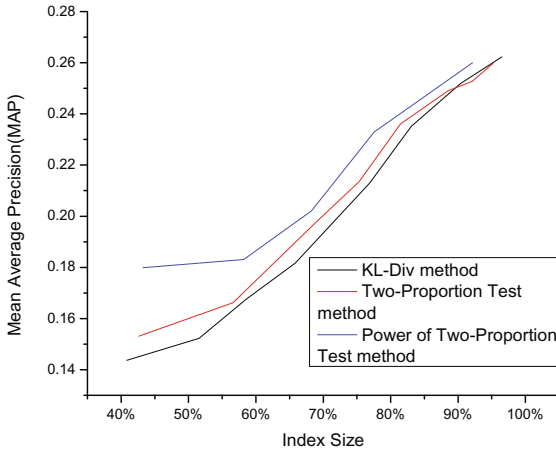
We calculate the size of a pruned index as a percentage of the complete unpruned index. Our goal is to test whether retrieval speed and effectiveness are substantially affected by pruning using the 2N2P tests, and to compare those tests to the baseline. We evaluate effectiveness by mean average precision (MAP) and mean precision at rank 10 (prec@10) over the 50 Terabyte queries. We evaluate retrieval speed by the total time it takes to process those queries.

### 4.5 Results

Table 1 shows the results of the KL-Divergence method. The various index sizes are obtained by repeating the experiments with increasing values of  $k$ , which is the number of terms stored from each document. The MAPs obtained at different index sizes are shown. Table 2 shows the results of varying a global  $Z$ -statistic for the 2N2P test to produce different index sizes. Table 3 shows the results using 2N2P power analysis with desired effect size  $h = 0.2$  and varying power  $\beta$ . Note that index sizes are not identical across the tables because there is

**Table 3.** Pruning using power analysis

Index size	MAP	Prec@10
100%	0.2642	0.5106
43.23%	0.1799	0.4345
58.21%	0.1831	0.4837
68.24%	0.2021	0.4900
77.54%	0.2329	0.4946
92.16%	0.2600	0.5070



**Fig. 2.** Index size vs. MAP for the three pruning methods

no way to guarantee that each method will result in the same number of pruned postings. We have chosen parameter values that produce roughly equal index sizes.

In all cases MAP and prec@10 decrease with index size, but it is clear from the results that, given an index size, the statistical hypothesis testing method presented in this paper provides a small increase in effectiveness. Furthermore, MAP scores obtained using power analysis show substantial improvement over both methods.

Since we cannot guarantee that the methods will produce the same-size index, effectiveness results are summarized graphically in Figure 2. Here we can see that the 2N2P methods produce nearly uniformly better results across index sizes, and the gain from using power analysis is strong.

Query times are illustrated in Figure 3. The two methods based on 2N2P are both slightly faster than the KL-Divergence method, though they are not substantially different from each other: with only 50 queries, the variance is high enough that these results are not significant. We did not have any specific hypothesis about query times; we present these results out of interest.

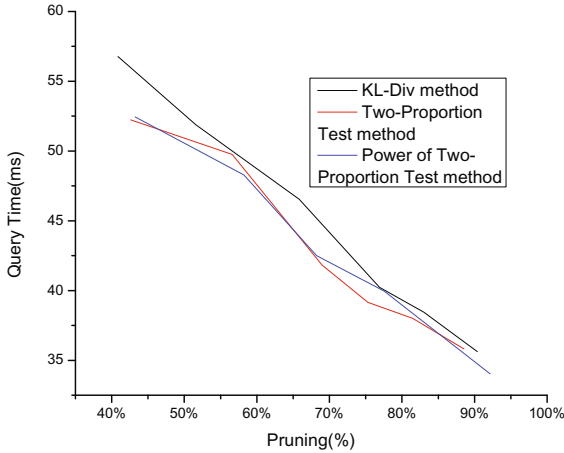


Fig. 3. Index size vs. query processing time for the three pruning methods

## 5 Conclusions and Future Work

We have presented a within-document term based index pruning method that uses formal statistical hypothesis testing. In this method, the terms in the document which have the least effect on the score of the document are pruned from the index, thus reducing its size with little compromise in theory on the effectiveness of the retrieval. The significance of the terms is calculated by using the  $Z$  statistic value from a two-sample two-proportion test that document term frequency is equivalent to collection term frequency.

We implemented two different approaches of this technique, one of which uses a constant threshold of  $Z$  irrespective of the document length, the other calculating a threshold of  $Z$  for each document based on its length using power analysis. From our experimental results, these methods not only decreased the index size but also were relatively successful in maintaining the performance of the system compared to the KL-Divergence method.

Our results are based on formal statistical analysis rather than heuristics, and derived from the same assumptions as the query-likelihood language model. Thus they suggest why static pruning methods work: they use evidence about documents and collections to eliminate information from the index that is irrelevant for scoring the document against queries. We believe similar statistical approaches could be used to prune indexes optimally for other retrieval methods, including BM25; an interesting future direction may be statistical pruning of more specific information such as term positions for use with more complex models such as Indri's inference network model.

## References

1. Anh, V.N., Moffat, A.: Inverted index compressed using word-aligned binary codes. *Inf. Retr.* 8(1), 151–166 (2005)
2. Anh, V.N., Moffat, A.: Pruned query evaluation using precomputed impacts. In: *Proceedings of SIGIR* (2006)
3. Azzopardi, L., Losada, D.E.: An efficient computation of the multiple-bernoulli language model. In: Lalmas, M., MacFarlane, A., Rüger, S.M., Tombros, A., Tsirikla, T., Yavlinsky, A. (eds.) *ECIR 2006*. LNCS, vol. 3936, pp. 480–483. Springer, Heidelberg (2006)
4. Blanco, R., Barreiro, A.: Boosting static pruning of inverted files. In: *Proceedings of SIGIR* (2007)
5. Büttcher, S., Clarke, C.L.A.: Efficiency vs. effectiveness in terabyte-scale information retrieval. In: *Proceedings of TREC* (2005)
6. Büttcher, S., Clarke, C.L.A.: A document-centric approach to static index pruning in text retrieval systems. In: *Proceedings of CIKM* (2006)
7. Büttcher, S., Clarke, C.L.A., Soboroff, I.: The TREC 2006 Terabyte track. In: *Proceedings of TREC* (2006)
8. Carmel, D., Cohen, D., Fagin, R., Farchi, E., Hersovici, M., Maarek, Y., Soer, A.: Static index pruning for information retrieval systems. In: *Proceedings of SIGIR*, pp. 43–50 (2001)
9. Clarke, C.L.A., Craswell, N., Soboroff, I.: Overview of the TREC 2004 Terabyte track. In: *Proceedings of TREC* (2004)
10. Clarke, C.L.A., Scholer, F., Soboroff, I.: The TREC 2005 Terabyte track. In: *Proceedings of TREC* (2005)
11. Cohen, J.: *Statistical Power Analysis for the Behavioral Sciences*, 2nd edn. Routledge Academic (1988)
12. Croft, W.B., Lafferty, J. (eds.): *Language Modeling for Information Retrieval*. Springer, Heidelberg (2003)
13. de Moura, E.S., dos Santos, C.F., Fernandes, D.R., Silva, A.S., Calado, P., Nascimento, M.A.: Improving web search efficiency via a locality based static pruning method. In: *Proceedings of WWW* (2005)
14. Moffat, A., Zobel, J.: Self-indexing inverted files for fast text retrieval. *ACM TOIS* 14(4), 349–379 (1996)
15. Nguyen, L.T.: Static index pruning for information retrieval systems: A posting-based approach. In: *Proceedings of LSDS-IR, CEUR Workshop*, pp. 25–32 (2009)
16. Persin, M., Zobel, J., Sacks-Davis, R.: Filtered document retrieval with frequency-sorted indexes. *JASIS* 47(10), 749–764 (1996)
17. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language-model based search engine for complex queries. In: *Proceedings of the International Conference on Intelligent Analysis* (2005)
18. Trotman, A.: Compressing inverted files. *Inf. Retr.* 6, 5–19 (2003)
19. Tsegay, Y., Turpin, A., Zobel, J.: Dynamic index pruning for effective caching. In: *Proceedings of CIKM* (2007)
20. Witten, I.H., Moffat, A., Bell, T.C.: *Managing Gigabytes*. Morgan Kaufmann, San Francisco (1999)
21. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. *ACM TOIS* 22(2), 179–214 (2004)

# SkipBlock: Self-indexing for Block-Based Inverted List

Stéphane Campinas<sup>1,2</sup>, Renaud Delbru<sup>1</sup>, and Giovanni Tummarello<sup>1</sup>

<sup>1</sup> Digital Enterprise Research Institute,  
National University of Ireland,  
Galway, Ireland

<sup>2</sup> École Pour l'Informatique et les Techniques Avancées,  
Le Kremlin-Bicêtre, France

{stephane.campinas,renaud.delbru,giovanni.tummarello}@deri.org

**Abstract.** In large web search engines the performance of Information Retrieval systems is a key issue. Block-based compression methods are often used to improve the search performance, but current self-indexing techniques are not adapted to such data structure and provide sub-optimal performance. In this paper, we present SkipBlock, a self-indexing model for block-based inverted lists. Based on a cost model, we show that it is possible to achieve significant improvements on both search performance and structure's space storage.

## 1 Introduction

The performance of Information Retrieval systems is a key issue in large web search engines. The use of compression techniques and self-indexing inverted files [8] is partially accountable for the current performance achievement of web search engines. On the one hand, compression maximises IO throughput [3] and therefore increases query throughput. On the other hand, self-indexing inverted files [8] enables the intersection of inverted lists in sub-linear time.

Nowadays efficient inverted index compression methods tend to have a block-based approach [6,10,11]. An inverted list is divided into multiple non-overlapping blocks of records. The coding is then done a block at a time independently. Despite block-based coding approaches providing incontestable benefits, the self-indexing method [8] achieves only sub-optimal performance on block-based inverted lists. The reason is that the self-indexing technique disregards the block-based structure of the inverted list that can be used for designing a more efficient self-indexing structure as we will show in this paper.

We present in this paper an approach for self-indexing of block-based inverted lists. We demonstrate the benefits of our block-based self-indexing technique by comparing it against the original self-indexing approach based on a cost model. In Section 2 we first review the original self-indexing technique based on the Skip List data structure, before presenting in Section 3 our approach. Section 4 discusses the problem of searching within Skip List intervals. In Section 5 we define a cost model and compare four implementations of the SkipBlock model

against the original Skip List model. In Section 6 we recall the main finding of the research and the remaining task.

## 1.1 Related Work

The Skip List data structure is introduced in [9] as a probabilistic alternative to balanced trees and it is shown in [5] to be as elegant and easier to use than binary search trees. Such a structure is later employed for self-indexing of inverted lists in [8]. Self-indexing of inverted lists enables a sub-linear complexity in average when intersecting two inverted lists. [2] proposes a way to compress efficiently a Skip List directly into an inverted list and shows that it is possible to achieve a substantial performance improvement. In [4], the authors introduce a method to place skips optimally based on a query distribution. In [7], the authors present a generalized Skip List data structure for concurrent operations. In this paper, we introduce a new model for self-indexing of block-based inverted lists based on an extension of the Skip List data structure. Our work is orthogonal to the previous works, since each of them could be adapted to our model.

## 2 Background: Self-indexing for Inverted Lists

An inverted list is an ordered list of compressed records (e.g., documents identifiers). When intersecting two or more inverted lists, we often need to access random records in those lists. A naive approach is to scan linearly the lists to find them. Such an operation is not optimal and can be reduced to sub-linear complexity in average by the use of the self-indexing technique [8]. Self-indexing relies on a Skip List data structure to build a sparse index over the inverted lists and to provide fast record lookups. In this section, we first present the Skip List model and its associated search algorithm. We finally discuss the effect of the probabilistic parameter with respect to the Skip List data structure and search complexity.

### 2.1 The Skip List Model

Skip List are used to index records in an inverted list at regular interval. These indexing points, called *synchronization points*, are organized into a hierarchy of linked lists, where a linked list at level  $i + 1$  has a probability  $p$  to index a record of the linked list at level  $i$ . The probabilistic parameter  $p$  is fixed in advance and indicates the *interval* between each synchronization point at each level. For example in Figure 1, a synchronization point is created every  $\frac{1}{p^1} = 16$  records at level 1, every  $\frac{1}{p^2} = 256$  records at level 2, and so on. In addition to the pointer to the next synchronization point on a same level, a synchronization point at level  $i + 1$  has a pointer to the same synchronization point at level  $i$ . For example in Figure 1, the first synchronization point at level 3 (i.e., for the record 4096) has a pointer to the level 2, which itself has a pointer to the level 1. This hierarchical structure enables to quickly find a given record using a top-down search strategy.

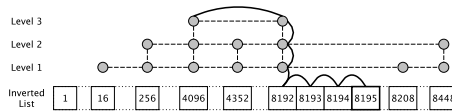
Given the probabilistic parameter  $p$  and the size  $n$  of an inverted list, we can deduce two characteristics of the resulting Skip List data structure: (1) the expected number of levels and (2) the size, i.e., the total number of synchronization points. The number of levels in the Skip List is defined by  $L(n) = \lfloor \ln_{\frac{1}{p}}(n) \rfloor$ , which is the maximum as stated in [9]. The total number of synchronization points is given by  $S(n) = \sum_{i=1}^{L(n)} \lfloor n \times p^i \rfloor$ , which sums up the number of synchronization points expected at each level.

### 2.2 Skip List Search Algorithm

Searching an element in a Skip List is performed with a top-down strategy. The search starts at the head of the top list and performs a linear walk over the list as long as the target is greater than a synchronization point. The search goes down one level if and only if the target is lower than the current synchronisation point, and resumes the linear walk. The search stops when the current synchronization point is (a) equal to the target, or (b) on the bottom level and the upper bound of the target. At this stage, it means we have found the interval of records containing our target element.

Figure 1 depicts with a solid line the search path in a Skip List with  $p = \frac{1}{16}$  and  $L(n) = 3$  levels to the record 8195. At the top of the Skip List, we walk to the record 8192. Then we go down to level 1 and stop because the current synchronization point, i.e., the record 8208, is greater than the target. At this point, we know that the target record is in the next interval on the inverted list.

The search complexity is defined by the number of steps necessary to find the record interval containing the target element. In the worst case, the number of steps at each level is at most  $\frac{1}{p}$  in at most  $L(n)$  levels. Consequently, the search complexity is  $\frac{L(n)}{p}$ .



**Fig. 1.** Skip List with  $p = \frac{1}{16}$ . Dashed lines denote pointers between synchronization points. The solid line shows the search path to the record 8195.

### 2.3 Impact of the Probabilistic Parameter

In this section, we discuss the consequences of the probabilistic parameter on the Skip List data structure. Table 1a reports for low (i.e.,  $\frac{1}{1024}$ ) and high (i.e.,  $\frac{1}{2}$ ) probabilities (1) the complexity  $\frac{L(n)}{p}$  to find the interval containing the target record, and (2) the size  $S(n)$  of the Skip List structure. There is a trade-off to achieve when selecting  $p$ : a high probability provides a low search complexity but at a larger space cost, and a low probability reduces considerably the required space at the cost of higher search complexity. The SkipBlock model provides a way to reduce even more the search complexity in exchange of a larger data structure.

**Table 1.** Search and size costs of Skip List and SkipBlock with  $n = 10^8$ .  $|I|$  stands for an interval length.  $C$  reports the search complexity to find an interval (Sections 2.2 and 3.2).

(a) Skip List with $ I  = \frac{1}{p}$ .						(b) SkipBlock with $ I  = \frac{ B }{p}$ .								
$ I $	2	16	64	128	1024	$ I $	16	64	128	1024				
$S(n)$	99 999 988 6 666 664 1 587 300 787 400 97 751					$p \cdot  B $	$\frac{1}{4} : 4$	$\frac{1}{4} : 2$	$\frac{1}{4} : 16$	$\frac{1}{4} : 8$	$\frac{1}{4} : 32$	$\frac{1}{4} : 16$	$\frac{1}{4} : 256$	$\frac{1}{4} : 128$
$C$	54	112	320	512	3072	$S_B(n)$	8 333 328 7 142 853	2 083 328 1 785 710	1 041 660 892 853	130 203 111 603				
						$C$	48	64	44	56	40	56	36	48

### 3 SkipBlock: A Block-Based Skip List Model

In this section, we introduce the SkipBlock model and present its associated search algorithm. Finally we discuss how the SkipBlock model offers finer control over the Skip List data structure in order to trade search against storage costs.

#### 3.1 The SkipBlock Model

The SkipBlock model operates on *blocks* of records of a fixed size, in place of the records themselves. Consequently, the probabilistic parameter  $p$  is defined with respect to a block unit. A synchronization point is created every  $\frac{1}{p^i}$  blocks on a level  $i$ , thus every  $\frac{|B|}{p^i}$  records where  $|B|$  denotes the block size. A synchronization point links to the first record of a block interval. Compared to Figure 1, a SkipBlock structure with  $p = \frac{1}{8}$  and  $|B| = 2$  also has an interval of  $\frac{|B|}{p^1} = 16$  records. However, on level 2, the synchronization points are separated by  $\frac{|B|}{p^2} = 128$  instead of 256 records. We note that with  $|B| = 1$ , the SkipBlock model is equivalent to the original Skip List model. Therefore this model is a generalization of the original Skip List model. The number of levels is defined by  $L_B(n) = \left\lceil \ln_{\frac{1}{p}} \left( \frac{n}{|B|} \right) \right\rceil$  and the size by  $S_B(n) = \sum_{i=1}^{L_B(n)} \left\lfloor \frac{n \times p^i}{|B|} \right\rfloor$ .

#### 3.2 SkipBlock Search Algorithm

Within the SkipBlock model, the search algorithm returns an interval of blocks containing the target record. In Section 4, we discuss for searching a record within that interval. The search strategy is identical to the one presented in Section 2.2: we walk from the top to the bottom level, and compare at each step the current synchronization point with the target. The search strategy applies the same termination criteria as in the Skip List search algorithm. The search complexity in the worst case becomes  $\frac{L_B(n)}{p}$ .

#### 3.3 Impact of the Probability and of the Block’s Size

The SkipBlock model provides two parameters to control its Skip List data structure: the probabilistic parameter  $p$  and the block size  $|B|$ . Compared to the



original Skip List model, the block size parameter enables a finer control over the Skip List structure. For example, to build a structure with an interval of length 64, the original Skip List model proposes only one configuration given by  $p = \frac{1}{64}$ . For this same interval length, SkipBlock proposes all the configurations that verify the equation  $\frac{|B|}{p} = 64$ . Table 11 reports statistics of some SkipBlock configurations for the same interval lengths as in Table 1a. Compared to Skip List on a same interval length, SkipBlock shows a lower search complexity in exchange of a larger structure.

## 4 Searching Records in an Interval

The Skip List and SkipBlock techniques enable the retrieval of a record interval given a target record. The next step consists in finding the target record within that interval. A first strategy (S1) is to linearly scan all the records within that interval until the target is found. Its complexity is therefore  $O(|I|)$  with  $|I|$  the length of an interval.

SkipBlock takes advantage of the block-based structure of the interval to perform more efficient search strategies. We define here four additional strategies for searching a block-based interval. The second strategy (S2) performs (a) a linear scan over the blocks of the interval to find the block holding the target and (b) a linear scan of the records of that block to find the target. The search complexity is  $\frac{1}{p} + |B|$  with  $\frac{1}{p}$  denoting the linear scan over the blocks and  $|B|$  the linear scan over the records of one block. Similarly to S2, the third strategy (S3) performs the step (a). Then, it uses an inner-block Skip List structure to find the target, restricted to one level only. The complexity is  $\frac{1}{p} + \frac{1}{q} + \lfloor |B| \times q \rfloor$  with  $q$  the probability of the inner Skip List. In contrast to S3, the fourth strategy (S4) uses a non-restricted inner-block Skip List structure. The complexity is  $\frac{1}{p} + \frac{L(|B|)+1}{q}$  with  $q$  the inner Skip List probability. The fifth one (S5) builds a Skip List structure on the whole interval instead of on a block. Its complexity is then  $\frac{L(\frac{|B|}{p})+1}{q}$ , with  $q$  the inner Skip List probability. The strategies S3, S4 and S5 are equivalent to S2 when the block size is too small for creating synchronization points.

## 5 Cost-Based Comparison

In this section, we define a cost model that is used to compare five SkipBlock implementations and the original Skip List implementation.

*Cost Model.* For both the Skip List and the SkipBlock, we define a cost model by (a) the cost to search for the target, and (b) the cost of the data structure's size. The search cost consists of the number of synchronization points traversed to reach the interval containing the target, plus the number of records scanned in that interval to find the target. The size cost consists in the total number of synchronization points in the data structure, including the additional ones in the intervals for S3, S4 and S5.

*Implementations.* We define as the baseline implementation, denoted  $I_1$ , the Skip List model using the strategy ( $S1$ ). We define five implementations of the SkipBlock model, denoted by  $I_2, I_3, I_4, I_5$  and  $I_6$ , based on the five interval search strategies, i.e.,  $S1, S2, S3, S4$  and  $S5$  respectively. The inner Skip List in implementations  $I_4, I_5$  and  $I_6$  is configured with probability  $q = \frac{1}{16}$ . The inner Skip List in  $I_5$  and  $I_6$  have at least 2 levels. The size costs are  $S(n)$  for  $I_1$ ,  $S_B(n)$  for  $I_2$ ,  $S_B(n) + \frac{n}{|B|}$  for  $I_3$ ,  $S_B(n) + \lfloor n \times q \rfloor$  for  $I_4$ ,  $S_B(n) + \frac{S(|B|) \times n}{|B|}$  for  $I_5$  and  $S_B(n) + \frac{S(p \times |B|) \times n}{p \times |B|}$  for  $I_6$ .

*Comparison.* With respect to the SkipBlock model, we tested all the possible configurations for a given interval length. We report that all of them were providing better search cost than the baseline. We report in Table 2 the search and size cost of the configurations that are providing the best search cost given an interval length. We observe that  $I_2$  already provides better search cost than the baseline  $I_1$  using the same search strategy  $S1$ , in exchange of a larger size cost. The other implementations, i.e.,  $I_3, I_4, I_5$  and  $I_6$  which use a more efficient interval search strategies further decrease the search cost. In addition, their size cost decreases significantly with the size of the interval. On a large interval (1152),  $I_5$  and  $I_6$  allow yet smaller search cost (69) than  $I_4$  with a similar size cost. Compared to the Skip List with a smaller interval (16), they achieve a smaller search cost with a similar size. To conclude,  $I_4, I_5$  and  $I_6$  seem to provide a good compromise between search and size costs with large intervals;  $I_5$  and  $I_6$  offering slightly better search cost in exchange of a slightly greater size cost.

**Table 2.** Search (i.e., SC) and size (i.e., ZC, in million) costs with  $n = 10^8$ . SkipBlock implementations report the best search cost with the associated size cost.

I	8						16						512						1152					
	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$
SC	72	56		54			112	62		56			1536	548	120	64	70		3456	1186	154	74	69	
ZC $\times e6$	14.3	16.7		50.0			6.7	12.5		25.0			0.2	0.3	1.8	6.5	6.9		0.09	0.17	1.7	6.4	6.6	6.7

## 6 Conclusion and Future Work

We presented SkipBlock, a self-indexing model for block-based inverted lists. The SkipBlock model extends the original Skip List model and provides a backbone for developing efficient interval search strategies. Compared to the original Skip List model, SkipBlock achieves with a structure of similar size a lower search cost. In addition, SkipBlock allows finer control over the Skip List data structure and so additional possibilities for trading search costs against storage costs. Future work will focus on real world data benchmarks in order to assess the performance benefits of the SkipBlock model.

## References

1. Anh, V.N., Moffat, A.: Index compression using 64-bit words. *Softw. Pract. Exper.* 40(2), 131–147 (2010)
2. Boldi, P., Vigna, S.: Compressed perfect embedded skip lists for quick inverted-index lookups. In: Consens, M.P., Navarro, G. (eds.) *SPIRE 2005*. LNCS, vol. 3772, pp. 25–28. Springer, Heidelberg (2005)
3. Büttcher, S., Clarke, C.L.A.: Index compression is good, especially for random access. In: *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM 2007*, pp. 761–770. ACM Press, New York (2007)
4. Chierichetti, F., Lattanzi, S., Mari, F., Panconesi, A.: On placing skips optimally in expectation. In: *WSDM 2008: Proceedings of the International Conference on Web Search and Web Data Mining*, pp. 15–24. ACM, New York (2008)
5. Dean, B.C., Jones, Z.H.: Exploring the duality between skip lists and binary search trees. In: *ACM-SE 45: Proceedings of the 45th Annual Southeast Regional Conference*, pp. 395–399. ACM, New York (2007)
6. Goldstein, J., Ramakrishnan, R., Shaft, U.: Compressing relations and indexes. In: *Proceedings of IEEE International Conference on Data Engineering*, pp. 370–379 (1998)
7. Messeguer, X.: Skip trees, an alternative data structure to skip lists in a concurrent approach. In: *ITA*, pp. 251–269 (1997)
8. Moffat, A., Zobel, J.: Self-indexing inverted files for fast text retrieval. *ACM Trans. Inf. Syst.* 14(4), 349–379 (1996)
9. Pugh, W.: Skip lists: a probabilistic alternative to balanced trees. *Commun. ACM* 33(6), 668–676 (1990)
10. Zukowski, M., Heman, S., Nes, N., Boncz, P.: Super-scalar ram-cpu cache compression. In: *ICDE 2006: Proceedings of the 22nd International Conference on Data Engineering*, p. 59. IEEE Computer Society, Washington, DC (2006)

# Weight-Based Boosting Model for Cross-Domain Relevance Ranking Adaptation\*

Peng Cai<sup>1</sup>, Wei Gao<sup>2</sup>, Kam-Fai Wong<sup>2,3</sup>, and Aoying Zhou<sup>1</sup>

<sup>1</sup> East China Normal University, Shanghai, China  
pengcai2010@gmail.com, ayzhou@sei.ecnu.edu.cn

<sup>2</sup> The Chinese University of Hong Kong, Shatin, N.T., Hong Kong  
{wgao,kfwong}@se.cuhk.edu.hk

<sup>3</sup> Key Laboratory of High Confidence Software Technologies, Ministry of Education, Beijing, China

**Abstract.** Adaptation techniques based on importance weighting were shown effective for RankSVM and RankNet, viz., each training instance is assigned a *target weight* denoting its importance to the target domain and incorporated into loss functions. In this work, we extend RankBoost using importance weighting framework for ranking adaptation. We find it non-trivial to incorporate the target weight into the boosting-based ranking algorithms because it plays a contradictory role against the innate weight of boosting, namely *source weight* that focuses on adjusting source-domain ranking accuracy. Our experiments show that among three variants, the *additive* weight-based RankBoost, which dynamically balances the two types of weights, significantly and consistently outperforms the baseline trained directly on the source domain.

## 1 Introduction

Learning to rank [4] is to derive effective relevance ranking functions based on a large set of human-labeled data. Boosting has been extensively studied for learning to rank [1,2,8,9]. However, existing ranking algorithms, including the boosting-based ones, are only proven effective for data from the same domain. In real applications, it is prohibitive to annotate training data for every search domain. Ranking performance may suffer when the training and test have to take place on different domains.

A promising direction is to learn a cross-domain adaptation model for ranking. Two key problems should be resolved: (1) how to measure the relatedness of two domains appropriately; (2) how to utilize this information in ranking algorithms for adaption. [3] adopted a classification hyperplane to derive the importance weight of documents in the source domain that reflects their similarity to the target domain and is then incorporated into the rank loss function.

When applying this method, we find it non-trivial to integrate importance weight into boosting-based algorithms such as RankBoost [2]. The reason is

---

\* This work is partially supported by NSFC grant (No. 60925008), 973 program (No. 2010CB731402) and 863 program (No. 2009AA01Z150) of China.

that these algorithms bear an inherently weight-based exponential loss and this innate weight in the loss function (*source weight*) plays a contradictory role with the importance weight introduced for adaptation purpose (*target weight*). An appropriate balance must be made between these two types of weight; otherwise adaptation may fail since the model can easily overfit source data due to the great impact of source weight on weak rankers selection.

In this work, we develop three Weight-based RankBoost (WRB) algorithms to balance the source and target weights, namely expWRB, linWRB and additive WRB (addWRB). The first two methods incorporate the target weight in straightforward and static ways, and the third combines the weights from a *global* perspective based on a forward stage-wise additive approach [6] to achieve a dynamic tradeoff. Our results demonstrate that addWRB consistently and significantly outperforms the baseline trained directly on the source domain.

## 2 Target Weight and Source Weight

### 2.1 Source Weight

RankBoost [2] aims to find a ranking function  $F$  to minimize the number of misordered document pairs. Given document pairs  $\{(x_i, x_j)\}$ , the ranking loss is defined as  $rLoss(F) = \sum_{i,j} W(x_i, x_j)I(F(x_i) \geq F(x_j))$ , where  $W(x_i, x_j)$  is the source weight distribution,  $I(\cdot)$  is an binary indicator function, and  $F(x_i) \geq F(x_j)$  suggests that the ranking function assigns a higher score to  $x_i$  than to  $x_j$  while the ground truth is  $x_i$  has a lower rating than  $x_j$ . At each round of training,  $W(\cdot)$  is updated for the next round to focus on those misordered pairs. The update formula in round  $t$  is given as follows:

$$W_{t+1} = \frac{1}{Z_t} W_t(x_i, x_j) \exp(\alpha_t(f_t(x_i) - f_t(x_j))) \quad (1)$$

where  $f_t(x)$  is the 0-1 valued weak ranker derived from a ranking feature  $x$ ,  $\alpha_t$  is the coefficient of  $f_t$  so that  $F = \sum_t \alpha_t f_t(x)$ , and  $Z_t$  is normalization factor. Inherently, source weight is designed to control the selection of weak rankers to minimize ranking errors in source domain.

### 2.2 Target Weight

In ranking adaptation, the knowledge of relevance judgement should be strengthened on those documents that are similar to the target domain so that the learning can be focused on correctly ranking these important documents. [3] used the cross-domain similarity to transfer ranking knowledge. The distance of a source-domain document to the classification hyperplane was calculated as target weight to measure the importance of the document. Then the pointwise weight was converted to pairwise for compatible with the popular pairwise approach (see [3] for details). The general loss term was extended as follows:

$$\sum_{i,j} w(x_i, x_j) * rLoss_{ij}(\cdot) \quad (2)$$

where  $rLoss_{ij}(\cdot)$  is the pairwise loss and  $w(x_i, x_j)$  is the target weight.

---

**Algorithm 1.** expWRB–Weighted RankBoost with target weight inside the exponent

---

**Input:** Query-document set of source domain; Target weights of  $M$  document pairs  $\{w(x_i, x_j)\}_1^M$  based on the ground truth.

**Output:** Ranking function  $F(x)$ .

1. Initialize  $W_1(x_i, x_j) = \frac{1}{M}$  for all  $i, j$ ;
  2. **for**  $t = 1$ ;  $t \leq T$ ;  $t++$  **do**
  3.   Select weak ranker  $f_t(x)$  using  $W_t$  and  $w$ ;
  4.   Set coefficient  $\alpha_t$  for  $f_t(x)$ ;
  5.   For each  $(x_i, x_j)$ , update source weight using  $W_{t+1} = \frac{1}{Z_t} W_t(x_i, x_j) \exp(\alpha_t w(x_i, x_j)(f_t(x_i) - f_t(x_j)))$ ;
  6. **end for**
  7. **return**  $F(x) = \sum_{t=1}^T \alpha_t f_t(x)$
- 

### 3 Weight-Based Boosting Models for Ranking Adaptation

In standard RankBoost, the source weight is updated iteratively so that the weak rankers can focus on those misordered pairs with large source weight that commonly reside near the decision boundary deemed more difficult to order. However, these pairs may be unnecessarily important to the target domain. Meanwhile, for those misordered pairs with low source weight, even though they contain some important cross-domain ranking knowledge (i.e., having high target weight), the algorithm does not prioritize to correct their ranking. The two types of weight play contradictory roles and must be appropriately balanced.

The objective of our adaptive RankBoost is to minimize the weighted ranking loss  $wLoss(F) = \sum_{i,j} W(x_i, x_j) I(F(x_i) \geq F(x_j)) w(x_i, x_j)$  following Eq. 2. There are two straightforward ways to incorporate the target weight into the source weight’s update formula (Eq. 1):

$$W_{t+1} = \frac{1}{Z_t} W_t(x_i, x_j) \exp(\alpha_t \boxed{w(x_i, x_j)} (f_t(x_i) - f_t(x_j))), \tag{3}$$

$$W_{t+1} = \frac{1}{Z_t} W_t(x_i, x_j) \boxed{w(x_i, x_j)} \exp(\alpha_t (f_t(x_i) - f_t(x_j))). \tag{4}$$

Thus, we obtain two versions of Weight-based RankBoost (WRB), namely expWRB corresponding to the target weight inside the exponent (Eq. 3) and linWRB corresponding to the linearly combined target weight (Eq. 4).

#### 3.1 expWRB

The procedure of expWRB is shown as Algorithm 1. Other than the updating of source weight in step 5, expWRB also differs from standard RankBoost in step 3 where both source and target weights are used to search for the objective function  $F$  to minimize the weighted rank loss.

In each round  $t$ , we can choose an appropriate  $\alpha_t$  and  $f_t(x)$  to minimize  $Z_t$  in step 3.  $Z_t$  is minimized by maximizing  $r_t = \sum_{i,j} W_t(x_i, x_j) w(x_i, x_j) (f_t(x_j) - f_t(x_i))$  and set  $\alpha_t = \frac{1}{2} \ln \frac{1+r_t}{1-r_t}$  in step 4 [2].

### 3.2 linWRB

Replacing the updating rule in step 5 in Algorithm 1 with Eq. 4, we can obtain linWRB with linearly combined target weight. Similarly, we minimize  $Z_t$  for minimizing the weighted loss in each round following 2. Given a binary weak ranker  $f_t(x) \in \{0, 1\}$  and  $a \in \{-1, 0, +1\}$ , we set  $R_a = \sum_{i,j} W(x_i, x_j)w(x_i, x_j)I(f_t(x_i) - f_t(x_j) = a)$ . Then  $Z_t = R_{+1} \exp(\alpha_t) + R_0 + R_{-1} \exp(-\alpha_t)$ .  $Z_t$  is minimized by setting  $\alpha_t = \frac{1}{2} \ln \frac{R_{-1}}{R_{+1}}$ . The weak ranker  $f_t$  is selected when  $Z_t$  is minimal.

### 3.3 Additive Weight-Based RankBoost

Standard RankBoost updates source weight in the round  $t + 1$  based on the current weak ranker  $f_t$  locally (see Eq. 1). A better way is to calculate  $W_{t+1}$  globally using the ensemble function  $F_t$  that combines all the weak rankers learned up to the current round like an additive approach 6. The update rule is given by  $W_{t+1} = \frac{1}{Z_t} \exp(F_t(x_i) - F_t(x_j))$  where  $F_t(x) = F_{t-1}(x) + \alpha_t f_t(x)$ , which eliminates the previous round source weight. Then the target weight can be incorporated straightforwardly as  $W_{t+1} = \frac{1}{Z_t} w(x_i, x_j) \exp(F_t(x_i) - F_t(x_j))$ .

However, the model easily overfits the source domain due to the great impact on the updating of source weight from the exponential term. We introduce a scaling factor  $\lambda$  to adjust the source weight dynamically. The idea is that we update  $\lambda$  considering ranking difficulty measured by the proportion of correctly ordered pairs in each round:

$$\lambda_t = \lambda_{t-1} * \frac{\# \text{ of correctly ordered pairs by } F_t}{\text{Total } \# \text{ of pairs to rank}} \quad (5)$$

In a difficult task where wrong pairs dominate,  $\lambda$  decreases quickly and cancel out the exponential growth of source weight so that target weight can affect weak ranker selection properly.

Based on this intuition, we propose the Additive Weight-based RankBoost (addWRB) given as Algorithm 2. A forward stagewise additive approach 6 is used to search for the strong ranking function  $F$ . That is, in each round, a weak ranker  $f_t$  is selected and combined with  $F_{t-1}$  using coefficient  $\alpha_t$ . The source weight is then updated in step 8, where the ensemble function is scaled by  $\lambda_t$  inside the exponent that is further combined with the target weight linearly.

## 4 Experiments and Results

Evaluation is done on LETOR3.0 benchmark dataset 5 with the Web track documents of TREC 2003 and 2004. We treat each ranking task, namely Home Page Finding (HP), Named Page Finding (NP) and Topic Distillation (TD) 7 as an individual domain. Generally, it is relatively easier to determine a homepage or named page than an entry point of good websites.

We use the same method as 3 to estimate target weights. Note that rank labels are not used for weighting. The *baseline* is a RankBoost directly trained

**Algorithm 2.** Additive Weight-based RankBoost (addWRB)

- 
1. Initialize  $W_1 = \frac{w(x_i, x_j)}{\sum_{i,j} w(x_i, x_j)}$  for all  $i, j$ ;
  2. Set  $\lambda_0 = 1, F_0 = 0$ ;
  3. **for**  $t = 1; t \leq T; t++$  **do**
  4.   Select weak ranker  $f_t(x)$  using distribution  $W_t$ ;
  5.   Set coefficient  $\alpha_t$  for  $f_t(x)$ ;
  6.    $F_t(x) = F_{t-1}(x) + \alpha_t f_t(x)$ ;
  7.   Compute  $\lambda_t$  using Eq. 5;
  8.   For each  $(x_i, x_j)$ , update source weight using  $W_{t+1} = \frac{1}{Z_t} w(x_i, x_j) \exp(\lambda_t (F_t(x_i) - F_t(x_j)))$ ;
  9. **end for**
  10. **return**  $F(x) = \sum_{k=1}^T \alpha_k f_k(x)$
- 

**Table 1.** MAP results of three adaptation tasks. †, ‡ and ‡ indicate significantly better than baseline, expWRB and linWRB, respectively (95% confidence level).

model	HP→NP		NP→TD			TD→NP	
	Y2003	Y2004	Y2003	Y2004	Y03-Y04	Y2003	Y2004
baseline	0.5834	0.5455	0.1734	0.1657	0.1062	0.4101	0.3061
expWRB	0.5481	0.5206	0.1352	0.1437	0.1485 <sup>†</sup>	0.5493 <sup>†‡</sup>	0.5159 <sup>†‡</sup>
linWRB	0.6245 <sup>†‡</sup>	0.5824 <sup>†‡</sup>	0.1755 <sup>‡</sup>	0.1444	0.1433 <sup>†</sup>	0.3344	0.2239
addWRB	0.6280 <sup>†‡</sup>	0.6025 <sup>†‡</sup>	0.2139 <sup>†‡‡</sup>	0.1505	0.1541 <sup>†</sup>	0.5537 <sup>†‡</sup>	0.5774 <sup>†‡</sup>

on the source domain without target weight. Always we leveraged on decision stumps to implement binary weak rankers.

We examined HP to NP, NP to TD and TD to NP adaptations to study if our algorithms can adapt across similar tasks, from easier to more difficult task and in the reverse case. The MAP results are reported in Table 1.

**HP to NP Adaptation.** We observe that addWRB outperforms all other algorithms. T-tests indicate that both addWRB and linWRB are significantly better than the baseline and expWRB ( $p < 0.02$ ). This indicates both algorithms can effectively balance the two types of weights.

Note that expWRB failed here. We found that lots of pairs were ordered correctly in HP training, resulting in small source weights. So the target weight inside the exponent quickly dominated the updating of source weight and the same weak ranker was chosen repeatedly. The model becomes not generalizable.

**NP to TD Adaptation.** NP is rather different from TD. On 2003 data, addWRB works better than other variants, and t-test indicates that the improvements are statistically significant ( $p < 0.001$ ). On 2004 data, all three variants underperform the baseline. This is consistent to [3] using other algorithms due to the shortage of training data from the source domain. Actually, only a half number queries are available in NP04 than NP03. To avoid under-training, we turned to examine NP03 to TD04 adaptation where our algorithms significantly outperformed the baseline ( $p < 0.001$ ).

**TD to NP Adaptation.** Here we study how our models can adapt from a difficult task to a simple one. We observe that expWRB and addWRB are significantly better than the baseline ( $p < 0.0001$ ) whereas linWRB fails. The target



weight affects linWRB little when source pairs are difficult to rank because of the exponential source weights. So the performance is mainly determined by source weight leading to the failing adaptation. In contrast, expWRB's target weight inside the exponent can effectively balance the growth of source weights.

**The Scaling Factor  $\lambda$ .** We also examine the influence of  $\lambda$  on 2003 data to unveil how  $\lambda$  reacts to different problem difficulty. We observe that  $\lambda$  in TD to NP is much lower and decreases much faster than NP to TD. Since TD is more complex where lots of pairs are wrong,  $\lambda$  decreases quickly to balance the exponential growth of source weights.

## 5 Conclusions

We proposed three variants of weight-based boosting models for ranking adaptation based on RankBoost algorithm, namely expWRB, linWRB and addWRB. The challenge is to balance the innate weight distribution of RankBoost and the target weight introduced for adaptation. expWRB and linWRB incorporate target weight in straightforward yet static ways. addWRB uses an additive approach, where the influence of source weight can be scaled dynamically according to the problem difficulty. Experiments demonstrate that the performance of expWRB and linWRB varies with the problem difficulty of source domain, and addWRB consistently and significantly outperforms the baseline.

## References

1. Amini, M.R., Truong, T.V., Goutte, C.: A boosting algorithm for learning bipartite ranking functions with partially labeled data. In: Proc. of SIGIR (2008)
2. Freund, Y., Iyer, R., Schapire, R., Singer, Y.: An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* (2004)
3. Gao, W., Cai, P., Wong, K.-F., Zhou, A.: Learning to rank only using training data from related domain. In: Proc. of SIGIR (2010)
4. Liu, T.-Y.: Learning to rank for information retrieval. In: *Foundations and Trends in Information Retrieval* (2009)
5. Qin, T., Liu, T.-Y., Xu, J., Li, H.: LETOR: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval* (2010)
6. Hastie, T., Tibshirani, R., Friedman, J.H.: *The elements of statistical learning*. Springer, Heidelberg (2001)
7. Voorhees, E.M.: Overview of TREC 2004. In: Proc. of TREC (2004)
8. Xu, J., Li, H.: Adarank: A boosting algorithm for information retrieval. In: Proc. of SIGIR (2007)
9. Zheng, Z., Zha, H., Zhang, T., Chapelle, O., Chen, K., Sun, G.: A general boosting method and its application to learning ranking functions for web search. In: Proc. of NIPS (2007)

# What Makes Re-finding Information Difficult? A Study of Email Re-finding

David Elswailer<sup>1</sup>, Mark Baillie<sup>2</sup>, and Ian Ruthven<sup>2</sup>

<sup>1</sup> Dept. Computer Science 8 (AI), University of Erlangen-Nuremberg, Germany

<sup>2</sup> Dept. Computer and Information Sciences, University of Strathclyde, Scotland  
david@elsweiler.co.uk, bailliem@gmail.com, ir@cis.strath.ac.uk

**Abstract.** Re-finding information that has been seen or accessed before is a task which can be relatively straight-forward, but often it can be extremely challenging, time-consuming and frustrating. Little is known, however, about what makes one re-finding task harder or easier than another. We performed a user study to learn about the contextual factors that influence users' perception of task difficulty in the context of re-finding email messages. 21 participants were issued re-finding tasks to perform on their own personal collections. The participants' responses to questions about the tasks combined with demographic data and collection statistics for the experimental population provide a rich basis to investigate the variables that can influence the perception of difficulty. A logistic regression model was developed to examine the relationships between variables and determine whether any factors were associated with perceived task difficulty. The model reveals strong relationships between difficulty and the time lapsed since a message was read, remembering *when* the sought-after email was sent, remembering *other recipients* of the email, the experience of the user and the user's filing strategy. We discuss what these findings mean for the design of re-finding interfaces and future re-finding research.

## 1 Introduction

Personal information management (PIM) as a research field covers efforts to understand PIM behaviour, the information strategies that people employ and attempts to develop systems that help people manage and re-find their information effectively. One PIM activity that has received growing research attention in recent times is information re-finding, where there has been a specific emphasis placed on improving the tools available to assist people with locating information that they have previously seen or accessed. In IR this has involved developing interfaces for Desktop Search [16] and creating testbeds to simulate re-finding behaviour and evaluate the performance of Desktop Search algorithms [12, 25].

In this paper we argue that while these aspects are important, before focusing too heavily on the development of improved tools, and before we can adequately simulate re-finding behaviour with test collections, we need an improved understanding of real behaviour. This includes learning about the re-finding tasks

that people perform, the user's perception of these tasks, the factors that affect this perception and how these factors influence the behaviour the user exhibits. We add to this understanding by analysing data collected from a study of email re-finding behaviour to determine the factors that influence the user's perception of task difficulty. Understanding what makes a re-finding task difficult offers researchers insight into which aspects of behaviour are important to study and where and how to provide appropriate user support. Our findings have implications for the design of re-finding tools and for the design of experiments used to understand behaviour and evaluate new tools.

## 2 Background Literature

This section describes related work that hints at factors that may influence user behaviour. Section 2.1 provides a summary of research relating to organising and management behaviours. In Section 2.2, the focus switches to information re-finding and research that has uncovered factors that may influence this activity. The highlighted factors form the basis of our data analyses described below.

### 2.1 Organising and Managing Behaviour

Much of the research on organising and managing personal information stems from Malone's seminal study of office information management behaviour [28]. Malone examined how office workers organised paper-based information and uncovered two principle strategies – filing and piling – which Malone related to the tidiness or messiness of the individual. The approach has since been replicated several times to study organisational habits for different kinds of information objects, including email messages, where user strategies include frequent filing, not filing and spring-cleaning [34], as well as web bookmarks [1] and computer files [7]. Boardman and Sasse [7] investigated the similarities and differences between the organisational techniques used to manage email messages, files and web bookmarks. Although they found some examples of overlap, particularly that pro-organising participants tended to be pro-organising for all of their collections and the opposite finding for non-organising participants, the way that collections were organised were, in the main, fundamentally different for different types of information object. This finding reveals that the organisational strategy applied depends not only on characteristics of the individual user, but also on the type of information being organised. Studies of classification decisions in offices have provided similar findings. Cole [13] and Case [11] both uncovered examples where information objects were classified not primarily by their semantic content, but by the physical form of the object e.g. journals were often organised differently to books. Kwasnik [26] demonstrated that in addition to these document attributes (topic, author and form), there are situational factors, such as the document's source or intended use that influence how it may be classified.

An important feature of previous research is that it demonstrates that different factors influence the way that people behave when managing their information. As pointed out by Gwizdka and Chignell [22], these factors can be at the

level of the individual user, e.g., their “tidiness” or “messiness”; at the level of users groups, e.g., people who have different types of job or apply different management strategies; or factors relating to the context or task being performed, e.g. the type of media being managed or whether in a paper-based or digital environment.

## 2.2 Re-finding Behaviour

There is an increasing body of evidence suggesting that people regularly need to re-access and re-use information they have accessed in the past [16,14,32]. Investigations have revealed that people perform three main types of re-finding tasks: lookup tasks, item tasks and multi-item tasks [20]. Lookup tasks involve searching for specific information from within a resource, for example finding a password from within an email. Item tasks involve looking for a particular item, perhaps to pass on to someone else or when the entire contents are needed to complete the task. Multi-item tasks involve finding more than one item and often require the user to process or collate information in order to solve the task. The evidence suggests that re-finding tasks can often be difficult, time consuming and frustrating [2,7,8,32]. No-one to date has compared the different factors which could make a task difficult in a single experiment. There is fragmentary evidence in the literature, however, as to which factors could play a role.

**Task-Context-level factors:** In a study of the factors that affect web page re-finding, Capra and Perez-Quinones [10] discovered that both the frequency with which a task is performed and the familiarity with the information source used to solve the task had an influence on method by which the participants chose to re-find. Elswailer and Ruthven [20] discovered a link between the time period that had elapsed since the sought-after information had been last accessed and how difficult the re-finding task was perceived to be. Barreau and Nardi [6] showed that people prefer to re-find older information in different ways to information that is new or used regularly. Their participants generally used search-based strategies to re-find older, archived information and browse-based strategies to re-find working or ephemeral information. Other studies suggest that time may have less of an impact on re-finding strategies, finding that the path originally taken to get to the information target is much more important [31,9].

**Group-level factors:** The way people organise and store their information has been shown to influence peoples’ behaviour. Elswailer et al. [18] demonstrated that the filing strategy that a participant employed influenced what they tended to remember about the email messages they were asked to find. Similarly, in studies of file re-finding [6] and web page re-finding [32], the participants who employed different filing strategies tended to use different retrieval strategies in different contexts.

**Individual-level factors:** Baelter [5] found that there was a link between the number of emails in a collection and the time taken to file and retrieve them. The re-finding experience of the user may also be important. Studies of search

behaviour have noted differing behaviour and differing success rates for novice and experienced users [24,29] and similar outcomes have been established for re-finding [2].

Therefore, according to previous research, there are many potential factors that may influence how difficult re-finding information will be, although the quality of evidence varies for different factors. To determine which of these variables have the largest influence and to understand the relationships between these variables, we describe a study of email re-finding, which takes into account many of variables outlined above. We analyse the data with the aim of establishing the factors, which influence the perception of email re-finding task difficulty.

### 3 Materials and Methods

The decision to focus on email re-finding tasks was taken for a number of reasons. First, log-based studies of re-finding behaviour have shown that people have to re-find email messages more than any other kind of information object [16,14]. Second, although email is an asynchronous communication tool, it is, in practice, used for collaborative working [15], data archiving [34,5] and for task [34] and contact management [33]. Conflicting strategies for different uses of email suggests that re-finding emails could be particularly challenging and it is of interest to uncover the factors that influence this.

#### 3.1 Experimental Design

We performed a controlled user study which examined the participants' perception of the difficulty of tasks when they were required to re-find email messages. The study population included 21 participants from a well-known British university, consisting of a mix of academic and research staff, undergraduate computer science students and a post-graduate class with a variety of undergraduate academic backgrounds, including former business, geography, modern-languages and philosophy students. The participants had been using their collections for varying time periods, with the post-grads having relatively new collections (the average age of collection was less than 3 months) and the academic staff comparatively older collections (avg. age ~3years). Reflecting this, some of the collections contained few messages (min = 95) and others several thousand (max = 8954, median = 5132). The participants also reported using email for different purposes. While the students tended to use email mainly for class announcements and collaborative working, the academics used email for a wide range of purposes, including task and contact management, data storage, version control, collaborative authoring, as well as simple communication.

We went to great lengths to establish realistic re-finding tasks for participants that could be performed on their own personal collections without invading individual privacy. This was achieved following the methodology proposed by [20] and involved performing a number of preliminary studies with the participants

and their peers, including interviews, collection tours and diary studies of the re-finding tasks people in these groups perform. This work allowed us to establish a pool of experimental tasks suitable for different groups of users, reflecting the contents of their collections and simulating the kinds of re-finding tasks they may perform in a naturalistic setting. Example tasks involved discovering when the security code for the computer labs was last changed, finding specific details from class announcements and finding emails regarding department events or activities<sup>[1]</sup>.

Each participant was allocated 9 tasks from these pools including 3 lookup, 3 item and 3 multi-item tasks, which were rotated appropriately. After each task was issued the participant was asked to subjectively rate the task in terms of difficulty on a 5-point scale (very easy, easy, okay, difficult, very difficult), before performing the task on his own personal collection. There was a good mix of task difficulties (median = 2, interquartile range (IQR) =1). Some of the tasks were perceived to be quite easy, while others were considered challenging. These ratings form the basis of our analyses in this paper. To determine the variables which influenced the perceived difficulty of a task we also asked a range of further questions including how long it had been since the participant had previously accessed the email to be found and the attributes of the email that the participant could remember. The recorded task data combined with the demographic data and collection statistics for the experimental population provided a rich basis to investigate the variables influencing the perception of difficulty of email re-finding tasks<sup>[2]</sup>.

### 3.2 Analysing the Data

A logistic regression model was developed to determine whether any factors were associated with perceived difficulty score assigned by the participants. All available factors (24 in total) collated from the user study were analysed initially using a stepwise procedure in order to isolate any significant relationships. The stepwise procedure automatically enters and removes factors at each step assessing the overall goodness of fit of the linear regression model ([30] provide an overview on generalized linear models and the stepwise procedure).

Following this initial investigation, we considered specific factors of interest (see Table 1) alongside those factors included in the model during the stepwise procedure. The reasoning behind this decision was that automatic model building procedures can be erroneous as it does not consider the real world importance of each factor. Therefore, the final model presented also included those factors believed to be important based on previous research (described in Section 2). These factors were entered into the model to assess both their effect (if any) on perceived difficulty and also the relationship between these factors and the

<sup>1</sup> [20] provides detailed descriptions of the process we followed and the experimental tasks can be found in [17].

<sup>2</sup> The full questionnaires can be found in [17]. Further details of the experimental design and user population can be found in our previous publications [18][19].

**Table 1.** Specific Factors of Interest

Task or context factors:
The type of re-finding task (look-up vs item vs multi-item [20])
The period of time since the information was last accessed (<week (hot), <month (warm) >month(cold))
How frequently the information is accessed by the participant
If the participant remembered when the information was sent
If the participant remembered other people associated with the information
If the participant remembered what the email was about or why it was sent
If the participant remembered other people associated with the email
The number of memory attributes remembered by the participant
Group factors
Filing strategy (filers vs no-filers vs spring-cleaners)
Re-finding strategy (preference for browsing vs preference for searching vs no preference)
User group (Undergraduate Students vs Postgraduate Students vs Research Staff)
Individual factors
Number of emails in collection
Re-finding experience
Age of oldest message in collection

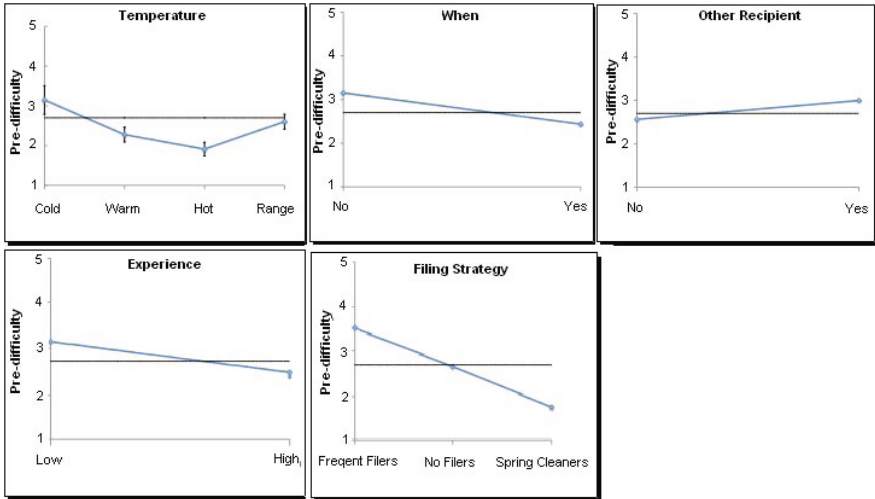
**Table 2.** Regression model for perceived difficulty (significant factors in bold)

	Coefficient estimate	Std. Error	Z value	Pr(>  Z )
(Intercept)	2.70	0.36	7.49	< 0.01
Experience: Low	-0.25	0.15	-1.67	0.10
<b>Filing.group: No filers</b>	<b>-0.54</b>	<b>0.21</b>	<b>-2.50</b>	<b>0.01*</b>
<b>Filing.group: Spring cleaners</b>	<b>-0.61</b>	<b>0.21</b>	<b>-2.89</b>	<b>&lt;0.01*</b>
<b>Other recipients: Yes</b>	<b>0.43</b>	<b>0.13</b>	<b>3.30</b>	<b>&lt;0.01*</b>
<b>Temperature: Warm</b>	<b>-0.42</b>	<b>0.19</b>	<b>-2.26</b>	<b>0.03*</b>
<b>Temperature: Hot</b>	<b>-0.79</b>	<b>0.17</b>	<b>-4.56</b>	<b>&lt;0.01*</b>
Temperature: Range	-0.09	0.19	-0.50	0.62
When: Yes	-0.26	0.14	-1.88	0.06

other remaining variables found to be significant. The final model is presented in Table 2 and the effect plots for the models are shown in Figure 1, with the regression coefficient, the standard error, Z value and p-value presented for all factors included. In the effect plots the solid line represents the mean perceived difficulty value and the probability of the sample assigning a value higher than the mean irrespective of the factor. Therefore, if the factor level is above the line this represents a positive effect and vice versa.

## 4 Results

The model demonstrates the main factors that influenced the perception of difficulty associated with the allocated tasks. Three of the five variables that feature in the model are at the task-context level. The temperature of the task, i.e. how long it had been since the required information had been accessed, was a significant factor in the model. Hot tasks involved finding information that had been accessed in the previous week, warm tasks information from the previous month and cold, information not accessed for longer than a month. The category range means that multiple messages were re-found and more than one temperature



**Fig. 1.** Main effect plots for regression model for perceived difficulty

category was appropriate. Intuitively, the data show that longer the time period between accessing and re-finding, the more difficult the task was perceived to be. Other important factors at the task-context-level include two memory variables. When the participant remembered *when* the required email was sent, the task tended to be perceived as being easier. This also makes sense given that this information could be used directly as part of a re-finding strategy. A less intuitive feature of the model was that when the participants thought they were able to remember *other recipients* of an email, the task tended to be perceived as more difficult. It seems to make little sense that remembering extra information about a email – information that could be used to re-find the mail – could lead to increased task difficulty. However, there are possible explanations for this outcome. It could be, for example, that when other recipients are remembered, such as remembering that a mail was sent to every member of the department or company, that this is an indicator of a less personal connection with the email, leading to a poorer overall recollection. Secondly, if emails are regularly sent to these recipients, as was often the case in our tasks, the remembered information is of little benefit in the re-finding process.

The one user group-level factor that featured in the developed model was filing strategy. The participants who employed a frequent-filing strategy tended to rate re-finding tasks as harder than the user groups who utilised other filing strategies. The participants who rated tasks as easiest were the spring-cleaners, who generally do not file, but periodically, when the inbox becomes too large, tidy up messages into appropriate folders. Again, initially, this finding makes little sense – the people who make a conscious effort to organise their mails for future retrieval in reality find tasks to be more difficult. Further, one could imagine that a spring-cleaning strategy could lead to confusion or doubt when



remembering if an email was stored in a folder or not. Nevertheless, the finding aligns with previous work, which found those who file frequently to have the poorest recollection for information they need to re-find whereas spring-cleaners had the best recollection [18].

The only factor in the model that was at the level of the individual user was experience. As would be expected, participants with more experience with email tended to rate tasks as less difficult. However, although featuring in the model, this factor was not significant ( $p=0.1$ )

There are some variables that are notably absent from the model. For example, despite the numerous differences between the user groups in the experiment and the vast differences in terms of collection sizes, we found no evidence for a correlation between either of these variables and the perceived difficulty of the task. This shows that despite many of the participants having new collections with relatively few messages, there were still tasks which they perceived as difficult. It is possible that experience - a variable that did feature in the model - played a role here, with experience and developed strategies compensating against increased collection sizes. Another point of note was that there was no evidence that our participants found any type of task, i.e., lookup, item or multi-item to be especially difficult.

## 5 Implications

We believe the findings have important implications for future re-finding research. First, they provide indicators as to where researchers should focus their attention, highlighting particular situations or people requiring support. Second, they provide insight into how research should be performed, with implications for the design of laboratory-based user studies. We continue to outline our thoughts on these issues below.

### 5.1 Situations to Support and Study

The findings suggest that specific support is required when the user is looking for older information. Aligning with the findings of past studies [6,20], our data show a clear correlation between the length of time that has passed since the participant previously accessed the message and the perception of the difficulty of the task. We know from Desktop Search Engine query logs that people seldom query based on date [16] and there are few other alternatives for the user to indicate that he is looking for information not seen for some time. If the user was able to communicate this information to the system, specialised support could be provided to ease the re-finding of older information. There is a need for such support and researchers should be investigating ways of providing it, either through specific interface features or search algorithms.

Our data also indicate that particular groups of users would benefit from specific re-finding support. Inexperienced participants, despite having much smaller collections, still tended to perceive tasks as more difficult than experienced participants who had collections several times larger. Observing the participants during the completion of the tasks revealed that inexperienced participants lacked the honed strategies that more experienced participants had developed over time. This is a problem not limited to re-finding and has been noted many times in the IR literature, e.g., [2,3,23]. The key question that needs addressing both in IR and PIM is how do we design interfaces that both support the user to use the system with his current abilities and actively encourage and progress the development of advanced user strategies?

A second group of users who, according to our findings, would benefit from additional support are those who file frequently. People file emails for many reasons [34] so rather than encouraging these users to adopt different behaviours, what is needed is an understanding of the reasons for Frequent-filers perceiving tasks as more difficult so that appropriate support can be provided. Elswailer et al. [18] proposed that the reason this group of users tend to remember less about their mails is that act of filing removes the message from its context and the message is thereafter seldom interacted with. This means that there is no reminder of the message and its content through natural interaction with the system. One means of addressing this problem would be to provide filers with the ability to toggle between different views of their collection. One view could show mails in their folder structure and another could show mails in a temporally ordered stream, perhaps colour-coded to indicate the folder it would normally reside in. Such a view could be filtered by various attributes as in [14] and therefore allow increased interaction with filed messages without losing the benefits offered by folders.

This point and the fact that two memory variables were shown to be significant factors in our model emphasizes the importance of memory and recollections to the process of re-finding. Memory has been studied in the PIM community e.g. [27,18,21]. Nevertheless, there is still much to learn regarding the role of memory and how it influences user behaviour and indeed how behaviour influences memory. Our discovery in this study that remembering other mail recipients was associated with more difficult tasks shows that remembering more is not necessarily always a positive sign. It would be interesting to learn about the relationship between memory and re-finding behaviour, e.g., the queries a user submits to a Desktop Search Engine. Is there an obvious mapping between recollection and behaviour and can this relationship be improved?

## 5.2 Performing Re-finding Experiments

One problem when performing lab-based user studies to investigate re-finding is the number of variables that need to be controlled or managed in the design and data analysis phases. Our findings suggest that some variables are more important than others to consider. To achieve a balanced set of tasks in terms of task difficulty, for example, it may be more effective to control the task

temperature (hot vs warm vs cold) rather than the type of task (lookup, item and multi-item) as we did. The wording of task descriptions is also important as the information provided will determine how much the participants remember when they go to complete the task. As remembering *when* an email was sent tended to be associated with easier tasks, to help create a set of tasks with a range of difficulties, we suggest that experimenters omit temporal cues from their task descriptions. This will help ensure difficult tasks are included, as it is unlikely that all participants will remember this information organically [18].

This work emphasizes that the outcomes of such studies are influenced by several variables, many of which will be confounded. We would recommend, based on our findings, that when analysing the results of any system evaluation, i.e., testing whether one system performs better than another in a user evaluation, researchers should specifically examine the data for any possible influences of the variables featured in our model, i.e., task temperature, memory variables, user filing strategy and user experience.

## 6 Conclusions and Future Work

To summarise, in this paper we have described a user study investigating the factors which lead to re-finding tasks being perceived as difficult, finding clear relationships between particular contextual variables and perceived task difficulty. While this work only provides understanding regarding one aspect of tasks, it opens up several important questions for further study. We have outlined our thoughts on how our findings should influence future work in terms of what researchers should be investigating and also in terms of how experimental designs and data analyses can be optimised.

In terms of our own future work, we are currently working on some of the ideas we discussed in Section 5.2. We are looking to design interfaces to support the re-finding of older information and for users who file. We are also building on these findings to improve user simulations for automated Desktop Search evaluation. Given the relationships between particular variables and task difficulty ratings, it is possible that different behaviour will be exhibited in these situations as has been shown previously for web search engine behaviour. In [3] when users were faced with a difficult task, they started to formulate more diverse queries, used advanced operators more, and spent longer on the result page. If similar behavioural changes occur when performing difficult re-finding tasks then it would have important consequences for the way behaviour is simulated in automated evaluation approaches for Desktop Search. Current approaches, e.g., [4,25] apply generalised techniques to generating queries. If we can discover different query characteristics in different situations, e.g., for difficult tasks, we can use the findings to seed improved simulations and allow the performance of algorithms to be tested in different situations, such as the re-finding of older information problem outlined above. We believe, therefore, our findings here have many important implications for future re-finding research.

## References

1. Abrams, D., Baecker, R., Chignell, M.: Information archiving with bookmarks: Personal web space construction and organization. In: *Proceedings of Human Factors in Computing Systems (CHI 1998)*, pp. 41–48. ACM Press, New York (1998)
2. Aula, A., Jhaveri, N., Kki, M.: Information search and reaccess strategies of experienced web users. In: *Proc. 14th Int'l Conf. World Wide Web*, pp. 583–592. ACM Press, New York (2005)
3. Aula, A., Khan, R.M., Guan, Z.: How does search behavior change as search becomes more difficult? In: *CHI 2010: Proceedings of the 28th International Conference on Human Factors in Computing Systems*, pp. 35–44. ACM, New York (2010)
4. Azzopardi, L., de Rijke, M., Balog, K.: Building simulated queries for known-item topics: an analysis using six european languages. In: *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 455–462. ACM, New York (2007)
5. Bälter, O.: Keystroke level analysis of email message organization. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 105–112. ACM Press, New York (2000)
6. Barreau, D.K., Nardi, B.: Finding and reminding: File organization from the desktop. *ACM SIGCHI Bulletin*. 27(3), 39–43 (1995)
7. Boardman, R., Sasse, M.A.: stuff goes into the computer and doesn't come out: a cross-tool study of personal information management. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 583–590. ACM Press, New York (2004)
8. Bruce, H., Jones, W., Dumais, S.: Keeping and re-finding information on the web: What do people do and what do they need? In: *Proceedings of the 67th ASIST Annual Meeting (October 2004)*
9. Capra, R.G., Perez-Quinones, M.A.: Re-finding found things: An exploratory study of how users re-find information. Technical report, Computer Science Dept., Virginia Tech. (2003)
10. Capra, R.G., Perez-Quinones, M.A.: Using web search engines to find and refind information. *Computer* 38(10), 36–42 (2005)
11. Case, D.O.: Conceptual organization and retrieval of text by historians: The role of memory and metaphor. *JASIST* 42(9), 657–668 (1991)
12. Chernov, S., Serdyukov, P., Chirita, P., Demartini, G., Nejdl, W.: Building a desktop search test-bed. In: *Proceedings of the 29th European Conference on IR Research*, pp. 686–690 (2007)
13. Cole, I.: Human aspects of office filing: Implications for the electronic office. In: *Proceedings Human Factors Society* (1982)
14. Cutrell, E., Robbins, D., Dumais, S., Sarin, R.: Fast, flexible filtering with phlat. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 261–270. ACM Press, New York (2006)
15. Ducheneaut, N., Bellotti, V.: E-mail as habitat: an exploration of embedded personal information management. *Interactions* 8(5), 30–38 (2001)
16. Dumais, S., Cutrell, E., Cadiz, J., Jancke, G., Sarin, R., Robbins, D.C.: Stuff i've seen: a system for personal information retrieval and re-use. In: *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 72–79 (2003)

17. Elswailer, D.: Supporting Human Memory in Personal Information Management. PhD thesis, The University of Strathclyde (2007)
18. Elswailer, D., Baillie, M., Ruthven, I.: Memory and email re-finding. ACM TOIS CFP special issue on Keeping, Re-finding, and Sharing Personal Information (2008)
19. Elswailer, D., Baillie, M., Ruthven, I.: On understanding the relationship between recollection and re-finding. *Journal of Digital Information* 10(5) (2009)
20. Elswailer, D., Ruthven, I.: Towards task-based personal information management evaluations. In: *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 23–30. ACM Press, New York (2007)
21. Gonçalves, D., Jorge, J.A.: Describing documents: what can users tell us? In: *Proceedings of the 9th International Conference on Intelligent User Interfaces*, pp. 247–249. ACM, New York (2004)
22. Gwizdka, J., Chignell, M.: Individual differences. In: *Personal Information Management*, pp. 206–220. University of Washington Press, Seattle (2007)
23. Hölscher, C., Strube, G.: Web search behavior of internet experts and newbies. In: *Proceedings of the International WWW Conference*, pp. 337–346 (2000)
24. Hsieh-Yee, I.: Effects of search experience and subject knowledge on the search tactics of novice and experienced searchers. *JASIST* 44(3), 161–174 (1993)
25. Kim, J., Croft, W.B.: Retrieval experiments using pseudo-desktop collections. In: *Proceeding of the 18th ACM Conference on Information and Knowledge Management*, pp. 1297–1306. ACM, New York (2009)
26. Kwasnik, B.H.: How a personal document's intended use or purpose affects its classification in an office. *SIGIR Forum* 23(SI), 207–210 (1989)
27. Lansdale, M., Edmonds, E.: Using memory for events in the design of personal filing systems. *International Journal of Man-Machine Studies* 36(1), 97–126 (1992)
28. Malone, T.W.: How do people organize their desks?: Implications for the design of office information systems. *ACM Trans. Inf. Syst.* 1(1), 99–112 (1983)
29. Marchionini, G., Lin, X., Dwiggin, S.: Effects of search and subject expertise on information seeking in a hypertext environment. In: *Proceedings of the 53rd Annual Meeting of the American Society for Information Science* (1990)
30. McCullagh, P., Nelder, J.A.: *Generalized Linear Models*, 2nd edn. Chapman & Hall/CRC (1989)
31. Teevan, J.: How people re-find information when the web changes. Technical report, MIT AI (June 2004)
32. Teevan, J., Alvarado, C., Ackerman, M.S., Karger, D.R.: The perfect search engine is not enough: a study of orienteering behavior in directed search. In: *Proceedings of SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, USA, pp. 415–422 (2004)
33. Whittaker, S., Jones, Q., Terveen, L.: Contact management: Identifying contacts to support long term communication. In: *Proceedings of Conference on Computer Supported Cooperative Work*, pp. 216–225. ACM Press, New York (2002)
34. Whittaker, S., Sidner, C.: Email overload: exploring personal information management of email. In: Tauber, M.J. (ed.) *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 276–283. ACM Press, New York (1996)

# A User-Oriented Model for Expert Finding

Elena Smirnova<sup>1</sup> and Krisztian Balog<sup>2,\*</sup>

<sup>1</sup> INRIA Sophia Antipolis - Méditerranée, France  
elena.smirnova@inria.fr

<sup>2</sup> ISLA, University of Amsterdam, The Netherlands  
k.balog@uva.nl

**Abstract.** Expert finding addresses the problem of retrieving a ranked list of people who are knowledgeable on a given topic. Several models have been proposed to solve this task, but so far these have focused solely on returning the most knowledgeable people as experts on a particular topic. In this paper we argue that in a real-world organizational setting the notion of the “best expert” also depends on the individual user and her needs. We propose a user-oriented approach that balances two factors that influence the user’s choice: time to contact an expert, and the knowledge value gained after. We use the distance between the user and an expert in a social network to estimate contact time, and consider various social graphs, based on organizational hierarchy, geographical location, and collaboration, as well as the combination of these. Using a realistic test set, created from interactions of employees with a university-wide expert search engine, we demonstrate substantial improvements over a state-of-the-art baseline on all retrieval measures.

## 1 Introduction

Expert finding addresses the task of identifying the right person with the appropriate skills and knowledge [6]. Experts can be required for a variety of purposes: problem solving, question answering, providing more detailed information on a topic, to name a few. The expert finding task has attracted a great deal of interest within the Information Retrieval (IR) community over the past few years [3, 4, 8, 11, 12, 28]. The main focus has been on developing content-based algorithms, similar to document search. State-of-the-art expertise retrieval algorithms identify experts based on the content of documents that they are associated with [6, 19, 26]. While these approaches have been very effective in finding the most knowledgeable people on a given topic, they abstract away from the actual user performing the search. Such abstractions are common in IR, especially at world-wide evaluation campaigns, like TREC, CLEF, or INEX, as they help to simplify the process of building reusable test collections. On the other hand, the abstraction of the expert finding task, as defined at the TREC Enterprise track [8, 11], ignores important aspects that may play a role when people locate

---

\* Current affiliation: Department of Computer and Information Science, Norwegian University of Science and Technology, Norway, krisztian.balog@idi.ntnu.no

and select experts. Behavioral studies of human expertise seeking have found that, besides topical knowledge, the factors most influencing the selection process are related to the time needed to contact a person—including accessibility within the organizational hierarchy and workload [1, 21, 29].

In this paper we focus on two factors that influence the user’s choice: (i) time to contact a person and (ii) the person’s knowledge about the topic, relative to that of the user, seeking for expertise. We propose a user-oriented model that is based on *rational user behaviour* [2]; it assumes that, when choosing an expert, individuals will balance between the time needed to contact an expert and the knowledge gain. To the best of our knowledge, this is the first study that utilizes a user-centered approach for expert search. This approach provides a general framework for expertise retrieval that encompasses existing expert finding methods as a special case, namely, where the cost of contacting a candidate expert is ignored. We define knowledge gain as the relative difference between the expertise levels of the user and a candidate expert on a given topic; this is estimated using an existing state-of-the-art content-based approach. Contact time between two people—the user and a candidate expert—is approximated using their distance in a social network. Specifically, we consider social graphs based on (i) organizational hierarchy, (ii) geographical location, and (iii) collaboration (co-authorship), as well as the combination of these three structures.

We perform evaluation using real user queries and graded relevance judgments, obtained from the interactions of employees with a university-wide expert search engine [18]. We demonstrate substantial improvements on all retrieval measures with respect to a knowledge-only baseline. Our model involves a single parameter that can be used to adjust the user’s preferences for knowledge gain versus contact time. We explore a range of values for this parameter and find that configurations with slightly more weight on contact time perform best.

## 2 Related Work

Expert finding has been addressed from different viewpoints, including *expertise retrieval*, which takes a mostly system-centered approach, and *expertise seeking*, which studies related human aspects [16]. Besides textual data, *social networks* have also been used for finding experts. Finally, improving the user’s search experience by providing customized results touches on the field of *personalization*.

*Expertise Retrieval.* To reflect the growing interest in entity ranking in general and expert finding in particular, TREC introduced an expert finding task at its Enterprise track in 2005 [11]. At this track it emerged that there are two principal approaches to expert finding [3, 8, 11, 28]. The two models have been first formalized and extensively compared by Balog et al. [5], and are called *candidate* and *document* models, or *Model 1* and *Model 2*, respectively. Candidate-based approaches (also referred to as profile-based [14] or query-independent [23] methods) build a textual (usually term-based) representation of candidate experts, and rank them based on a query/topic, using traditional ad-hoc retrieval models. The other type of approach, document-based models (also referred to as

query-dependent approaches [23]), first find documents which are relevant to the topic, and then locate the associated experts. Nearly all systems that took part in the 2005–2008 editions of the Expert Finding task at TREC implemented (variations on) one of these two approaches. Building on either candidate or document models, further refinements to estimating the association of a candidate with the topic of expertise have been explored. For example, instead of capturing the associations at the document level, they may be estimated at the paragraph or snippet level [7, 20, 24]. Other extensions incorporate additional forms of evidence through the use of priors [14], document structure [33], hierarchical, organizational, and topical context and structure [6, 23], and Web data [25].

*Expertise Seeking.* Several studies have identified factors that may play a role in decisions of which expert(s) to select or recommend; most importantly, factors related to the person’s expertise level, social and physical proximity [1, 9, 21, 27]. In [21] the importance of the accessibility of an expert is pointed out—people prefer to contact those who are physically or organizationally close. Shami et al. [27] find that users prefer to contact the persons they know, even when they could receive potentially more information from people located outside their social network. Hofmann et al. [16] include additional factors besides topical knowledge (including organizational structure, position, experience, reliability, up-to-dateness, and contacts) into a system that recommends similar experts. Our work is different from [16] both in the task and in that we take the social distance between the user and the expert into account.

*Social Networks.* The use of social information for expert search has mainly been investigated from two directions. One uses graph-based measures (such as HITS or PageRank) on social networks, extracted from email communications, to produce a ranking of experts [10, 13]. Alternatively, others assume homogeneity among neighbours in a social network (based on co-authorship or organizational hierarchy) and define a smoothing procedure to relevance-based expert scores [17, 32]. It is important to mention that in this paper we consider not only people, but also other types of entities (organizational units and geographical locations) as nodes.

*Personalized Search.* Providing personalized results for users, depending on their information needs, has attracted significant interest in recent years. User modeling is one approach to search personalization; it can affect search in different phases: during the retrieval process, as a re-ranking step or in the query pre-processing procedure [22]. Adding a personalization component to the retrieval process can considerably slow down the system and query expansion does not guarantee to affect the result list. Re-ranking, on the other hand, remains a relatively cheap and reliable way to improve search quality; Google Personalized Search provides a large-scale example of this approach [30]. In this work we take a re-ranking approach to personalization for an expert finding system.



### 3 Baseline Model

Probabilistic approaches to expertise retrieval have been both popular and successful at the TREC Enterprise track [3, 8, 11, 28]. We take one of the most widely used models as our baseline, the so-called “Model 2” approach, proposed by Balog et al. [5]. Model 2 is attractive because it is an easy-to-understand, yet very effective method, with solid theoretical foundations. The key idea behind the model is to simulate how a user may search for experts using a standard document search engine: first, finding documents which are relevant, and then, examining each of these documents for associated persons. By scanning through a number of documents the user can obtain an idea of which people are more likely to be experts on the query topic [6].

Formally, the task of expert finding is stated as estimating the probability of an expert  $e$  to be knowledgeable on a given query topic  $q$ :  $p(e|q)$ . After applying Bayes’ rule, this problem transforms into estimating the probability of a query given an expert,  $p(q|e)$ ; assuming uniform prior on experts,  $p(e)$ :

$$p(e|q) \propto p(q|e) \cdot p(e). \quad (1)$$

The generative language modeling technique naturally follows from here as a way to estimate the probability of query terms to be generated by the expert. Under the Model 2 approach, the degree of topical association between an expert and a query,  $p(q|e)$ , is estimated as a weighted sum of relevance scores of documents  $p(q|d)$  related to an expert:

$$p(q|e) = \sum_d p(q|d) \cdot p(d|e). \quad (2)$$

Weights  $p(d|e)$  express the degree of association between a document and an expert. Probabilities  $p(q|d)$  are provided by the document language model, using the standard language modeling approach [31]. Since uniform priors were assumed, Eq. 2 provides the final ranking of expert candidates.

### 4 Experimental Setup

The main research question we aim to answer is this: Does modeling the user, seeking for experts, lead to improvements over a strictly content-based method?

*Data Collection.* The data set we use for experimental evaluation is the UvT Expert Collection [6]. It was collected from a publicly accessible database of employees involved in research or teaching at Tilburg University (UvT), The Netherlands. The data set comprises of four types of documents: research descriptions, course descriptions, publications, and personal homepages—in total 38,422 documents. The collection contains information about 1,168 experts, including their geographical location (building and room no.) and position within the organizational hierarchy. The UvT collection is bilingual (English/Dutch), but we did not resort to any special (language-dependent) treatment of the documents, apart from removing a standard list of English/Dutch stopwords.

*Topics and assessments.* In recent work, Liebrechts and Bogers [18] performed the evaluation of an expert finding system developed for UvT. As part of their study, 30 randomly selected UvT researchers were asked to formulate queries on which they themselves were knowledgeable. These participants were then asked to rate their and their colleagues' relative expertise levels on this area on a five-point scale (from 0 = "no expertise" to 4 = "high expertise"). This resulted in a "user-centered" set of 30 queries and 268 graded and realistic relevance judgements; we use this topic set for experimental evaluation<sup>1</sup>.

*Evaluation Measures.* We use standard Information Retrieval measures, namely, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and precision at rank 5 (P@5). We also report on a standard preference measure, Normalized Discounted Cumulative Gain at the number of relevant experts (NDCG@R), which emphasizes the quality at the top of the ranked list.

*Associating people and documents.* To compute the baseline model (Eq. 2) it is necessary to estimate the degree of association between an expert and a document,  $p(d|e)$ . Since each UvT document describes a particular expert's activity (research, teaching, publication), these associations can be established unambiguously. Following [6] we set  $p(d|e)$  to 1 if person  $e$  is an author of document  $d$ , and to 0 otherwise. The baseline model was computed using the EARS toolkit<sup>2</sup>.

## 5 User-Oriented Model

In this section we introduce our user-oriented model for expert finding that is based on *rational user behaviour* [2]. Under this approach, a "rational" user  $u$  is optimizing between the time needed to contact an expert  $e$ :  $time(e|u)$ , and the expertise level or knowledge of that expert relative to the expertise of the user (that is, the knowledge gain) given a particular query  $q$ :  $knowledge(e|u, q)$ . We combine these two factors in a mixture model:

$$score(e|u, q) = \lambda \cdot knowledge(e|u, q) - (1 - \lambda) \cdot time(e|u), \quad (3)$$

where the parameter  $\lambda \in [0; 1]$  controls the balance between the two components and reflects the user's preference in terms of expertise level versus contact time. We consider a range of values for  $\lambda$  in Section 6. Next, we discuss the two main components of our model—knowledge gain and contact time—in detail.

### 5.1 Knowledge Gain

This component expresses the level of expertise that user  $u$  could gain from expert  $e$  on a specific topic  $q$ . We estimate this value by considering the difference between their knowledge on topic  $q$ :

$$knowledge(e|u, q) = p(q|e) - p(q|u), \quad (4)$$

<sup>1</sup> Note that Liebrechts and Bogers [18] use a newer version of the UvT Expert Collection (with more experts and documents) that is not yet publicly available. Therefore, our retrieval scores are not directly comparable with those reported in [18].

<sup>2</sup> <http://code.google.com/p/ears/>

where  $p(q|e)$  and  $p(q|u)$  denote the level of expertise of  $e$  and  $u$ , respectively, on a given topic  $q$ ; these values are estimated using the baseline model (Eq. 2). Note that using this component alone for ranking (i.e., by setting  $\lambda$  to 1 in Eq. 3) would produce a ranking identical to that of the baseline model.

### 5.2 Contact Time

Intuitively, the contact time between a user and an expert could be estimated by their “social distance.” Indeed, shortest-path can be meant as the “most efficient” or “fastest” connection between nodes in a social network [15]. The more intermediaries that separate a user and an expert, the longer it takes for them to contact each other due to the weakening of ties. Based on the data that is available in the UvT collection, we constructed multiple social networks of researchers, induced by particular types of relationships: organizational, geographical, and co-authorship.

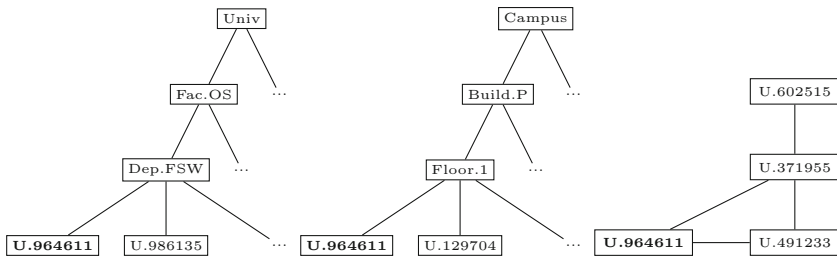
Formally speaking, we assume an undirected connected graph  $G(V, U, E)$ , where nodes  $V$  represent people (UvT employees), nodes  $U$  represent auxiliary nodes, and edges  $E$  reflect relations. The weight of an edge between nodes  $v_i, v_j \in V \cup U$  is denoted by  $w(v_i, v_j)$  and by default is set to 1. In a connected graph there exists a path  $p = \{v_0, \dots, v_k\}$  between any two nodes. The weight of the path is computed as a sum of the weights of its constituent edges:

$$w(p) = \sum_{i=1}^k w(v_{i-1}, v_i). \tag{5}$$

We define the contact time between a user and an expert  $time(e|u)$  to be the length of the shortest path between the corresponding network nodes, normalized by the diameter of the network:

$$time(e|u) = \min_{u \rightarrow e} w(p) / \max_{e, u} \min_{u \rightarrow e} w(p). \tag{6}$$

The proposed approach is very general, and can be applied to any graph that defines relationships between people. In the remainder of this section we present



**Fig. 1.** Illustration of social network structures built from various relationships: (Left) organizational, (Middle) geographical, and (Right) co-authorship

the actual construction of three different types of networks; while these are discussed in the context of a specific organization—Tilburg University—, these types of structures are typically available in organizational settings.

**Organizational network.** Organizational network of the Tilburg University takes the form of hierarchy—see Figure 1 (Left). Therefore, in the graph  $G(V, U, E)$ , nodes  $U$  constitute to organizational units (departments, faculties, university), and edges  $E$  reflect organizational belonging. At the bottom level each individual belongs to one or more departments of the university. Then, departments are hierarchically embedded into corresponding faculties. Finally, faculties are linked to form the university. Employees with missing information about organizational position are considered to be inside the university and each belong to an artificial department and faculty as the only member (thereby, we set the maximum distance value to researchers without organizational information). This network defines a connected graph.

**Geographical network.** The geographical network also forms a hierarchy—see Figure 1 (Middle). In the graph  $G(V, U, E)$ , nodes  $U$  constitute to geographical units (floors, buildings, campus) and edges reflect geographical belonging. Researchers on the same floor are grouped at the lowest level of the hierarchy. Next, floors become a part of the corresponding building. Finally, the campus combines all buildings. Employee profiles with missing information about the individual’s location are considered to be inside the campus and belong each to an artificial building and floor as the only member. Since it is a hierarchy, the geographical network is connected.

**Collaboration network.** In this network there are no auxiliary nodes (the set  $U$  is therefore empty), links exist only between people (i.e., nodes in  $V$ ); two researchers are connected with an edge if they co-authored a paper—see Figure 1 (Right). To ensure that the graph is connected, we add an edge between any pairs of disconnected nodes in  $V$ , and set its weight equal to the diameter of the largest connected component.

The standard properties of these networks are summarized in Table 1. We note that in a realistic setting a user makes use of different types of relationships at the same time in order to reach an expert. Therefore, we also considered combining different types of relations into one network. As a result, nodes in a combined network can be connected by mixed-type paths. Table 1 shows that combined networks indeed exhibit shorter average shortest paths<sup>3</sup>.

<sup>3</sup> We also experimented with alternative ways of calculating the “social distance” between the user and an expert besides the length of the shortest path. Namely, the inverse value of the number of all paths connecting two nodes and the average length of these paths. The former distance favors larger number of paths in the network connecting the user and an expert and captures the premise that an increasing number of alternative paths increase the chances to reach the expert. The latter distance accounts for the fact that the user may use a non-optimal way to contact an expert. Our results showed that the shortest-path distance outperformed the above mentioned alternatives on all network configurations and for all retrieval measures.

**Table 1.** Statistical properties of social networks. Columns from left to right: total number of nodes in the network (Nodes), number of edges (Edges), maximum (Diam.) and average (AvSPath) shortest paths between two nodes in the network.

Network type	Nodes	Edges	Diam.	AvSPath
Org	1263	3210	6	4.76
Geog	1266	3033	6	4.92
Collab	1168	635	16	15.49
Org+Geog	1361	6243	6	4.09
Org+Collab	1263	3838	6	4.62
Collab+Geog	1266	3667	6	4.76
Org+Collab+Geog	1361	6870	6	3.921

## 6 Experimental Evaluation

In this section we present an experimental evaluation of our user-oriented model. This model involves a parameter  $\lambda$  that controls the interplay between the knowledge of an expert and contact time (see Eq. 3). In Table 2 we report retrieval performance for the baseline and for the user-oriented models, considering various types of networks (organizational, geographical, and collaboration), as well as their combinations. The scores we present here are achieved using optimized  $\lambda$  settings; we come back to the setting of this parameter in Section 7.1.

The results obtained show that the user-oriented model markedly outperforms the baseline; all measures are improved, significantly and substantially, independent of the type of the network used. The observed improvements confirm that considering the social distance between people leads to a much more accurate model of selecting experts in an organizational context, and that our approach can effectively incorporate this information into the retrieval model.

We find that the user-oriented model performs better on combined networks. The largest improvement against the baseline is witnessed for the Collab+Geog network; the relative performance increase is over 50% for all metrics.

**Table 2.** Comparison of retrieval performance. The optimal  $\lambda$  values are displayed in brackets. Significance is tested against the baseline using a two-tailed paired t-test;  $\dagger$  and  $\ddagger$  reflect significant changes for  $\alpha = 0.01$  and  $\alpha = 0.001$ , respectively. Best results for each metric are in boldface.

	MAP	MRR	P@5	NDCG@R
Baseline	0.2419	0.5845	0.3067	0.3466
<i>User-oriented model</i>				
Org	0.3727 $\ddagger$ ( $\lambda=0.2$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.4$ )	0.4333 $\ddagger$ ( $\lambda=0.6$ )	0.5321 $\ddagger$ ( $\lambda=0.4$ )
Geog	0.3731 $\ddagger$ ( $\lambda=0.4$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.4$ )	0.4533 $\ddagger$ ( $\lambda=0.6$ )	0.5549 $\ddagger$ ( $\lambda=0.4$ )
Collab	0.3387 $\ddagger$ ( $\lambda=0.5$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.3$ )	0.4000 $\ddagger$ ( $\lambda=0.4$ )	0.4944 $\dagger$ ( $\lambda=0.2$ )
Org+Collab	0.3782 $\ddagger$ ( $\lambda=0.4$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.4$ )	0.4400 $\ddagger$ ( $\lambda=0.6$ )	0.5331 $\ddagger$ ( $\lambda=0.4$ )
Org+Geog	0.3809 $\ddagger$ ( $\lambda=0.2$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.4$ )	0.4400 $\ddagger$ ( $\lambda=0.6$ )	0.5414 $\ddagger$ ( $\lambda=0.4$ )
Collab+Geog	<b>0.4035</b> $\ddagger$ ( $\lambda=0.4$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.4$ )	<b>0.4800</b> $\ddagger$ ( $\lambda=0.4$ )	<b>0.5694</b> $\ddagger$ ( $\lambda=0.4$ )
Org+Collab+Geog	0.3932 $\ddagger$ ( $\lambda=0.3$ )	<b>0.8944</b> $\ddagger$ ( $\lambda=0.4$ )	0.4667 $\ddagger$ ( $\lambda=0.4$ )	0.5446 $\ddagger$ ( $\lambda=0.4$ )

Augmenting this network with information about the organizational hierarchy does not bring in further improvements (Collab+Geog vs. Org+Collab+Geog). We note that the best performing run is significantly different from all other runs in terms of MAP and also (with the exception of Org+Collab+Geog) in terms of NDCG@R. Interestingly, the best performance was almost always (with the exception of P@5) achieved by putting slightly more weight on the contact time aspect over knowledge (i.e.,  $\lambda < 0.5$ ).

## 7 Analysis

So far, we presented retrieval results using optimized parameter settings, averaged over all queries. Below, we analyze the sensitivity of our model w.r.t. its parameter, and take a closer look at differences on the level of individual topics.

### 7.1 Parameter Sensitivity Analysis

The parameter  $\lambda$  plays an important role in the user-oriented model: it regulates the balance between knowledge gain and time to contact an expert. Figure 2 shows retrieval scores for different values of this parameter. We display results for the three single-typed social networks (Org, Collab, Geog) and for the best combination (Collab+Geog); other combinations show very similar behaviour to that of Collab+Geog, but are not reported in the interest of readability.

It is clear from the plots that the combination always performs better than considering either knowledge ( $\lambda = 1$ ) or contact time ( $\lambda = 0$ ) alone. While there are differences in terms of absolute values, all network types exhibit very similar behavior given a metric. We also find that—with the exception of P@5—the optimal combination is shifted towards contact time relatively to the knowledge of an expert, and retrieval performance tops around  $\lambda = 0.4$ . In fact, we observe relatively flat curves when  $\lambda$  is in the range of  $[0.1, 0.4]$ ; this indicates the stability and robustness of the model with respect to the choice of this parameter.

### 7.2 Topic-Level Analysis

We looked at results aggregated over all queries in Section 6, and now we continue our comparison by contrasting the performance of the baseline and user-oriented

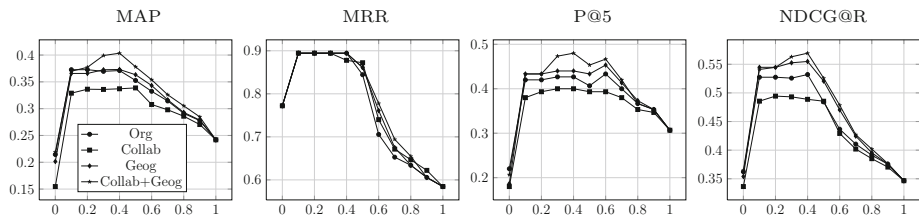
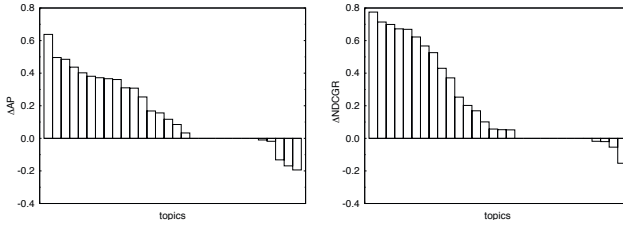


Fig. 2. The effect of varying  $\lambda$  on retrieval scores using different types of social networks



**Fig. 3.** Topic-level differences between the baseline and the best performing user-oriented model in terms of (Left) AP and (Right) NDCG@R

**Table 3.** Topics displaying the largest difference compared to the baseline. Columns: user provided ranking (exp. level) with the corresponding position in the baseline (BL) and user-oriented (UO) rankings; geographical information: building (Build) and room (Room) number; whether an expert collaborated with the user (Collab.).

Expert ID	Build.	Room	Collab.	exp. level	ranked BL	ranked UO
521705	Y	232	No	3	232	1
270229	Y	231	No	3	12	2
171409	Y	252	No	2	25	3
198560	Y	238	No	2	67	16
289604	D	335	No	1	153	20

(a) Positive example #1: user #521705, query “philosophy of science.”

Expert ID	Build.	Room	Collab.	exp. level	ranked BL	ranked UO
523860	P	811	No	4	4	150
961051	K	1008	No	4	2	28
975900	K	1007	No	3	38	203
985430	K	1014	No	3	11	177
316326	K	1006	No	3	23	189
749796	D	405	No	3	4	43
329827	E	116	No	3	14	180

(b) Negative example #1: user #749796, query “marketing communications.”

Expert ID	Build.	Room	Collab.	exp. level	ranked BL	ranked UO
655248	P	1164	No	4	202	20
964611	P	1165	No	4	141	1
800353	P	1162	No	3	999	53
720437	P	3107	No	3	367	99
371955	P	1159	Yes	3	137	2
120146	P	3101	No	1	161	89
491233	P	3104	Yes	1	165	3
578111	P	1161	No	1	826	74

(c) Positive example #2: user #964611, query “customer satisfaction.”

Expert ID	Build.	Room	Collab.	exp. level	ranked BL	ranked UO
890847	M	315	No	4	3	99
265543	K	302	No	3	7	264
700940	K	415	No	3	31	470
968270	M	309	No	3	17	100
938920	P	1180	No	2	1	1
551309	P	1133	No	2	6	54
303267	M	312b	No	2	9	267
917125	K	316	No	2	26	467
859044	M	613	No	2	77	488

(d) Negative example #2: user #938920, query “sociale zekerheid.”

approaches on the level of individual queries. Figure 3 presents differences in average precision (AP) and NDCG@R scores against the baseline. It shows that the user-oriented approach considerably improved performance for more than half of the topics (17 for both AP and NDCG@R), for about a third it has remained the same (8 for AP and 9 for NDCG@R) and declined for the rest (5 for AP and 4 for NDCG@R). With the exception of a handful of topics (3 for AP and 2 for NDCG@R), the rate of decrease, however, is barely noticeable.

Next, we zoom in on four topic examples—two displaying the largest amount of performance increase against baseline, while the other two are taken from the opposite end of the spectrum. From the positive examples (Tables 3a and 3c) we see that almost all ranked experts belong to the same building and floor as the

user (the first digit of the room number indicates the floor). For these topics the user-oriented system correctly places colleagues sharing the same floor higher, as well as the two collaborators in the case of user #964611. In case of the negative examples, the user-oriented system again tried to promote people closest to the user. In one case (Table 3b) the user chose several experts, as most knowledgeable, mostly from a neighbouring building and located on the same floor; here, some personal connections between the user and those experts might play a role, but we do not have access to that type of social information from inside the organization. As to the other negative example (Table 3d), it appears that this particular user’s preference was highly shifted towards knowledge and away from proximity; addressing this issue is a question of setting the knowledge/time balance parameter  $\lambda$  in accordance with the user’s preference.

## 8 Conclusions and Future Work

In this work, we addressed the task of expert finding—ranking people with respect to their expertise on a given topic. We argued that, in a real-world organizational setting, the notion of the “best expert” depends on the individual user performing the search, and we proposed a user-oriented model that incorporates user-dependent factors. This model is based on the assumption that a user’s preference for an expert is balanced between the time needed to contact the expert and the knowledge value gained after. We defined the contact time between the user and an expert as their distance in a social graph, and examined different types of social relations that the user can engage in. Our approach provides a general framework for expert finding that encompasses existing approaches (which focus only on returning the most knowledgeable persons). We performed evaluation against a state-of-the-art baseline on the UvT Expert Collection using graded relevance judgements collected from real users, and demonstrated that the user-oriented approach significantly and substantially outperforms the baseline for all retrieval measures. We completed our investigation with parameter sensitivity examination and a topic-level success and failure analysis.

In future work, we plan to complement our current approach with automatic parameter learning. We also wish to extend this work with user studies that explore additional user-related factors that may play a role in expert finding.

**Acknowledgments.** Balog was supported by the Center for Creation, Content and Technology (CCCT). Smirnova was supported by INRIA, CORDI fellowship and AxIS team via the EC funded Elliot project. Smirnova thanks K. Avrachenkov and B. Trousse for their valuable comments and feedback.

## References

- [1] Ackerman, M., Wulf, V., Pipek, V.: *Sharing Expertise: Beyond Knowledge Management*. MIT Press, Cambridge (2002)
- [2] Anderson, J.: *The adaptive character of thought*. Lawrence Erlbaum Assoc., Mahwah (1990)



- [3] Bailey, P., Craswell, N., de Vries, A.P., Soboroff, I.: Overview of the TREC 2007 enterprise track. In: *The Sixteenth Text REtrieval Conference Proc.* (2008)
- [4] Balog, K.: The SIGIR 2008 workshop on future challenges in expertise retrieval (fCHER). *SIGIR Forum* 42(2), 46–52 (2008)
- [5] Balog, K., Azzopardi, L., de Rijke, M.: Formal models for expert finding in enterprise corpora. In: *SIGIR 2006*, pp. 43–50 (2006)
- [6] Balog, K., Bogers, T., Azzopardi, L., de Rijke, M., van den Bosch, A.: Broad expertise retrieval in sparse data environments. In: *SIGIR 2007*, pp. 551–558 (2007)
- [7] Balog, K., Azzopardi, L., de Rijke, M.: A language modeling framework for expert finding. *Inf. Processing and Management* 45(1), 1–19 (2009)
- [8] Balog, K., Soboroff, I., Thomas, P., Craswell, N., de Vries, A.P., Bailey, P.: Overview of the TREC 2008 enterprise track. In: *TREC 2008* (2009)
- [9] Borgatti, S., Cross, R.: A relational view of information seeking and learning in social networks. *Management Science* 49(4), 432–445 (2003)
- [10] Campbell, C.S., Maglio, P.P., Cozzi, A., Dom, B.: Expertise identification using email communications. In: *CIKM 2003*, pp. 528–531 (2003)
- [11] Craswell, N., de Vries, A.P., Soboroff, I.: Overview of the TREC-2005 Enterprise Track. In: *The Fourteenth Text REtrieval Conference Proceedings* (2006)
- [12] CriES. Cross-lingual Expert Search workshop at CLEF (2010), <http://www.multipa-project.org/cries>
- [13] Dom, B., Eiron, I., Cozzi, A., Zhang, Y.: Graph-based ranking algorithms for e-mail expertise analysis. In: *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 42–48 (2003)
- [14] Fang, H., Zhai, C.: Probabilistic models for expert finding. In: Amati, G., Carpineto, C., Romano, G. (eds.) *ECIR 2007*. LNCS, vol. 4425, pp. 418–430. Springer, Heidelberg (2007)
- [15] Hanneman, R., Riddle, M.: *Introduction to social network methods*. Cambridge University Press, Cambridge (2005)
- [16] Hofmann, K., Balog, K., Bogers, T., de Rijke, M.: Contextual factors for finding similar experts. *Journal of the American Society for Information Science and Technology* 61(5), 994–1014 (2010)
- [17] Karimzadehgan, M., White, R.W., Richardson, M.: Enhancing expert finding using organizational hierarchies. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) *ECIR 2009*. LNCS, vol. 5478, pp. 177–188. Springer, Heidelberg (2009)
- [18] Liebrechts, R., Bogers, T.: Design and evaluation of a university-wide expert search engine. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) *ECIR 2009*. LNCS, vol. 5478, pp. 587–594. Springer, Heidelberg (2009)
- [19] Macdonald, C., Ounis, I.: Voting for candidates: adapting data fusion techniques for an expert search task. In: *CIKM 2006*, pp. 387–396 (2006)
- [20] Macdonald, C., Hannah, D., Ounis, I.: High quality expertise evidence for expert search. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008*. LNCS, vol. 4956, pp. 283–295. Springer, Heidelberg (2008)
- [21] McDonald, D.D., Ackerman, M.: Just talk to me: a field study of expertise location. In: *CSCW 1998*, pp. 315–324 (1998)
- [22] Micarelli, A., Gasparetti, F., Sciarone, F., Gauch, S.: Personalized search on the world wide web. In: *The Adaptive Web: Methods and Strategies of Web Personalization*, pp. 195–230. Springer, Heidelberg (2007)
- [23] Petkova, D., Croft, W.B.: Hierarchical language models for expert finding in enterprise corpora. In: *ICTAI 2006*, pp. 599–608 (2006)

- [24] Petkova, D., Croft, W.B.: Proximity-based document representation for named entity retrieval. In: CIKM 2007, pp. 731–740 (2007)
- [25] Serdyukov, P., Hiemstra, D.: Being omnipresent to be almighty. In: SIGIR 2008 Workshop on Future Challenges in Expertise Retrieval, pp. 17–24 (2008)
- [26] Serdyukov, P., Hiemstra, D., Fokkinga, M.M., Apers, P.M.G.: Generative modeling of persons and documents for expert search. In: SIGIR 2007, pp. 827–828 (2007)
- [27] Shami, S., Ehrlich, K., Millen, D.: Pick me!: link selection in expertise search results. In: CHI 2008, pp. 1089–1092 (2008)
- [28] Soboroff, I., de Vries, A., Crawell, N.: Overview of the TREC 2006 Enterprise Track. In: The Fifteenth Text REtrieval Conference Proceedings (2007)
- [29] Woudstra, L.S.E., Van den Hooff, B.J.: Inside the source selection process: Selection criteria for human information sources. *Information Processing and Management* 44, 1267–1278 (2008)
- [30] Zamir, O., Korn, J., Ficks, A., Lawrence, S.: Personalization of placed content ordering in search results. Google Inc., Assignee (2005)
- [31] Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.* 22(2), 179–214 (2004)
- [32] Zhang, J., Tang, J., Li, J.: Expert finding in a social network. In: Kotagiri, R., Radha Krishna, P., Mohania, M., Nantajeewarawat, E. (eds.) DASFAA 2007. LNCS, vol. 4443, pp. 1066–1069. Springer, Heidelberg (2007)
- [33] Zhu, J., Huang, X., Song, D., Ruger, S.: Integrating multiple document features in language models for expert finding. *Knowl. and Inf. Systems* 23(1), 29–54 (2010)

# Simulating Simple and Fallible Relevance Feedback

Feza Baskaya, Heikki Keskustalo, and Kalervo Järvelin

Department of Information Studies and Interactive Media,  
FIN-33014 University of Tampere, Finland

{Feza.Baskaya,Heikki.Keskustalo,Kalervo.Jarvelin}@uta.fi

**Abstract.** Much of the research in relevance feedback (RF) has been performed under laboratory conditions using test collections and either test persons or simple simulation. These studies have given mixed results. The design of the present study is unique. First, the initial queries are realistically short queries generated by real end-users. Second, we perform a user simulation with several RF scenarios. Third, we simulate human fallibility in providing RF, i.e., incorrectness in feedback. Fourth, we employ graded relevance assessments in the evaluation of the retrieval results. The research question is: how does RF affect IR performance when initial queries are short and feedback is fallible? Our findings indicate that very fallible feedback is no different from pseudo-relevance feedback (PRF) and not effective on short initial queries. However, RF with empirically observed fallibility is as effective as correct RF and able to improve the performance of short initial queries.

**Keywords:** Relevance feedback, fallibility, simulation.

## 1 Introduction

Query modification (QM) means query reformulation by changing its search keys (or modifying their weights) in order to make it better match relevant documents. Query formulation, reformulation, and expansion have been studied extensively because the selection of good search keys is difficult but crucial for good results. Real searchers' first query formulation often acts as an entry to the search system and is followed by browsing and query reformulations [9]. Relevance feedback (RF) based on initial query results and query expansion (QE) have been the main approaches to QM. Efthimiadis [2], Ruthven and Lalmas [11], Ruthven, Lalmas and van Rijsbergen [12] provide useful reviews of the techniques.

In the present paper we focus on interactive RF. In this method, users either point out relevant documents and the retrieval system infers the expansion keys for the feedback query, or the retrieval system presents a list of candidate expansion keys for the user to choose from. Knowledgeable experienced searchers may benefit more of RF because they recognize relevant vocabulary and are better able to articulate their needs initially [13]. Users also seem more likely to identify highly relevant documents than marginal ones [18].

There are two difficulties in providing feedback: searcher's capability and willingness [11]. Pseudo-relevance feedback (PRF) [11] avoids these challenges by assuming

that the first documents of an initial search result are relevant. Long documents and non-relevant documents however introduce noise in the PRF process thus causing query drift. To counteract this, one may use query-biased summaries [8], [16] for the identification of expansion keys. Lam-Adesina & Jones [8] and Järvelin [5] have shown that query-biased summaries positively affect PRF effectiveness. Yet another challenge to PRF is that real users tend to issue very short queries [4] and employ shallow browsing. As a consequence, the initial query results tend to be of poor quality and sparse regarding relevant documents, thus making PRF ineffective regarding the computational effort. Query-biased summaries may nevertheless counteract the latter to some degree [8].

Järvelin [5] argued that while RF is more effective than PRF, the performance difference does not justify the necessary searcher's effort. His results were however based on long queries (Title+Description). In the present paper we examine the effectiveness of RF and PRF under short initial queries. This is motivated by observed searcher behavior [4]. This leaves a chance for RF score higher than PRF since the initial performance may not be good enough for PRF to be effective.

However, searcher's capability to identify relevant documents may be limited. Humans are fallible. Turpin and colleagues [17] showed that snippets (i.e. query-biased summaries) are important in IR interaction and bad snippets may lead to incorrect relevance decisions. Vakkari and Sormunen [18] showed that humans may well err on marginal and non-relevant documents while are likely to identify the highly relevant ones correctly. Foley and Smeaton [3] examine collaborative IR where the collaborators may err. These findings suggest that the effect of correctness of RF should be examined. Since searcher performance may vary greatly across situations, we investigate in the present paper a range of fallibility scenarios.

Some earlier studies [3] and [5] suggest that RF is most effective when little feedback is given as early as possible – that is, the searcher should identify one or two first relevant documents in the initial result and stop browsing there. One should not be picky regarding the quality of the feedback documents, i.e. marginal ones would do. Therefore in the present study, our main RF scenario is based on shallow browsing (max top-10) and identifying the first two relevant documents of whatever relevance degree (perhaps erroneously) as feedback.

We base our experiments on searcher simulation (like [3] and [7]) rather than tests with real users. Simulation has several advantages, including cost-effectiveness and rapid testing without learning effects as argued in the SIGIR SimInt 2010 Workshop [1]. Besides, the simulation approach does not require a user interface. The informativeness and realism of searcher simulation can be enhanced by explicitly modeling, in the present case, those aspects of searchers and RF that pertain to RF effectiveness. In the present paper, two issues are significant: (a) realistic short queries, and (b) realistic fallibility of searchers' relevance judgments. While we perform our study in a test collection, we employed test persons to generate short queries (length 1 – 3 words). These are more realistic and controllable than, e.g. the title elements of TREC topics. To study the effects of fallibility, we employ several fallibility scenarios ranging from random judgments to perfect judgments with one scenario based on the empirical findings by Vakkari & Sormunen [18]. We implement them as probability distributions over possible degrees of relevance. In this way, we may employ both analytical variety and empirical grounding in our simulations.

Our evaluations are based on three metrics (MAP, P@10 and P@20) and three levels of relevance. Regarding the metrics, the main role is given to P@10 and P@20 as clearly user-oriented measures – users frequently avoid browsing beyond the first results page, i.e. 10 links/documents [4]. After giving RF and already browsing up to 10 documents, the P@20 can be seen as evaluation for quasi first page. For comparison, MAP is reported as well. The three levels of evaluation are liberal (i.e. even marginal documents are taken as relevant), fair (medium and highly relevant documents are relevant), and, strict (only highly relevant documents matter). This is justified because the user may not benefit from many marginal documents at all, and because there are systematic performance differences across the evaluation levels.

We utilize the TREC 7-8 corpus with 41 topics for which graded relevance assessments are available [14]. The search engine is Lemur. The fallibility simulations are based on the relevance degrees of documents given in the recall base of the test collection (the qrels files) and probability distributions across the possible (partially erroneous) simulated user judgments. A random number generator is used to drive the judgments. All experiments are run 50 times with random decisions and the reported results are averages over the 50 runs. We will use PRF results as baselines to our simulated RF experiments.

## 2 Study Design

### 2.1 Research Questions

Our overall research question is: how does RF affect IR performance when short initial queries are employed and fallible feedback is provided? More specifically:

- RQ 1: How effective are PRF and RF when employed on the results of short initial queries and shallow browsing?
- RQ 2: Does RF effectiveness seriously deteriorate when RF is of progressively lower quality?
- RQ 3: How does RF effectiveness in RQ2 depend on evaluation by liberal, fair vs. strict relevance criteria?

### 2.2 The Test Collection, Search Engine, and Query Expansion Method

We used the reassessed TREC 7-8 test collection including 41 topics [14]. The document database contains 528155 documents indexed under the retrieval system *Lemur Indri*. The index was constructed by stemming document words. The relevance assessments were done on a four-point scale: (0) irrelevant, (1) marginally relevant, (2) fairly relevant, and (3) highly relevant document. In the recall base there are on average 29 marginally relevant, 20 fairly relevant and 10 highly relevant documents for each topic. For three topics there were no highly relevant documents. This recall base with its intrinsic human judgment errors is taken as a gold standard for further fallibility study and evaluation.

The research questions do not require any particular interactive query expansion method to be employed. We simulate interactive RF that takes place at document level: the simulated users point to relevant documents and the RF system then automatically extracts the expansion keys. We follow Tombros and Sanderson [16], Lam-Adesina & Jones [8] and Järvelin [5] who have shown that query-biased summaries positively affect RF effectiveness. Given a query and an indicated relevant document, our QE method ranks the document sentences by their query similarity, then extracts the top- $n$  ( $n=5$ ) sentences, and then collects the non-query words from these sentences, scores them by their ( $tf \cdot idf$  based) discrimination power, and chooses the top- $k$  ( $k=30$ ) most significant words as expansion keys to be appended to the RF query. When multiple documents are indicated for feedback, top- $n$  sentences are collected from each and then pooled before sentence scoring and key extraction. The parameter values for  $n$  and  $k$  were found reasonable in prior studies [5]. When scoring sentences, if a non-stop query word did not match any sentence word, an  $n$ -gram type of approximate string matching with a threshold was attempted [10].

Initial short queries, 1-3 words in length, were constructed based on real searchers' suggestions (see below) but the query keys were stemmed. Multi-word queries were constructed as bag-of-word queries. Feedback queries were constructed by appending the feedback keys to the initial query as a second bag-of-words.

### 2.3 User Modeling for RF Simulation

The design of RF simulation requires several decisions to be made: (1) user's willingness to browse the initial result, (2) user's willingness to provide RF, (3) the level of relevance of the RF documents, and (4) user's fallibility in making relevance judgments. The first three decisions are suggested in Keskustalo and colleagues [7] as a user model. Their general recommendation was also that RF is most effective when the browsing depth is shallow (we use 10 documents here), when only little RF is given as early as possible (we provide the first two relevant document as RF, and then stop to browse), and that even marginal documents as RF as early as possible are better than highly relevant documents given late (we provide the first two relevant document as RF whatever their degree of relevance). Järvelin [5] confirmed these findings. In these simulation studies, the recall base of the test collection was used as the source of relevance judgments for RF. This means that the initial query result was scanned and each document ID on the ranked list was checked against the recall base of the topic in question.

The fourth decision, on human fallibility, is a novelty in RF simulation. This is motivated by Turpin and colleagues [17] and Vakkari and Sormunen [18], who point out errors in human relevance judgments. In the present study, the recall base is still a source in relevance judgment, but not taken as a fact as such. We simulate users that with some probability make correct judgments, and with some other probabilities err more or less. We have thus a probability distribution around the correct judgment. For example, such a distribution could state for a document of relevance degree, say 'fair', that there is a 10% probability for the user to assess the document as non-relevant, 20% probability as marginal, 50% as fair (correct), and 20% as highly relevant. Table 1 summarizes the fallibility scenarios employed in the present study.

**Table 1.** Fallibility probability distributions

Fallibility Scenario	Human Judgment Probabilities				
	n	m	f	h	
<b>1.00</b>	n	<b>1.0</b>	0.0	0.0	0.0
	m	0.0	<b>1.0</b>	0.0	0.0
	f	0.0	0.0	<b>1.0</b>	0.0
	h	0.0	0.0	0.0	<b>1.0</b>
<b>0.75</b>	n	<b>0.75</b>	0.125	0.075	0.05
	m	0.10	<b>0.75</b>	0.10	0.05
	f	0.05	0.10	<b>0.75</b>	0.10
	h	0.05	0.075	0.125	<b>0.75</b>
<b>0.50</b>	n	<b>0.50</b>	0.25	0.15	0.10
	m	0.20	<b>0.50</b>	0.20	0.10
	f	0.10	0.20	<b>0.50</b>	0.20
	h	0.10	0.15	0.25	<b>0.50</b>
<b>0.25</b>	n	<b>0.25</b>	0.25	0.25	0.25
	m	0.25	<b>0.25</b>	0.25	0.25
	f	0.25	0.25	<b>0.25</b>	0.25
	h	0.25	0.25	0.25	<b>0.25</b>
<b>0.50-0.80</b>	n	<b>0.5</b>	0.4	0.1	0.0
	m	0.4	<b>0.5</b>	0.1	0.0
	f	0.0	0.1	<b>0.8</b>	0.1
	h	0.0	0.0	0.2	<b>0.8</b>

In Table 1, the row sets represent fallibility scenarios. The first set, labeled 1.00, represents the gold standard for RF, always correct judgments of the feedback documents. The rows within 1.00 represent ground truth relevance of non-relevant (n), marginal (m), fair (f), and highly relevant (h) documents. The human judgment probabilities in columns represent the simulated human judgments. In the gold standard all judgments are correct, indicated by probability 1.0 in the diagonal.

The next three sets are labeled as 0.75, 0.50, and 0.25, indicating progressively more random judgments among the retrieved ranked documents, from

fairly consistent to fully random. The final set, labeled as 0.50-0.80, is based on Vakari and Sormunen's [18] empirical findings. They reported that searchers are able to recognize highly relevant documents quite consistently but tend to err on marginal and non-relevant ones. Also Sormunen [14] found the judges inconsistent: most inconsistency occurred between neighboring relevance classes. Therefore the scenarios in Table 1 allocate intuitively more of the probability mass to neighboring classes than to more distant ones.

In our simulations, we use a random number generator together with the judgment scenarios to drive simulated relevance judgments. Because RF effectiveness is bound to be sensitive to random judgments, we run each RF experiment 50 times over and report the average effectiveness.

## 2.4 Short Initial Queries

Test collections such as the TREC collections provide their test topics structured as titles (T), descriptions (D), and narratives (N). In our TREC7-8 test collection, the titles of the 41 topics vary in length from 1 to 3 words, with 2.4 words average. The descriptions have an average length of 14.5 words. Real-life searchers often prefer very short queries [4] [15]. Jansen and colleagues [4] analyzed transaction logs containing thousands of queries posed by Internet search service users. They discovered that one in three queries had only *one* keyword. The average query length was 2.21 keys. Less than 4 % of the queries in Jansen's study had more than 6 keywords. The

average number of keywords per query was even less, 1.45, in Stenmark's study [15], focusing on intranet users. Therefore it makes sense to test the effectiveness of initial queries of length of 1 to 3 words in RF scenarios. A further point is that test collection topic titles are carefully crafted to summarize each topic whereas end users are rather characterized by trial-and-error carelessness. Therefore we wanted to have end-user created short queries for our experiments.

The 41 topics were analyzed intellectually by test persons to form query candidate sets. A group of seven undergraduate information science students performed the analysis. Regarding each topic a printed topic description and a task questionnaire were presented for the test persons. Each of the 41 topics was analyzed by a student. The subjects were asked to directly select and think up good search keys from topical descriptions and to create various query candidates.

First a two-page protocol explaining the task was presented by one of the researchers. Information in the description and narrative fields of the test collection topics was presented to the users. Descriptions regarding non-relevance of documents were omitted to make the task more manageable within the time limitation of 5 minutes per topic. The test persons were asked to mark up all potential search words directly from the topic description and to express the topic freely by their own words. Third, they were asked to form various query candidates (using freely any kinds of words) as unstructured word lists: (i) the query they would use first ("1<sup>st</sup> query"); (ii) the one they would try next, assuming that the first attempt would not have given a satisfactory result ("2<sup>nd</sup> query"). Finally, the test persons were asked to form query versions of various lengths: (iii) one word (1w), (iv) two words (2w), and (v) three or more words (3w+). The very last task was to estimate how appropriate each query candidate was using a four-point scale. During the analysis the test persons did not interact with a real IR system.

In the present experiment, we used the short queries, ranging from 1 to 3 words, from this data set as the initial queries. The results of these were subject to RF under various feedback and fallibility scenarios.

## 2.5 Experimental Protocol

Figure 1 illustrates the overall experimental protocol. TREC topics are first turned to initial short queries (stemmed) of given length and executed with Lemur, followed by feedback document selection. This is based on the simulated searcher's feedback scenario (in the present experiments browsing up to 10 documents and returning the first two documents fallibly judged relevant as RF). The random judgments were repeated 50 times. In each case, the feedback documents for each query are split into sentences, and the sentences are scored on the basis of the query word scores. Word to word matches are facilitated by stemming and, in the case of Out-of-Vocabulary words (OOVs), by n-gram string matching. The sentences are ranked and the  $k$  best ones are extracted for each document. After processing the feedback documents, the  $m$  ( $m=5$ ) overall best sentences are identified for expansion key extraction. For each query's set of feedback sentences, their non-query, non-stop words are ranked by their scores and the 30 overall best keys are identified as expansion keys for the query and



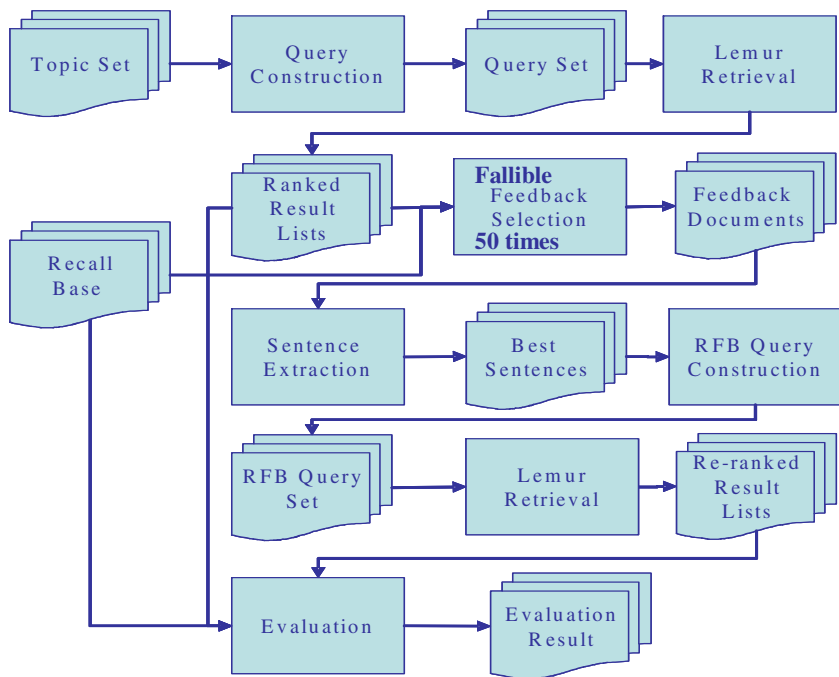


Fig. 1. Query-biased summarization process

added to the initial query. The new query is executed and both the original and feedback query results go to evaluation.

## 2.6 Evaluation and Statistics

In evaluation we employ full freezing (e.g. [7]) of the all documents 'seen', this is, (1) freezing all initially scanned (say,  $f$ ) documents for RF, relevant or not, at their ranks, (2) removing all initially seen documents from the RF query result, and (3) then filling the positions from  $f+1$  with the feedback query results. We use standard evaluation metrics available in the TREC-eval package and report evaluation results for  $P@10/20$  documents, and mean average precision MAP. The former are motivated by real life findings – people most often are precision-oriented and avoid excessive browsing – great results beyond the first pages don't matter. We employ liberal RF but three final evaluation levels, where liberal accepts all at least marginal documents as relevant, fair accepts all at least fairly relevant as relevant, and strict only highly relevant as relevant. Statistical testing is based on Friedman's test between RF runs and the baseline. PRF on the initial query result provides the stronger baseline, and therefore PRF is used as the baseline when statistical significance is evaluated. We ran several PRF experiment with 1, 2, 5 and 10 PRF documents. We report results for 2 PRF documents because using more did not consistently improve effectiveness.

### 3 Findings

#### 3.1 Initial and PRF (Baseline) Queries

Table 2 reports the initial query performances for user-defined one, two and three-word queries, as well as PRF queries at the three evaluation scenarios (liberal, fair and strict). The best query performance values are indicated by dark gray background. We see, among others, that the initial one-word queries are 3.4 (at fair evaluation) to 4.2 (at liberal) % units (MAP) weaker than 2-word queries except at strict level. Initial query MAP values for 3-word queries are 1.9 (at fair) to 4.0 (at liberal) % units better than one-word initial query values, and 1.3 (at fair) to 2.3 (at liberal) % units better than two-word query results. At strict evaluation results are slightly worse than one-word query results. On the other hand, P@10 initial values for two-word queries improve continuously the initial one-word query results from 9.3 % units (at liberal) to 1.1 % units (at strict). Compared to one-word queries, P@10 initial values for three-word queries improve also the initial results from 10.8 % units (at liberal) to 1.8 % units (at strict). P@20 initial query values for two-word queries improve continuously the initial query results from 7.8 % units (at liberal) to 1.4 % units (at strict). P@20 initial values for three-word queries improve also the initial results for one-word queries.

The PRF for one-word queries improves both MAP and P@10 only around 1 % and 0.5 % units respectively at liberal evaluation. At strict evaluation it decreases the MAP reading 1.7 %. The greatest PRF improvement in P@10 for one-word queries is 0.5 % units (at liberal). We can confirm earlier findings that tighter evaluation weakens PRF effectiveness [6]. The greatest PRF improvements in MAP for two-word queries are from 1.8 % units (at liberal) to 0.5 % units (at fair). The greatest PRF improvements in P@10 for two-word queries are 2.4 % units (at strict) to 1.0 % units (at fair) and in P@20 for two-word queries are 2.2 % units (at liberal) to 0.2 % units (at fair). When initial query length grows, the initial query effectiveness grows greatly, e.g. with liberal evaluation, P@10 grows by 10.7 % units and P@20 grows by 8.3% units. Likewise, the PRF to initial query effectiveness for P@10 improves by 3.9 % – 2.6% units depending on query length and evaluation stringency. Further, the shorter the initial queries are, the less PRF contributes. Thus PRF seems not capable of improving poor initial results. These findings hold for all evaluation metrics.

The findings above are deliberately for short initial queries reflecting real life searcher behavior. PRF on top of the RF query results (with no fallibility) did not yield any improvement.

#### 3.2 Expanded Runs and Fallibility in the Process

Table 2 also reports RF query effectiveness for all metrics (MAP, P@10 and P@20) under several user fallibility and evaluation scenarios. Refer to Table 1 for the explanation of the fallibility scenarios. Friedman’s test indicates overall significant statistical differences in each block of experiments defined by initial query length, metric and evaluation scenario ( $p < 0.05$ ). This allows examining the pair wise significant differences among the results in each block. Table 2 indicates (by ‘\*’) those pair wise differences between the PRF as baseline and fallible RF that are significant at the risk

**Table 2.** Simulated RF effectiveness for short queries

Queries		Liberal			Fair			Strict		
Fallibility		MAP	P@10	P@20	MAP	P@10	P@20	MAP	P@10	P@20
1-Word	Initial	0.143	0.246	0.209	0.164	0.210	0.171	0.190	0.111	0.080
	PRF	0.151	0.251	0.212	0.164	0.210	0.171	0.173	0.111	0.079
	1.00	0.161	0.261	0.243*	0.172*	0.215	0.192*	0.195*	0.108	0.090
	0.75	0.159	0.258	0.235	0.170	0.213	0.186	0.194	0.107	0.087
	0.50	0.158	0.257	0.232	0.169	0.213	0.182	0.193	0.108	0.085
	0.25	0.154	0.253	0.223	0.166	0.210	0.175	0.191	0.107	0.081
	0.5-0.8	0.161	0.261	0.242*	0.172*	0.215	0.191*	0.195*	0.108	0.089
2-Word	Initial	0.185	0.339	0.287	0.198	0.278	0.224	0.178	0.121	0.095
	PRF	0.203	0.356	0.309	0.203	0.288	0.227	0.192	0.145	0.097
	1.00	0.215	0.376	0.334*	0.218	0.302	0.243	0.197*	0.145	0.109
	0.75	0.213	0.376	0.330	0.216	0.305	0.241	0.195	0.145	0.108
	0.50	0.210	0.373	0.324	0.213	0.302	0.236	0.192	0.143	0.106
	0.25	0.206	0.367	0.315	0.209	0.298	0.231	0.189	0.141	0.102
	0.5-0.8	0.215*	0.378	0.336*	0.218*	0.306	0.244	0.196*	0.145	0.110*
3-Word	Initial	0.183	0.354	0.292	0.182	0.266	0.209	0.187	0.129	0.095
	PRF	0.209	0.393	0.326	0.199	0.305	0.235	0.195	0.155	0.107
	1.00	0.219	0.400	0.339	0.204	0.295	0.237	0.205	0.153	0.108
	0.75	0.217	0.394	0.339	0.203	0.291	0.237	0.203	0.151	0.109
	0.50	0.215	0.389	0.338	0.200	0.287	0.237	0.201	0.149	0.107
	0.25	0.208*	0.380	0.328	0.194*	0.281*	0.230	0.196	0.145	0.103
	0.5-0.8	0.220	0.398	0.340	0.205	0.294	0.238	0.205	0.151	0.109

**Legend:** \* indicates statistically significant difference to PRF baseline, Friedman’s test,  $p < 0.05$ .

level  $p < 0.05$ . In Table 2, background shading indicates the best performance in each column – lighter shading the strongest initial query and darker shading the strongest (P)RF query. PRF is also highlighted with a gray background.

Correct RF nearly always yields better effectiveness than PRF, but the difference is not always statistically significant. In MAP the difference is 0.6 to 2.2 % units, in P@10, -1.0 to 2.0 % units, and in P@20, 0.1 to 3.1 % units depending on initial query length and evaluation scenario. In MAP, there is a tendency for the difference to grow by tighter evaluation. In P@10 and P@20, the difference of correct feedback to PRF diminishes by tightening the evaluation. While both PRF and correct RF generally benefit from growing query length, PRF seems to benefit more.

The distribution of the fallibility results for MAP, P@10 and P@20 follows the judgment capability of the user. As the probability of incorrect judgments increases, the results are decreasing. A clear trend between 100 % correct RF and random RF (fallibility 0.25) is that the latter delivers worse results. Random RF rarely yields results significantly different from PRF, which was expected. While both generally

yield some improvement over the initial query baseline, the difference is not significant and tends to shrink by tighter evaluation criteria, being sometimes negative by strict criteria. Further, better relevance judgment capability clearly improves the results. In case of fallibility 0.75 the results are slightly better than with fallibility 0.5. The empirically grounded fallibility in RF is never significantly different in effectiveness from correct RF. The difference is  $\pm 0.4$  % units. This means that RF with empirically observed fallibility is as good as correct RF.

In summary, when initial queries are realistically short, the initial query results are relatively weak. This renders blind techniques, PRF and random RF ineffective. There is room for effective human interaction even when the initial queries are short. Despite their fallibility, humans can identify the relevant bits in poor results reliably enough for the benefit of their searching. However, RF requires human effort while PRF is automatic. The practical effectiveness difference is not material.

## 4 Discussion and Conclusion

Simulation entails using a symbolic model of a real-world system in order to study the real-world problems. The model is a simplified representation of the real world. The relevant features of the real world should be represented while other aspects may be abstracted out. This motivates our present study in which we model user interaction features during RF and vary them systematically. The validity of our simulation model is justified by observations in IR literature regarding query lengths, RF behavior and relevance judgment fallibility.

We started our simulation experiment by discussing relevant features of the real world searching. In the most general level one can observe that interaction is vital in real life IR. Secondly, individual users vary greatly. However, typical real life user interaction can be characterized as being simple and error-prone, more specifically: (1) searchers prefer using short (or even very short) queries; (2) searchers prefer shallow browsing (e.g., at most the top-10 documents observed, not top-1000); (3) searchers may be reluctant to give RF, (4) even if they are eager to give RF, they may make errors.

In the present paper we performed a simulation based on modeling real life features listed above, in other words, (1) very short initial queries are used (one, two, and three-word queries); (2) shallow browsing is assumed (at most top-20 documents per query); (3) PRF is also modeled, because it avoids requiring direct RF from the user; (4) fallibility is modeled based on several scenarios assuming that the simulated user makes errors during the selection of feedback documents. These scenarios range from assuming perfect user judgments (no errors) to random judgments (lots of errors). Importantly, we also construct a scenario based on empirical findings on the level of fallibility when the user attempts to recognize relevant documents belonging to various relevance levels [18]. In all, five different fallibility scenarios were studied. All experiments were run 50 times with random decisions and the reported results were averaged over the 50 runs.

Evaluation of the experiments was based on user-oriented measures, P@10 / P@20, and the traditional system-oriented measure, MAP. We used three distinct relevance levels because in real life different kinds of users exist. Some users prefer

finding mixed-level documents, while others want to focus on the best (highly-relevant) documents. We used full freezing during evaluation because it closely imitates the point of view of a real user who has wasted effort in inspecting any number of documents, regardless of their relevance level.

Regarding the first research question, our results suggest that using query-biased summaries is a promising method to approach both PRF and direct user-RF when initial very short queries are assumed. For the second research question we observed that although increasing fallibility decreases the performance compared to perfect RF, it is slightly better than the best performing PRF. Surprisingly, RF with a realistic level of fallibility yields results that are close to perfect RF. Third, when realistic fallibility is assumed and a user-oriented evaluation measure ( $P@10/P@20$ ) is used, at the liberal relevance level RF systematically improves the performance of all short-query types (one word, two word, and three word queries). However, when strict evaluation is demanded, RF does not improve the performance of all short queries against PRF (Table 2). This suggests that the results of very short initial queries do not provide often enough sufficiently good RF documents even for human eyes. This may in part explain the low pick-up rate of RF in real life. Searchers rather issue a new query.

In the future we aim at developing simulation of user interaction in IR toward more fine-grained models of user interaction.

## Acknowledgement

This research was funded by Academy of Finland grant number 133021.

## References

1. Azzopardi, L., Järvelin, K., Kamps, J., Smucker, M.: Report on the SIGIR 2010 Workshop on the Simulation of Interaction. *SIGIR Forum* 44(2), 35–47 (2010)
2. Efthimiadis, E.N.: Query expansion. In: Williams, M.E. (ed.) *Annual Review of Information Science and Technology ARIST*, vol. 31, pp. 121–187. Information Today, Inc., Medford (1996)
3. Foley, C., Smeaton, A.F.: Synchronous Collaborative Information Retrieval: Techniques and Evaluation. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) *ECIR 2009*. LNCS, vol. 5478, pp. 42–53. Springer, Heidelberg (2009)
4. Jansen, M.B.J., Spink, A., Saracevic, T.: Real Life, Real Users, and Real Needs: A Study and Analysis of User Queries on the Web. *Information Processing & Management* 36(2), 207–227 (2000)
5. Järvelin, K.: Interactive Relevance Feedback with Graded Relevance and Sentence Extraction: Simulated User Experiments. In: Cheung, D., et al. (eds.) *Proceedings of the 18th ACM Conference on Information and Knowledge Management (ACM CIKM 2009)*, Hong Kong, November 2–6, pp. 2053–2056 (2009)
6. Keskustalo, H., Järvelin, K., Pirkola, A.: The Effects of Relevance Feedback Quality and Quantity in Interactive Relevance Feedback: A Simulation Based on User Modeling. In: Lalmas, M., MacFarlane, A., Rüger, S.M., Tombros, A., Tsirikla, T., Yavilinsky, A. (eds.) *ECIR 2006*. LNCS, vol. 3936, pp. 191–204. Springer, Heidelberg (2006)

7. Keskustalo, H., Järvelin, K., Pirkola, A.: Evaluating the Effectiveness of Relevance Feedback Based on a User Simulation Model: Effects of a User Scenario on Cumulated Gain Value. *Information Retrieval* 11(5), 209–228 (2008)
8. Lam-Adesina, A.M., Jones, G.J.F.: Applying Summarization Techniques for Term Selection in Relevance Feedback. In: *Proc. of the 24th Annual ACM Conference on Research and Development in Information Retrieval*, pp. 1–9. ACM Press, New York (2001)
9. Marchionini, G., Dwiggins, S., Katz, A., Lin, X.: Information seeking in full-text end-user-oriented search systems: The roles of domain and search expertise. *Library and Information Science Research* 15(1), 35–70 (1993)
10. Pirkola, A., Keskustalo, H., Leppänen, E., Känsälä, A.-P., Järvelin, K.: Targeted S-Gram Matching: A Novel N-Gram Matching Technique for Cross- and Monolingual Word Form Variants. *Information Research* 7(2) (2002), <http://InformationR.net/ir/7-2/paper126.html>
11. Ruthven, I., Lalmas, M.: A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review* 18(2), 95–145 (2003)
12. Ruthven, I., Lalmas, M., van Rijsbergen, K.: Incorporating user search behaviour into relevance feedback. *Journal of the American Society for Information Science and Technology* 54(6), 529–549 (2003)
13. Sihvonen, A., Vakkari, P.: Subject knowledge improves interactive query expansion assisted by a thesaurus. *J. Doc.* 60(6), 673–690 (2004)
14. Sormunen, E.: Liberal Relevance Criteria of TREC - Counting on Negligible Documents? In: *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 320–330. ACM Press, New York (2002)
15. Stenmark, D.: Identifying Clusters of User Behavior in Intranet Search Engine Log Files. *Journal of the American Society for Information Science and Technology* 59(14), 2232–2243 (2008)
16. Tombros, A., Sanderson, M.: Advantages of query biased summaries in information retrieval. In: *Proc. of the 21st Annual ACM Conference on Research and Development in Information Retrieval*, pp. 2–10. ACM Press, New York (1998)
17. Turpin, A., et al.: Including Summaries in System Evaluation. In: *Proc. of the 32nd Annual ACM Conference on Research and Development in Information Retrieval*, pp. 508–515. ACM Press, New York (2009)
18. Vakkari, P., Sormunen, E.: The influence of relevance levels on the effectiveness of interactive IR. *J. Am. Soc. Inf. Sci. Tech.* 55(11), 963–969 (2004)

# AutoEval: An Evaluation Methodology for Evaluating Query Suggestions Using Query Logs

M-Dyaa Albakour<sup>1</sup>, Udo Kruschwitz<sup>1</sup>, Nikolaos Nanas<sup>2</sup>, Yunhyong Kim<sup>3</sup>,  
Dawei Song<sup>3</sup>, Maria Fasli<sup>1</sup>, and Anne De Roeck<sup>4</sup>

<sup>1</sup> University of Essex, Colchester, UK  
`malbak@essex.ac.uk`

<sup>2</sup> Centre for Research and Technology - Thessaly, Greece

<sup>3</sup> Robert Gordon University, Aberdeen, UK

<sup>4</sup> Open University, Milton Keynes, UK

**Abstract.** User evaluations of search engines are expensive and not easy to replicate. The problem is even more pronounced when assessing adaptive search systems, for example system-generated query modification suggestions that can be derived from past user interactions with a search engine. Automatically predicting the performance of different modification suggestion models *before* getting the users involved is therefore highly desirable. *AutoEval* is an evaluation methodology that assesses the quality of query modifications generated by a model using the query logs of past user interactions with the system. We present experimental results of applying this methodology to different adaptive algorithms which suggest that the predicted quality of different algorithms is in line with user assessments. This makes *AutoEval* a suitable evaluation framework for adaptive interactive search engines.

## 1 Introduction

Interactive search interfaces are becoming more popular in modern search engines. Google wonder wheel<sup>[1]</sup> and AquaBrowser<sup>[2]</sup> are examples of such interfaces which provide visualised query refinement suggestions to guide users in search and navigation in addition to providing a list of documents.

In order to provide suggestions for query modification, a domain model that reflects the domain characteristics could be used, e.g. a taxonomy or simply some term association graph. Several methods have been proposed in the literature to build such models. Some of these methods perform statistical and lexical analysis on the document contents to derive term relations, e.g. [10]. With the increasing availability of search logs obtained from user interactions with search engines, new methods have been developed for mining search logs to capture “collective intelligence” for providing query suggestions. This can be done, for example, by looking at the actual queries submitted and building query flow graphs [1], query-click graphs [2] or association rules [4].

<sup>1</sup> <http://www.googlewonderwheel.com>

<sup>2</sup> <http://serialssolutions.com/aquabrowser/>

The evaluation of these domain models in providing query recommendations remains a major challenge. The standard evaluation mechanism is to conduct user studies, e.g. [10], but such studies are expensive, not easy to reproduce and they involve a great deal of subjectivity. The automatic evaluation of search systems that does not rely on expensive user judgements has long been attracting IR researchers, e.g. [11]. This is however not an easy exercise and unlike commonly understood TREC measures (such as precision and recall), there is no commonly agreed automatic evaluation measure for *adaptive* search. One approach for automatic evaluation is using search logs. Joachims shows how clickthrough data can replace relevance judgements by experts or explicit user feedback to evaluate the quality of retrieval functions [5]. Zhang *et al.* have recently shown how test collections specific to a library domain can be derived from search logs [12].

In this paper we explore experimentally a new evaluation approach based on search logs. Search logs contain information of what users entered and clicked. It is a reflection of a reality and is representative to both its document collection and its search transactions. *AutoEval* is a methodology that performs simulated query recommendation experiments based on past log data to evaluate different models for generating query suggestions.

The rest of the paper is structured as follows. In Section 2 we give an overview of the *AutoEval* methodology. In Section 3 we describe the experiments we have run. Results are discussed Section 4.

## 2 The AutoEval Methodology

*AutoEval* is based on the idea that we can assess the quality of a domain model by comparing suggestions derived from the model to query modifications actually observed in the log files. The idea has been proposed recently [8], but no experimental justification has been provided as yet. With *AutoEval*, the model's evaluation is performed on arbitrary intervals, e.g. on a daily basis. For example, let us assume that during the current day, three query modifications have been submitted. For each query modification pair, the domain model is provided with the initial query and returns a ranked list of recommended query modifications. We take the rank of the actual modified query (i.e., the one in the log data) in this list, as an indication of the domain model's accuracy. The assumption here is that an accurate domain model should be able to propose the most appropriate query modification at the top of the list of recommended modifications. This is based on the observation that users are much more likely to click on the top results of a ranked list than to select something further down [6], and it seems reasonable to assume that such a preference is valid not just for ranked lists of search results but for lists of query modification suggestions as well. So for the total of three query modifications in the current day, we can calculate the model's Mean Reciprocal Rank (*MRR*) score as  $(1/r_1 + 1/r_2 + 1/r_3)/3$ , where  $r_1$  to  $r_3$  are the ranks of the actual query modifications in the list of modifications recommended by the model in each of the three cases. More generally, given a day  $d$  with  $Q$  query modification pairs, the model's Mean Reciprocal Rank score for that day  $MRR_d$  is given by equation 1 below.



$$MRR_d = \left( \sum_{i=1}^Q \frac{1}{r_i} \right) / Q \quad (1)$$

Note that in the special case where the actual query modification is not included in the list of recommended modifications then  $1/r$  is set to zero. The above evaluation process results in a score for each logged day. So overall, the process produces a series of scores for each domain model being evaluated. These scores allow the comparison between different domain models. A model  $M_1$  can therefore be considered superior over a model  $M_2$  if a statistically significant improvement can be measured over the given period.

The described process fits perfectly a static model, but in the case of dynamic experiments as we are conducting here, the experimental process is similar. We start with an initially empty domain model, or an existing domain model. Like before, the model is evaluated at the end of each daily batch of query modifications, but unlike the static experiments it uses the daily data for updating its structure.

### 3 Experimental Setup

The aim of the experiment is to find out whether the performance predicted by *AutoEval* is in line with how users would judge the results. Here, we are not interested in the *absolute* values but instead we would like to know if the *relative* comparison between different systems can be replicated by user judgements. In other words, we would like to find out whether a query suggestion model deemed better by *AutoEval* is in fact producing “better” query suggestions when consulting real users.

We select two adaptive domain models which are continuously learning query modification suggestion from past queries as recorded in log files. In addition, we use an association rule-based approach that operates on the same log data. The three models can be summarized as follows:

- **ACO** uses an ant colony optimization (ACO) approach to learn a graph of related queries that can then be used to make query modification suggestions. The algorithm is described elsewhere [3]. Generally speaking, the model is used to provide suggestions for query modification by first finding the original query phrase in the graph, then listing all the associated nodes (query phrases) ranked by their associated weight.
- **Nootropia** is an immune inspired model for *adaptive information filtering* [9]. Here Nootropia is cast to the problem of continuously learning a domain model for query recommendations, by treating each query as a textual feature and each query session as a “bag” of textual features.
- **Fonseca** is an alternative to graph-based structures which derives query modification suggestions using association rules [4]. The idea is to use session boundaries and to treat each session as a transaction. Related queries are derived from queries submitted within the same transaction.

Following Fonseca’s approach and to reduce noise, in all our experiments we only consider sessions where the number of queries is less than 10 and those which span over the period of less than 10 minutes. We use weekly batches to update the domain models.

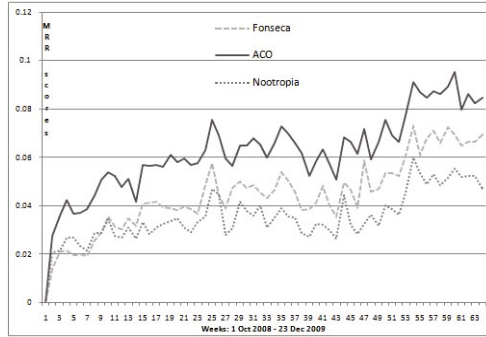
The search log data in our experiments are obtained from the University of Essex website search engine. These logs have been collected since November 2007 (more than 1.5 million queries have been submitted so far). Each record in our query logs contains a time stamp of the transaction, the query that has been entered and the session identifier.

We first run *AutoEval* on the log data over the period of 64 weeks between the beginning of the academic year 2008 to the end of the autumn term in 2009 using the different models for suggesting query modifications. This gives us *MRR* scores for each system on weekly intervals. In order to validate our automatic evaluation methodology we performed a user-based assessment as proposed in the literature [10]. In this approach participants are given queries and their refinements and they are asked to determine whether these refinement suggestions are relevant to the original query. We sampled 20 queries from the entire log data. Apart from frequent queries (that make up a large proportion of all queries) we also sampled queries of medium frequency similar to [1]. We randomly selected 10 queries from the top 50 queries in the log data. Then we selected 10 queries within a range of medium frequency (between 50-1000), these do not overlap with the top 50 queries.

In order to select a sensible number of query modification suggestions, for each sampled query we selected the three best (highest weighted) related terms using five different models:

- **ACO1:** this is the ant colony optimisation model learnt over the entire 64 weeks period used in the *AutoEval* run.
- **ACO2:** this is the ant colony optimisation model learnt over a shorter period which is only the autumn term of the academic year 2008.
- **Fonseca:** this the domain model learnt using Fonseca’s association rules over the entire 64 weeks period used in the *AutoEval* run.
- **Nootropia:** this the domain model learnt using Nootropia over the entire 64 weeks period used in the *AutoEval* run.
- **Baseline:** As a baseline we selected a method that does not rely on log data (and does not get updated in weekly batches). We assume that the top matching results of a commercial search engine will be a useful resource to derive query modification suggestions. We derived nouns and noun phrases from the top ten snippets returned by *Yahoo!* (restricting the search to the University of Essex website). We identify nouns and noun phrases using text processing methods applied in previous experiments [7].

This has resulted in 214 distinct query pairs after removing duplicates due to the overlap of some of the suggestions coming from different systems. An online survey was prepared, and we asked 16 subjects (students and staff at Essex University) to fill in the survey. Participants were not told that various different



**Fig. 1.** AutoEval run for ACO, Nootropia, and Fonseca for a period of 64 weeks

techniques have been used to generate these query pairs. The form contained a list of all query pairs in random order. With each query pair the participants had to decide whether the refinement is relevant or not relevant. They were also given the choice to choose “do not know” if they were not sure.

## 4 Results and Discussion

Figure 1 illustrates the results of running *AutoEval*. We see that despite a few spikes the general trend is upwards indicating that different adaptive learning methods are able to learn from past log data over time. The figure suggests that the ACO method is significantly more effective than learning based on association rules and Nootropia.

The results of the user study are in line with this finding. The aggregated results are shown in Table 1. For each user we calculated the percentage of pairs that were judged relevant and then we aggregated the results among the assessors. ACO is the best performing system overall being significantly better than any of the alternatives ( $p < 0.05$ ). The differences in the user assessment scores reflect the differences observed in Figure 1. The order of the three different adaptive approaches is consistent with the automatic evaluation. The user assessment also shows that ACO and Fonseca adaptive models are considered better by the users than the snippet baseline approach which is in line with our previous experiments [3]. Furthermore, ACO1 is significantly better than ACO2 ( $p < 0.0001$ ), i.e. the increase in performance observed over time in the automatic evaluation is reflected in the user assessment. It also means the ACO adaptive model is capable of learning better query suggestions over time.

**Table 1.** Query suggestions judged ‘relevant’ by users

	ACO1	ACO2	Fonseca	Nootropia	Baseline
Relevant	59.38%	50.00%	55.63%	29.16%	54.06%

As a conclusion, *AutoEval* appears to be a sensible methodology capable of identifying performance improvement of an adaptive model for providing query suggestions over time. We show that this methodology can perform comparative experiments where different adaptive models can be tested under the same experimental conditions. For future work we propose to explore the ACO model with different settings, e.g. updating the weights using clickthrough data by giving more rewards for suggestions that lead to a landing page.

## Acknowledgements

This research is part of the AutoAdapt research project. AutoAdapt is funded by EPSRC grants EP/F035357/1 and EP/F035705/1.

## References

1. Boldi, P., Bonchi, F., Castillo, C., Donato, D., Vigna, S.: Query suggestions using query-flow graphs. In: Proceedings of WSCD 2009, Barcelona, pp. 56–63 (2009)
2. Craswell, N., Szummer, M.: Random Walks on the Click Graph. In: Proceedings of SIGIR 2007, Amsterdam, pp. 239–246 (2007)
3. Dignum, S., Kruschwitz, U., Fasli, M., Kim, Y., Song, D., Cervino, U., De Roeck, A.: Incorporating Seasonality into Search Suggestions Derived from Intranet Query Logs. In: Proceedings of WI 2010, Toronto, pp. 425–430 (2010)
4. Fonseca, B.M., Golgher, P.B., de Moura, E.S., Ziviani, N.: Using association rules to discover search engines related queries. In: Proceedings of the First Latin American Web Congress, Santiago, pp. 66–71 (2003)
5. Joachims, T.: Evaluating retrieval performance using clickthrough data. In: Franke, J., Nakhaeizadeh, G., Renz, I. (eds.) Text Mining, pp. 79–96. Springer, Heidelberg (2003)
6. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Gay, G.: Accurately interpreting clickthrough data as implicit feedback. In: Proceedings of SIGIR 2005, Salvador, pp. 154–161 (2005)
7. Kruschwitz, U.: *Intelligent Document Retrieval: Exploiting Markup Structure*. The Information Retrieval Series, vol. 17. Springer, Heidelberg (2005)
8. Nanas, N., Kruschwitz, U., Albakour, M.-D., Fasli, M., Song, D., Kim, Y., Cervino, U., De Roeck, A.: A Methodology for Simulated Experiments in Interactive Search. In: Proceedings of the SIGIR 2010 SimInt Workshop, Geneva, pp. 23–24 (2010)
9. Nanas, N., Roeck, A.: Autopoiesis, the immune system, and adaptive information filtering. *Natural Computing: an International Journal* 8(2), 387–427 (2009)
10. Sanderson, M., Croft, B.: Deriving concept hierarchies from text. In: Proceedings of SIGIR 1999, Berkeley, CA, pp. 206–213 (1999)
11. Soboroff, I., Nicholas, C., Cahan, P.: Ranking retrieval systems without relevance judgments. In: Proceedings of SIGIR 2001, New Orleans, pp. 66–73 (2001)
12. Zhang, J., Kamps, J.: A search log-based approach to evaluation. In: Lalmas, M., Jose, J., Rauber, A., Sebastiani, F., Frommholz, I. (eds.) ECDL 2010. LNCS, vol. 6273, pp. 248–260. Springer, Heidelberg (2010)

# To Seek, Perchance to Fail: Expressions of User Needs in Internet Video Search

Christoph Kofler, Martha Larson, and Alan Hanjalic

Multimedia Information Retrieval Lab, Delft University of Technology,  
Mekelweg 4, 2628 CD Delft, The Netherlands  
{c.kofler,m.a.larson,a.hanjalic}@tudelft.nl

**Abstract.** This work investigates user expressions of content needs in Internet video search, focusing on cases in which users have failed to meet their search goals, although relevant content is reasonably certain to exist. We study expressions of user needs in the form of requests (i.e., questions) formulated in natural language and published to Yahoo! Answers. Experiments show that classifiers can distinguish requests associated with search-goal failure. We identify a group of ‘easy-to-predict’ requests (cases for which the classifier predicts search-goal failure well) and compile an inventory of strategies used by users to express search goals in these cases. In a final set of experiments, we demonstrate the feasibility of predicting search-goal failure based on query-like representations of the original natural-language requests. The results of our study are intended to inform the future development of indexing and retrieval techniques for Internet video that target difficult queries.

**Keywords:** Multimedia retrieval, Internet video, user information need, search-goal failure, crowdsourcing.

## 1 Introduction

A better understanding of the ways in which users express their needs for Internet video content can inform the development of more effective video indexing and retrieval algorithms. In this work, we investigate how users have formulated their needs in cases in which they have failed to meet their search goals, despite the fact that relevant content is reasonably certain to exist on the Internet. In particular, we focus on expressions of content needs for which it is easy to automatically predict search-goal failure using a conventional text classifier. We are interested in obtaining a better understanding of the strategies that users deploy to express search goals in failure-prone cases, with an eye to our ultimate aim of developing retrieval techniques for Internet video (e.g., query prediction) that will address such cases specifically.

Our collection of expressions of user needs for Internet video is a set of requests for help finding videos that have been posted to the popular question-answering forum Yahoo! Answers (<http://answers.yahoo.com/>). Alternatively, expressions of needs could have been collected via a user study or transaction log analysis.

Yahoo! Answers, however, provides us with a good trade off between data set size and resources invested. Critically, our data set gives us direct access to information concerning failure of users' previous attempts to meet their search goals, which would not be available in a transaction log. We use this data set to investigate three research questions: *Q1: Is search-goal failure predictable given user-expressed content needs?*, *Q2: How are content needs expressed in failure-prone cases?*, *Q3: Is it possible to predict search-goal failure given a conventional keyword query rather than a natural-language expression of a content need?*

Work related to our investigation includes transaction log analysis for multimedia retrieval [1,2] and especially work dealing with video-search tactics [5] and image-search failure [4]. Our work is set apart from previous studies because of its use of user needs that are expressed in natural language and are also explicitly associated with search-goal failure. We adopt our basic definition of user needs and goals from [3], which defines a search goal as, 'an atomic information need, resulting in one or more queries'.

The contributions of this paper are threefold. First, we introduce, for the first time to the best of our knowledge, the use of an Internet question-answering forum for studying expressions of user content needs in the area of video search. Second, we demonstrate experimentally that it is feasible to predict search-goal failures both on the basis of a content need (expressed in natural language) and a query (expressed as a keyword set). Third, we formulate a qualitative characterization of strategies that users use to express their content needs by carrying out a manual analysis of requests associated with 'easy-to-predict' search-goal failure. The paper is structured as follows. We describe our experimental framework in the next section, then we present and discuss our experiments and the results of our analysis and, finally, we finish with conclusion and outlook.

## 2 Data Set and Experimental Set-Up

To create our data set, we used simple heuristics to collect requests from Yahoo! Answers that were identified as related to video search. Requests were considered video-search related if they contained the words 'find' and 'video', but not words like 'game' or 'camera'. The resulting data set is not an exhaustive set of all video-search requests on Yahoo! Answers (our heuristics are not perfect), but does provide us with a sizeable sample, which we take to be representative. Within this set, we are particularly interested in requests in which the user makes a statement that reveals that previous search efforts have not succeeded, as in this example, 'Does anyone know where I can find a video that has the monkey from an old pizza commercial in it? I think he was a sock monkey and I'm not sure what pizza company he was from [...]. I can't find the video anywhere.' In this request, we consider, 'I can't find the video anywhere.' to be a *failure statement* concerning the user's search goal. We refer to requests containing such failure statements as search-goal-failure requests, designated +sgf requests. In contrast, the request, 'Where do i find the video of bon qui qui at king burger? which website ?' does not contain a search-goal failure statement and is designated a -sgf request. It could be argued that if users generally try to find the video

themselves before posting a request on a forum, then all requests on Yahoo! Answers reflect, in a sense, failed search. However, we avoid assumptions about users' search histories, but rather take at face value what they state in their requests. We interpret the presence of a failure statement in a request to signal a particularly confounding case that can be considered 'search-goal failure prone' and deserving of special attention within our study.

We identified **+sgf**/**-sgf** requests in our data set using the crowdsourcing platform Amazon Mechanical Turk (<http://www.mturk.com/>). Workers were first recruited with a trial run that served to ensure that they understood the concept of search-goal failure. Each request was annotated by three recruited workers and we adopted the majority opinion. The final data set contained a total of 592 video search requests, with 213 labeled as **+sgf** and 379 as **-sgf**. Note that **-sgf** is not equated with a satisfied search goal, but rather with an unknown status. For each request we also asked our recruited workers to create a short keyword query that could help to find the content the requester is looking for. On average, term overlap between worker-suggested queries was approximately 63%. We merged the three queries to create a pseudo-query that allows us to carry out experiments on encodings of the user need that are more query-like than the original request, which is expressed in natural language and often quite verbose.

For each request, our data set includes four representations: (1) **need\_exp\_orig** the original expression of the user information need (i.e., video-search request from Yahoo! Answers) (2) **need\_exp\_edit** a version of **need\_exp\_orig** that has been normalized by manually removing any failure statement (3) **query\_orig** the pseudo-query and (4) **query\_exp** an expanded version of the pseudo query. Expansion was carried out using the top-ten videos returned when YouTube (<http://www.youtube.com/>) was queried with the pseudo-query. If, within the top-ten, a term in a title or tag list of a YouTube result had a higher-than-average co-occurrence with a query word, it was used to expand the query. How and why we use these representations is further elucidated in the next section.

Our experiments involve **+sgf**/**-sgf** classification. We use feature vectors of term frequencies, applying feature selection but no other preprocessing. We experiment with two classifiers, standard for text classification problems, a decision tree (J48) and a support vector machine (SVM), using Weka (<http://www.cs.waikato.ac.nz/ml/weka/>) as our implementation. For feature selection we use Weka's **cfssubset**, applying subset selection as the feature evaluator and best first search to select features. The decision tree gives us information about which terms are indicative and the SVM represents a state-of-the-art classifier for text. Results are generated using 5-fold cross-validation.

### 3 Results and Discussion

The results of our experiments, reported in terms of accuracy, are summarized in Table II. We compare each condition with a naïve baseline under which all items are classified into the dominant class, **-sgf**. The naïve baseline achieved an accuracy of 64% (379 **-sgf** requests/592 total requests). All improvements

over the random baseline are statistically significant with respect to Wilcoxon signed-rank test ( $p := 0.05$ ). The first two lines of Table 1 address research question Q1. We see that a standard classifier has the ability to distinguish between `+sgf` and `-sgf` user needs as expressed by original Yahoo! Answers requests (cf. `need_exp_orig` in Table 1). Indicative terms included ‘can’t’, ‘cannot’ and ‘find’, suggesting that the classifier is exploiting features from failure statements to identify `+sgf` requests. In order to focus on specifically the expression of content needs, we removed failure statements from the requests by hand and ran the classification experiment again (cf. `need_exp_edit` in Table 1). The classification performance deteriorated sharply, but we can still answer Q1 with the observation that it appears feasible to automatically generate a prediction of failure using user expressions of Internet video needs.

**Table 1.** Classification accuracy, true positives (TP), true negatives (TN)

	J48 (TP/TN)	SVM (TP/TN)
<code>need_exp_orig</code>	82.8% (168/322)	85.3% (165/340)
<code>need_exp_edit</code>	65.2% (45/341)	73.3% (67/367)
<code>query_orig</code>	64.0% (0/379)	68.8% (28/379)
<code>query_ext</code>	65.0% (24/361)	77.5% (84/375)

We further investigated ‘easy-to-predict’ search goal failure by carrying out an analysis of those expressions of user needs that the SVM classifies correctly. We chose this group because it will be the first focus of our future research, which will be guided by the pragmatic assumption that failure needs to be diagnosable before it can be explicitly addressed. We analyzed 67 true positives by hand, examining the original answers on Yahoo! Answers and also using general Internet search to eliminate examples for which it is highly plausible that search failure can be attributed to the non-existence of relevant material, rather than the particular expression of the user need in the request. What remained was a set of 40 requests for which we are reasonably certain that relevant content exists. We coded the requests with a set of ‘expression strategy’ labels that described the ways that users express their content needs in the requests. We iteratively passed through the examples, refined the coding until all examples were covered by at least one label and no two labels could be reasonably conflated. The result was an inventory of strategies that provide an answer to research question Q2. The strategies are listed in Table 2 and typical examples (which has been edited slightly for length) are given for each. We also note whether requests using that strategy appear to target known items and/or ad hoc sets of videos. It is important to give this inventory the appropriate interpretation. It is a list of expression strategies that we observed are associated with search goal failure. The inventory is not exhaustive and the strategies are not necessarily mutually exclusive. We do not claim that the strategies are either necessary or sufficient to diagnose search-goal failure. Instead, this inventory provides assurance that if we create retrieval algorithms that address user needs expressed with these



**Table 2.** Strategies used by users in the expression of Internet video search needs

Strategies	Example requests from Yahoo! Answers
Uses production information (e.g., title words)	Can someone lhelp me find the official music video to this James Blunt song? its Goodbye My lover. ( <i>known item</i> )
Includes identities (i.e., names of people appearing in or mentioned in the video)	Where can I find the actual video of Taylor Swift and Kanye West? The VMA awards a few weeks ago when Kanye interrupted Taylor and said that Beyonce had a great music video too... ( <i>known item</i> )
Describes what is depicted visually in the video content	I'm looking for a video of a big robot walking through a mall. Can't find it anywhere, thanks for any help. It was a demo of this robot walking through a mall. It was pretty big, about 8ft tall making it's way through a mall. ( <i>known item</i> )  Where can I find a video of a hen protecting her chicks? I'm looking for a video of a hen protecting her chicks under her wings... ( <i>ad hoc</i> )
Specifies theme or topic	i watched a featured video on youtube yesterday about a future shelter that could be built in case of any disasters. it could house up to 4000 people... ( <i>known item</i> )
Specifies particular aspect of a specific version of a video	Anyone know where I can find the video of Spongebob and Patrick making fun of Texas? Just the parts where they're making all the Texas jokes, not the whole video. Everything I find has been dubbed over! ( <i>known item</i> )
Includes a quote in the description	Does anyone know where I can find the video of Jack Buck's post 9/11 speech? I found the video of his poem but it does not include the famous "Should we be here?" quote. ( <i>known item</i> )
Specifies a skill that the user wishes to acquire	Where can I find a diagram/video/etc. that shows the parts of a motorcycle? I'm about to start learning to ride a motorcycle, and I'd like to know where things are on one! ( <i>ad hoc</i> )

strategies that we will indeed be working on at least some difficult cases of video search, rather than exclusively on cases for which existing algorithms already achieve good results.

In a final set of experiments, we investigate a separate, but related issue. We examine the feasibility of predicting search failure using expressions of needs closer to what users generally supply to search engines (i.e., conventional keyword queries). To this end, we perform classification experiments on the pseudo-queries corresponding to the requests (cf. `query_orig` in Table II). Here, the performance drops off dramatically. We conjecture that the queries are simply too short to provide the classifier with enough material to generalize and we carry out an additional experiment using the expanded pseudo-queries (cf. `query_exp` in Table II). The performance still leaves much room for improvement, but does serve to provide a positive answer to research question Q3. The results suggest that it is possible to automatically generate a prediction of search failure given an expression of the user need of the size and form of a conventional query.

## 4 Conclusion and Outlook

We have presented an investigation of the relationship between user expression of Internet video needs and search-goal failure, for which we used an Internet question-answering forum as a novel source of a collection of expression of user information needs. We have determined that it is feasible to automatically generate a prediction of search failure for user needs, both when they are expressed as natural-language video requests and when they are encoded as conventional keyword-based queries. A manual analysis of cases in which users failed to achieve their search goals despite the existence of relevant material on the Internet yielded an inventory of strategies. This inventory provides a picture of how users express their Internet video needs that emphasizes the perspective of difficult cases. We do not claim that it is possible to uniquely identify the exact source of search failure for any given user need or query. For example, the word ‘series’ turned out to be indicative of search failure for the expanded pseudo-queries (cf. `query_exp` in Table I). This may reflect either that ‘series’ are in general difficult for users to find on the Internet, or that the strategy of specifying a series title or description (possibly in lieu of detailed episode information) is not particularly effective with search engines currently available on the Internet. The main conclusions of our study are that failure-prone Internet video search is potentially predictable and that users formulate information needs making use of general strategies that could be targeted by retrieval algorithms in order to improve search performance on difficult Internet video queries. Future work will involve both the extension of the analysis to a larger data set and also more detailed consideration of the answers to the requests posted on Yahoo! Answers. The results of our investigation will serve to inform long-term work on the development of video retrieval techniques for difficult queries.

*Acknowledgments.* The research leading to these results has received funding from the European Commission’s 7th Framework Programme (FP7) under grant agreement no. 216444 (EU PetaMedia Network of Excellence).

## References

1. Huurnink, B., Hollink, L., van den Heuvel, W., de Rijke, M.: Search behavior of media professionals at an audiovisual archive: A transaction log analysis. *Journal of the American Society for Information Science and Technology* 61(6), 1180–1197 (2010)
2. Jansen, B.J., Goodrum, A., Spink, A.: Searching for multimedia: analysis of audio, video and image web queries. *World Wide Web* 3(4), 249–254 (2000)
3. Jones, R., Klinkner, K.L.: Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In: *CIKM 2008: Proceeding of the 17th ACM Conference on Information and Knowledge Management*, pp. 699–708. ACM, New York (2008)
4. Pu, H.-T.: An analysis of failed queries for web image retrieval. *J. Inf. Sci.* 34(3), 275–289 (2008)
5. Wildemuth, B.M., Oh, J.S., Marchionini, G.: Tactics used when searching for digital video. In: *IiX 2010: Proceeding of the Third Symposium on Information Interaction in Context*, pp. 255–264. ACM, New York (2010)

# Passage Reranking for Question Answering Using Syntactic Structures and Answer Types

Elif Aktolga, James Allan, and David A. Smith

Center for Intelligent Information Retrieval, Department of Computer Science,  
University of Massachusetts Amherst, Amherst MA, 01002, USA  
{elif,allan,dasmith}@cs.umass.edu

**Abstract.** Passage Retrieval is a crucial step in question answering systems, one that has been well researched in the past. Due to the vocabulary mismatch problem and independence assumption of bag-of-words retrieval models, correct passages are often ranked lower than other incorrect passages in the retrieved list. Whereas in previous work, passages are reranked only on the basis of syntactic structures of questions and answers, our method achieves a better ranking by aligning the syntactic structures based on the question's answer type and detected named entities in the candidate passage. We compare our technique with strong retrieval and reranking baselines. Experimental results using the TREC QA 1999-2003 datasets show that our method significantly outperforms the baselines over all ranks in terms of the MRR measure.

**Keywords:** Passage Retrieval, Question Answering, Reranking, Dependency Parsing, Named Entities.

## 1 Introduction

Prior work in the area of factoid question answering (QA) showed that the passage retrieval phase is critical [2], [10], [11]: many of the retrieved passages are non-relevant because they do not contain a correct answer to the question despite the presence of common terms between the retrieved passages and the question. Therefore such passages must be eliminated post-retrieval. Otherwise, this affects the succeeding answer extraction phase, resulting in incorrect answers. One cause of this problem is a failure to consider dependencies between terms in the original question and matching query terms in the passage during retrieval. This issue was tackled in recent work [2], [11]: a better ranking was obtained by analyzing syntactic and semantic transformations in correct question and answer pairs, utilizing the fact that if a passage contains an answer to a question (we call this a true QA pair), there is some similarity in their parse structures.

The main problem with approaches in previous work is that dependencies between parts of sentences of a QA pair are evaluated without first considering whether the candidate sentence bears an answer to the question at all [2]. Wrong passages could easily be eliminated by scanning them for possible answer candidates, an approach also employed by humans when searching for an

answer to a question. Since certain syntactic substructures such as noun or verb phrases frequently co-occur in sentences irrespective of the QA pair being true, passages that are merely scored based on such parse structure similarities with the question yield a suboptimal ranking.

For factoid question answering we therefore propose an improved method that performs the candidate-answer check by determining the answer type of the question. For example, ‘Who is John?’ has the answer type *person* and ‘Where is Look Park?’ has the answer type *location*. For such questions we know that the answer passage must contain a named entity of the detected answer type, otherwise the passage is likely to be non-relevant. If a passage passes this initial check, then the parse structures of the QA pair can be analyzed with respect to the named entities that are found as candidate answers for determining the relevance of the answer passage. This ensures that only relevant parts of the syntactic structures are compared and evaluated, and that passages are not accidentally ranked highly if they contain some common subphrase structures with the question.

In this paper, we view the task of enhancing passage retrieval from two angles: the first objective is to improve retrieval per se, i.e. to increase the number of questions for which we retrieve at least one passage containing the right answer. For this, we experiment with various passage retrieval models detailed in Section 3. Then, we aim at obtaining a better ranking so that correct passages are ranked higher than incorrect ones. Our results demonstrate that our improved reranking technique performs up to 35.2% better than previous methods when interpolated with the original retrieval scores.

## 2 Related Work

Question Answering (QA) has been researched extensively since the beginning of the TREC QA evaluations in the late 1990s. Typically, QA systems are organized in a pipelined structure, where the input to the system is a natural language question and the output is a ranked list of  $n$  answers. The importance of the retrieval component in a QA system has been highlighted in the field [2, 9, 10, 11]. If a bad set of passages is retrieved, not even containing a single answer to the posed question, the QA system fails at the retrieval step itself for that question. The ranking of the passages is also very important, since answer extraction is typically applied to the top-ranked list of retrieved passages rather than to the whole set.

In order to overcome the limitations of passage retrieval, reranking methods have been applied as a post-retrieval step. Cui et al. [2] proposed a fuzzy relation matching technique that learns a dependency parse relation mapping model by means of expectation maximization (EM) and translation models. This model measures sentence structure similarity by comparing dependency label paths between matched terms in questions and answers. When matching terms, only the most probable alignment of terms is considered. Wang et al. [11] developed an improved approach to reranking passages: their model learns soft alignments

by means of EM, however instead of translation models they used a probabilistic quasi-synchronous grammar that allows more flexible matching of subsets of parse trees (called ‘configurations’) rather than single terms. Quasi-synchronous grammars were originally proposed by Smith and Eisner [8] for machine translation. Unlike Cui et al.’s approach, the final score for a passage under this approach is obtained by summing over all alignments of terms. Our work combines some of the advantages from both papers, extending them in a new direction: we train Cui et al.’s model by means of EM and translation models *with respect to the question’s detected answer type and named entities in the answer passage*. We also employ more flexible matching of terms by means of Wordnet synonyms and we sum over all alignment scores as Wang et al.

We use the open domain question answering system OpenEphyra 0.1.1 as our system [5], [6], [7], which is a full implementation of a pipelined question answering system. Since we only measure passage retrieval and reranking performance, we disabled the answer extraction component. For analyzing parse structures of questions and answers, we integrated the dependency parser MSTParser [3] into the system, and extended OpenEphyra further for our passage retrieval and reranking algorithms.

### 3 Passage Retrieval

In this section we describe the passage retrieval techniques that are applied before passage reranking. All our algorithms employ the query likelihood language model as their basis. The differences in our techniques are in how query generation and formulation are achieved.

We create our passages as follows: paragraphs in the TREC QA datasets are processed with OpenNLP’s sentence detector so that they can be broken down into sentences where possible. So ideally passages correspond to sentences. This yields the best results in our experiments, since returned passages contain as little non-relevant material as possible. Since we only apply our methods to factoid questions, for which the answer is always contained within a single sentence, this representation is sufficient. Further, this allows us to compare our methods to previous work, such as Cui et al. [2].

#### 3.1 Bag Of Words (Q-BOW)

We begin with our simplest baseline model *Q-BOW*, which is just a query likelihood of unigram phrases. Mathematically, we can state this model as:

$$P(Q|D) = P(q_1, \dots, q_n | M_D) = \prod_{i=1}^n P(q_i | M_D) \quad (1)$$

where  $M_D$  is the language model of passage  $D$ . We use Dirichlet smoothing with  $\mu = 2500$  for our experiments.

### 3.2 Adding Question Analysis (QuAn)

For the next baseline, *QuAn*, we extend *Q-BOW* in several ways: we allow the addition of n-gram phrases as keywords so that the model does not consist of unigrams only; further, we add the output of OpenEphyra’s front end question analysis phase. This generates two types of phrase queries from a question, question reformulations and question interpretations [5].

Mathematically, we use the same model as in (1), with the difference that the  $q_i$  can now be n-gram phrases referring to reformulations, interpretations, or phrases extracted from the question.

### 3.3 Expanding with Synonyms (QuAn-Wnet)

*QuAn-Wnet* is a further extension of our previous techniques, for which we expand the query generated by *QuAn* with at most 10 keywords obtained from Wordnet. These keywords are n-gram synonyms of phrases generated through *Q-BOW* and *QuAn*.

## 4 Passage Reranking

The next step is to rerank passages, for which we take the output from the passage retrieval phase as generated by *QuAn-Wnet*, and apply one of our reranking techniques to it. We use *QuAn-Wnet* since it performs best (Section 5).

### 4.1 Extraction of Dependency Relation Paths

A dependency parse of a sentence yields a parse tree consisting of nodes that correspond to words in the sentence and labelled edges describing the type of *dependency relation* between two connected nodes or words. Therefore, a *dependency relation path* is a sequence of dependency relations between two nodes in the parse tree. For example, from the parse of the sentence ‘But there were no buyers’ depicted in Figure 1, we can infer the relation path DEP NP-PRD DEP between the words ‘But’ and ‘no’. Dependency relations are usually directed, but we ignore the directions of dependency relations for the analysis of questions and answers as in previous work [2].

Let  $q_i$  denote a word in the question  $Q = q_1, \dots, q_n$  and  $a_i$  a word in the answer  $A = a_1, \dots, a_m$ . We extract relation paths from questions and answers

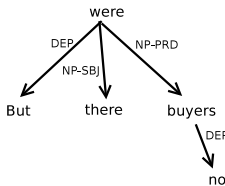


Fig. 1. Dependency Parse tree for the sentence ‘But there were no buyers’

whenever we find a matching pair  $[(q_k, a_l), (q_r, a_s)]$ , where  $k \neq r$  and  $l \neq s$ . In this work, a *match* between two words  $w_i$  and  $w_j$  occurs if  $w_i$  and  $w_j$  share the same root (or stem) or if  $w_i$  is a synonym of  $w_j$ . Therefore matching words can only be nouns, verbs, adjectives, and adverbs. For our improved reranking methods employing question words and candidate answer terms (see Section 4.4). As is the case with IBM translation models, words are matched in a many-to-one fashion due to computational reasons. That is, each word in the question is assigned at most one word from the answer, but answer words can be assigned to multiple question words.

In order to decide whether an answer passage and a question belong together, we compare all possible extracted dependency relation paths from the question and the answer. The idea behind this approach is to find certain patterns of relation paths by means of which we can detect true QA pairs to score them higher than other pairs. This is the foundation of the dependency relation path based reranking methods detailed below.

## 4.2 Training the Model

Given a matching pair  $[(q_k, a_l), (q_r, a_s)]$ , we extract from this the relation paths  $\langle path_q, path_a \rangle$ , where  $path_q$  is the relation path between  $q_k$  and  $q_r$  in the question and  $path_a$  is the path between  $a_l$  and  $a_s$  in the answer accordingly. In order to compare the extracted dependency relation paths  $\langle path_q, path_a \rangle$ , we need a model that captures which relation paths are more probable to be seen together in true QA pairs. For this, we trained a translation model with GIZA++ using IBM Model 1 as described by Cui et al. [2]. We extracted 14009 true QA pairs from sentences in our training set of the TREC QA 1999-2003 corpora. The trained translation model on the relation paths  $\langle path_q, path_a \rangle$  then yields the probability  $P(label_a | label_q)$ , where  $label_a$  is a single dependency relation in  $path_a$ , and  $label_q$  is a relation in  $path_q$ . We use these probabilities in our reranking techniques detailed below to score  $\langle path_q, path_a \rangle$ .

## 4.3 Dependency Relation Path Similarity (Cui)

This technique is our improved implementation of Cui et al.'s [2] approach. It reranks passages only based on the similarity of the dependency relation paths between passages and questions. More specifically, given a question  $Q$  and an answer candidate passage  $A$ , we score  $A$  as follows:

$$\sum_{\langle path_q, path_a \rangle \in Paths} scorePair(path_q, path_a) \quad (2)$$

That is, we sum up the scores of all extracted  $\langle path_q, path_a \rangle$  pairs by aligning  $Q$  and  $A$  in all possible ways [11]. This is different from previous work [2], where only the most probable alignment is considered. The score for an individual path pair  $scorePair(path_q, path_a)$  is then calculated as follows:

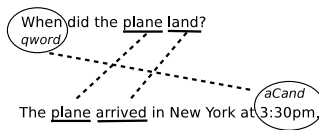
$$\frac{1}{|path_a|} \prod_{label_{a_j}} \sum_{label_{q_i}} P(label_{a_j}|label_{q_i}) \quad (3)$$

where  $P(label_{a_j}|label_{q_i})$  is obtained from our trained translation model, and  $\frac{1}{|path_a|}$  is used as a normalization factor, since the lengths of the answer paths vary whereas those of the question remain the same [2]. We adapt an IBM Model 1 way of scoring here with a product of a sum over the labels since we consider all possible alignments. The derivation of this formula for translation modeling is described in detail in Brown et al.’s work [1].

#### 4.4 Dependency Relation Path Similarity with Answer Type Checking and Named Entities (Atype-DP)

This is our improved reranking method for which we employ answer type checking and the analysis of named entities (NE) in the candidate answer passages. For answer type checking, we use OpenEphyra’s answer type classification module [7], which can detect 154 answer types organized in different hierarchies. Therefore, often several answer types of varying granularity are assigned to a question. This allows a greater flexibility when matching answer types to named entities. The detected answer types in the question are used to look for candidate answer terms in passages. For named entity detection, we use OpenEphyra’s built-in named entity extraction module [5], which comprises about 70 NE types.

For this model, we revise the definition of how *matching* is performed, originally introduced in Section 4.1: a matching pair is now a tuple  $[(qword, aCand), (q_r, a_s)]$ , where  $qword \neq q_r$  and  $aCand \neq a_s$ , with  $qword$  being the question word (e.g. ‘who’), and  $aCand$  a candidate answer term (e.g. ‘Kate’) matching the question word’s answer type.  $(q_r, a_s)$  are other words fulfilling the earlier criteria for matching (words sharing the same root or being synonyms). Figure 2 shows an example.



**Fig. 2.** QA pair with the question word, candidate answer, and matching terms highlighted. There are two matching pairs:  $[(when, 3:30pm), (plane, plane)]$  and  $[(when, 3:30pm), (land, arrived)]$ .

Note that there can be multiple answer candidates of the same NE type in a single answer passage. In this model, we consider all alignments of  $Q$  and  $A$  given the best answer candidate. Under this model, the score for a passage  $A$  can formally be stated as:

$$\max_i \sum_{\langle path_q, path_a \rangle \in Paths_{aCand_i}} scorePair(path_q, path_a) \quad (4)$$



where  $Paths_{aCand_i}$  are all dependency relation paths  $\langle path_q, path_a \rangle$  extracted from matching pairs containing the answer candidate  $aCand_i$ . The calculation of score  $scorePair(path_q, path_a)$  remains the same as (3), with the exception that we use a retrained translation model with our revised matching approach. The probabilities  $P(label_{a_j} | label_{q_i})$  are therefore obtained from this new translation model.

The main difference of this reranking approach to previous techniques [2], [11] lies in how we perform matching: by considering the answer type and the best answer candidate during matching, we ensure that only relevant dependency relation paths between the candidate answer (or question word) and other matching terms are considered. This way, the obtained score reflects the dependencies within a passage towards its relevance for a given question more accurately. In previous approaches, arbitrary relation paths were compared in QA pairs, which often does not yield a good ranking of passages.

#### 4.5 Interpolation with Retrieval Baseline (Atype-DP-IP)

This is a variation of the *Atype-DP* reranking method where we score a passage  $A$  given a question  $Q$  by interpolating *Atype-DP* with the best retrieval baseline *QuAn-Wnet* as follows:

$$score(A) = \lambda score_{QuAn-Wnet}(A) + (1 - \lambda) score_{Atype-DP}(A) \quad (5)$$

$score_{QuAn-Wnet}$  is the retrieval score, whereas  $score_{Atype-DP}$  is our reranking score. In order to interpolate the scores we normalize them so that they are between 0 and 1 and rerank passages based on this new score. The advantage of this approach is that the original retrieval scores are not discarded and impact the results. This helps in cases where *Atype-DP* does not work well due to poor named entity detection, as we will see in Section 5.

#### 4.6 Elimination of Non-Answer-Type-Bearing Passages (QuAn-Elim)

This approach is different from the other reranking methods in that it does not rearrange the order of the retrieved passages, but it eliminates retrieved passages that are likely to not contain an answer based on candidate named entity matches, acting similarly to a passage answer type filter in other QA systems [4]. This approach is interesting for comparing with our reranking methods *Atype-DP* and *Atype-DP-IP* that involve an analysis of parse structures.

## 5 Experiments

The objectives of our research are (1) to show that we can improve the retrieval of passages by employing models that exploit findings from the question analysis phase and synonyms of query terms; (2) to demonstrate that we can obtain a better ranking of passages by employing a candidate-answer check and by

**Table 1.** Evaluation of Retrieval Baselines. All results are averages from the testing datasets TREC 2000 and TREC 2001.

<i>Model</i>	<i>Success@5</i>	<i>Success@10</i>	<i>Success@20</i>	<i>Success@100</i>
Q-BOW	0.325	0.43	0.512	0.732
QuAn	0.356	0.457	0.552	0.751
QuAn-Wnet	<b>0.364*</b>	<b>0.463</b>	<b>0.558</b>	<b>0.753</b>

analyzing their parse structure with respect to candidate named entities. We compare our reranking techniques *Atype-DP* and *Atype-DP-IP* with *Cui* and *QuAn-Elim* only, since in Cui et al.’s paper the technique was shown to be superior to various existing methods [2].

## 5.1 Data and Evaluation

For all our experiments, we use the TREC QA 1999-2003 corpora, questions, and evaluation data. For correct evaluation, we eliminated NIL questions. We split the questions into a test set with 1125 questions, consisting of the TREC QA 2000 and TREC QA 2001 data, and a training set, comprising of the remaining 1038 TREC QA 1999-2003 questions. The training set is also used for the translation models described in Section 4.2.

We determine the correctness of a passage with respect to the question as follows: we consider a passage as relevant and bearing the correct answer to a question if (1) it comes from a document with a correct document ID, and (2) it contains at least one answer pattern.

In the next sections, we report the results for the retrieval and reranking performances separately. For measuring retrieval performance, we use the success measure, which determines the percentage of correctly answered questions at different ranks. For reporting passage ranking quality, we utilize the mean reciprocal rank measure (MRR), which has widely been used in QA for answer ranking.

## 5.2 Retrieval Performance

For this evaluation we utilized 1074 questions from TREC 2000 and TREC 2001 – these are all questions for which queries for retrieval were successfully generated by the question analysis phase in OpenEphyra. Table 1 shows the results of the runs with the *Q-BOW*, *QuAn*, and *QuAn-Wnet* baselines. We can clearly see that over all ranks, as the baseline retrieval method uses more sophisticated queries, the percentage of correctly answered questions increases in terms of the success measure. Since *QuAn-Wnet* performs best (and significantly better with p-value smaller than 0.01 for high ranks), we used this retrieval baseline for our further experiments.

## 5.3 Reranking Performance

For our reranking evaluation, we utilized 622 questions from TREC 2000 and TREC 2001 due to the following requirements: (1) Successful question analysis

**Table 2.** Evaluation of Reranking Techniques. All results are averages from the testing datasets TREC 2000 and TREC 2001, evaluated on the top 100 retrieved passages.

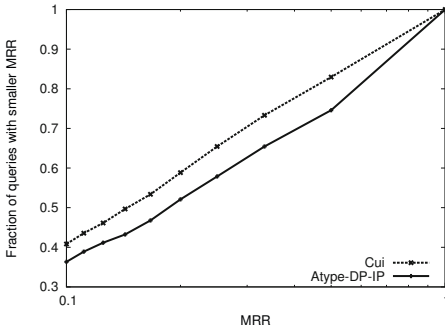
<i>Model</i>	<i>MRR@1</i>	<i>MRR@5</i>	<i>MRR@10</i>	<i>MRR@20</i>	<i>MRR@50</i>	<i>MRR@100</i>
Q-BOW	0.168	0.266	0.286	0.293	0.299	0.301
QuAn-Wnet	0.193	0.289	0.308	0.319	0.324	0.325
Cui	0.202	0.307	0.325	0.335	0.339	0.341
Atype-DP	0.148	0.24	0.26	0.273	0.279	0.28
Atype-DP-IP	<b>0.261*</b>	<b>0.363*</b>	<b>0.38*</b>	<b>0.389*</b>	<b>0.393*</b>	<b>0.394*</b>
% Improvement over Cui	<b>+29.2</b>	+18.24	+16.9	+16.12	+15.9	+15.54
% Improvement over QuAn-Wnet	<b>+35.2</b>	+25.6	+23.4	+21.9	+21.3	+ 21.2

is required as for retrieval: We eliminate questions for which OpenEphyra could not generate a query; (2) As with retrieval, at least one true passage must be present within the first 100 ranks of retrieved passages with *QuAn-Wnet*, since this is the range of passages we parse and analyze. (3) Candidate answer type detection with OpenEphyra must be successful. Since we measure the success of our reranking techniques depending on candidate answer type checking, we only evaluate those questions for which our system detects an answer type.

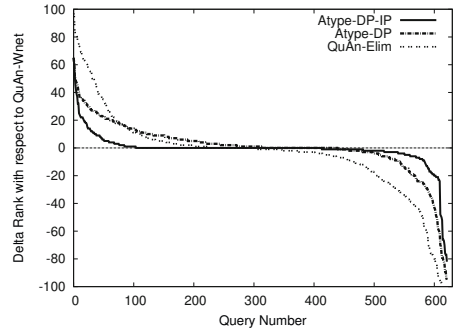
Table 2 illustrates the reranking results with the techniques *Cui*, *Atype-DP*, *Atype-DP-IP*, and the retrieval baselines. We first note that while our reranking approach *Atype-DP* performs worst, its interpolated version *Atype-DP-IP* significantly outperforms all other techniques with p-value less than  $10^{-4}$  over all ranks. The highest percent improvement is at rank 1 with a gain of 29.2% over *Cui* and 35.2% over the retrieval baseline *QuAn-Wnet*. We investigated the reasons for this performance: *Atype-DP* depends on successful named entity extraction, since dependency relation paths are extracted from passages between matching terms and a candidate answer of the required answer type. For many passages though, OpenEphyra’s named entity extraction module could not detect a single named entity of the required answer type. Hence, affected passages are ranked low.

We further observe that our enhanced implementation of Cui et al.’s technique *Cui* [2] performs worse than *Atype-DP-IP* and only a little better than *QuAn-Wnet*. Wang et al. [11] also report the method being rather brittle. This technique does not depend on other factors than retrieval and dependency parsing. Figure 3 provides a more accurate view of the results: The MRR scores of *Cui* and *Atype-DP-IP* are plotted as a cumulative distribution function. Over all ranks, *Atype-DP-IP* has a smaller fraction of questions with smaller MRRs than *Cui*.

We found that the technique *QuAn-Elim*, which does not use any parse structure information – but merely eliminates non-answer bearing passages – on average performs comparably to *Atype-DP-IP*, although it slightly suffers at rank 1 with an MRR of 0.243. This difference is however not significant. A detailed analysis of the three methods *Atype-DP*, *Atype-DP-IP*, and *QuAn-Elim* can be seen in Figure 4: we compared the differences in ranking of the first correct answer



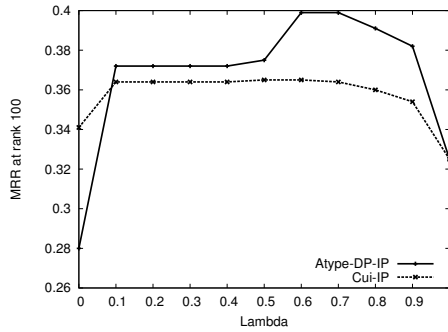
**Fig. 3.** Cumulative distribution function of the MRR scores in logscale (lower is better)



**Fig. 4.** Differences in Ranking of the first correct answer passage in Atype-DP-IP, Atype-DP, and QuAn-Elim with respect to the retrieval baseline QuAn-Wnet

passage of these methods with respect to the baseline *QuAn-Wnet*. *Atype-DP* and *Atype-DP-IP* improve about 310 questions, whereas *QuAn-Elim* only ranks 230 questions higher than the baseline. Note that *QuAn-Elim* ranks some of these questions higher than the other methods. *Atype-DP* also ranks questions higher than *Atype-DP-IP*, since the parse structure analysis with respect to NEs works well. In the middle range, *Atype-DP-IP* has a larger number of questions whose first correct answer passage appears in the same rank as in the retrieval baseline. We analyzed these 300 questions, of which 119 already have a correct answer at rank 1, so they cannot be improved further. On the right end we can observe that *QuAn-Elim* decreases the ranking of correct answers for more questions, and to a higher degree than the other two methods: whereas we get a good gain for a small number of questions, the method worsens performance for about 260 questions. As for the other methods we notice that *Atype-DP* ranks roughly the same amount of questions much lower than *Atype-DP-IP*: This is where the interpolation helps. If named entity detection does not work, the retrieval score prevents the passage from being ranked low. Increasing the impact of the retrieval score further however disturbs the ranking as it can be seen in Figure 5: we adjusted the interpolation parameter  $\lambda$  to achieve the best performance at  $\lambda = 0.7$  as observed in training data. These observations support the hypothesis that the reranking technique is useful on its own, and that the retrieval score aids in recovering errors arising from poor named entity detection.

The results in Table 2 suggest that since *Atype* performs better than all other techniques by means of a little tweak with the baseline, then *Cui* should perform even better than *Atype-DP-IP* when interpolated with the retrieval baseline. Surprisingly, the results in Table 3 and Figure 5 show that *Cui* does not improve with the same effect. This supports our hypothesis that reranking by parse structure analysis with respect to candidate named entities is more effective in optimizing precision.



**Fig. 5.** Effect of varying  $\lambda$  in the interpolated methods Atype-DP-IP and Cui-IP

**Table 3.** Average MRR results of interpolating Atype-DP-IP and Cui-IP with  $\lambda = 0.7$  for the retrieval baseline

<i>Model</i>	<i>MRR@1</i>	<i>MRR@5</i>	<i>MRR@10</i>	<i>MRR@20</i>	<i>MRR@50</i>	<i>MRR@100</i>
Atype-DP-IP	0.261	0.363	0.38	0.389	0.393	0.394
Cui-IP	0.225	0.331	0.347	0.356	0.361	0.362

## 6 Conclusions

We have presented a passage reranking technique for question answering that utilizes parse structures of questions and answers in a novel manner: the syntactic structures of answer passages are analyzed with respect to present candidate answer terms, whose type is determined by the question’s answer type. This way we ensure that only relevant dependency relation paths between the candidate answer and other parts of the sentence are analyzed with respect to the corresponding paths in the question. Our results show that this method outperforms previous approaches and retrieval baselines up to 35.2%, given that cases where named entity extraction does not succeed are backed off to the retrieval score. In future work, we expect that utilizing a more accurate named entity extraction module with respect to answer types will improve the results even further. It would also be interesting to compare our reranking techniques with those of Wang et al., who take a completely different approach to the problem.

**Acknowledgments.** This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors’ and do not necessarily reflect those of the sponsor.

## References

1. Brown, P.F., Pietra, V.J., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics* 19, 263–311 (1993)

2. Cui, H., Sun, R., Li, K., Kan, M.-Y., Chua, T.-S.: Question answering passage retrieval using dependency relations. In: SIGIR 2005, pp. 400–407. ACM, New York (2005)
3. McDonald, R., Crammer, K., Pereira, F.: Online large-margin training of dependency parsers. In: ACL 2005, Morristown, NJ, USA, pp. 91–98 (2005)
4. Prager, J., Brown, E., Coden, A., Radev, D.: Question-answering by predictive annotation. In: SIGIR 2000, pp. 184–191. ACM, New York (2000)
5. Schlaefler, N., Gieselman, P., Sautter, G.: The Ephyra QA system at TREC 2006 (2006)
6. Schlaefler, N., Gieselmann, P., Schaaf, T., Waibel, A.: A pattern learning approach to question answering within the ephyra framework. In: Sojka, P., Kopeček, I., Pala, K. (eds.) TSD 2006. LNCS (LNAI), vol. 4188, pp. 687–694. Springer, Heidelberg (2006)
7. Schlaefler, N., Ko, J., Betteridge, J., Pathak, M.A., Nyberg, E., Sautter, G.: Semantic Extensions of the Ephyra QA System for TREC 2007. In: TREC (2007)
8. Smith, D.A., Eisner, J.: Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In: Proc. HLT-NAACL Workshop on Statistical Machine Translation, pp. 23–30 (2006)
9. Sun, R., Ong, C.-H., Chua, T.-S.: Mining dependency relations for query expansion in passage retrieval. In: SIGIR 2006, pp. 382–389. ACM, New York (2006)
10. Tellex, S., Katz, B., Lin, J., Fernandes, A., Marton, G.: Quantitative evaluation of passage retrieval algorithms for question answering. In: SIGIR 2003, pp. 41–47. ACM, New York (2003)
11. Wang, M., Smith, N.A., Mitamura, T.: What is the Jeopardy model? A quasi-synchronous grammar for QA. In: EMNLP-CoNLL 2007, pp. 22–32. Association for Computational Linguistics, Prague (2007)

# An Iterative Approach to Text Segmentation

Fei Song<sup>1</sup>, William M. Darling<sup>1</sup>, Adnan Duric<sup>1</sup>, and Fred W. Kroon<sup>2</sup>

<sup>1</sup> School of Computer Science, University of Guelph, 50 Stone Road East,  
Guelph, Ontario, N1G 2W1, Canada

{fsong, wdarling, aduric}@uoguelph.ca

<sup>2</sup> PryLynx Corporation, 21 Oneida Place, Kitchener, Ontario, N2A 3G2, Canada  
fred.kroon@prylynx.com

**Abstract.** We present *divSeg*, a novel method for text segmentation that iteratively splits a portion of text at its weakest point in terms of the connectivity strength between two adjacent parts. To search for the weakest point, we apply two different measures: one is based on language modeling of text segmentation and the other, on the interconnectivity between two segments. Our solution produces a deep and narrow binary tree – a dynamic object that describes the structure of a text and that is fully adaptable to a user’s segmentation needs. We treat it as a separate task to flatten the tree into a broad and shallow hierarchy either through supervised learning of a document set or explicit input of how a text should be segmented. The rich structure of our created tree further allows us to segment documents at varying levels such as topic, sub-topic, etc. We evaluated our new solution on a set of 265 articles from *Discover* magazine where the topic structures are unknown and need to be discovered. Our experimental results show that the iterative approach has the potential to generate better segmentation results than several leading baselines, and the separate flattening step allows us to adapt the results to different levels of details and user preferences.

**Keywords:** Text Segmentation; Language Modeling.

## 1 Introduction

A natural language document typically has an underlying hierarchical organization where one or few common themes are supported by a set of interrelated topics, which are in turn made up of subtopics, sub-subtopics, etc. Text segmentation is the task of dividing a document into a sequence of segments that correspond to its constituent parts. Traditionally, segmentation has been seen as a linear task where breaks are inserted into a flat text. However, full text segmentation would allow segments to be further divided into smaller segments, or “subtopics”, to the point where a full underlying hierarchical organization of a document could be discovered.

Text segmentation is useful for many applications including information retrieval, text summarization, and information visualization. In information retrieval, for example, a query is matched with either documents or passages of the

documents. The drawback is that the user may get too many passages or have to examine long documents to find the relevant information contained within. By breaking documents into topics and / or subtopics, we can use the related segments as suitable search results that provide greater precision. Text segmentation is also an important part of topic detection and tracking research (TDT). TDT aims to discover and correlate related topics in streams of data such as broadcast news for eventual archiving and other uses. An overview is available in [3].

There are three major approaches to text segmentation in the literature. They include similarity curves ([7]), dotplots ([9]), and language models ([11]). These methods are similar in that they all use word frequency metrics to measure the similarity between two regions of text so that a document can be separated at the points where the connections between the regions are weak, but within the regions are strong. They differ, however, in how the similarity between portions of the text is used to find such regions. In [7], Hearst computes the similarity within a sliding window and follows the peaks and valleys of the similarity curve to determine where to segment a text. In [9], Reynar calculates a similarity matrix between all pair-wise sentences and identifies patterns along the diagonal line for separating segments. In [11], Utiyama and Isahara generate all possible segment partitions through dynamic programming and use the probability distribution of the words to rank and select the best segment partitions.

In this paper, we propose a new method for text segmentation that iteratively partitions a portion of the text in order to build a hierarchical organization of the input. Similar to Utiyama and Isahara's language model, we generate multiple partitions for a document, but instead of enumerating all possible partitions with a different number of segments, we examine all the two-way partitions for a portion of the text and select the weakest point and then split the text into two parts. By doing such two-way partitions iteratively, we build a deep and narrow binary tree as the output. We leave it as a separate step to flatten the tree into a broad and shallow hierarchy, since there are considerable variations among different users in producing such hierarchical organizations ([7]) and many approaches take the number of desired segments for each document as an input from the user ([4] and [5]). We demonstrate our new method's segmentation ability on a set of 265 articles from *Discover* magazine. The topic structures of the articles are unknown; it is therefore desirable to re-discover the underlying hierarchical organizations. The experimental results demonstrate the effectiveness of our new method.

The rest of the paper is organized as follows. In section 2, we introduce related work that helps us measure the connectivity strength between two adjacent segments. In section 3, we describe the detailed steps in our iterative approach for text segmentation. Our solution produces a deep and narrow hierarchy and we offer two possible methods for evaluating hierarchical organizations of text segmentation in section 4. In section 5, we discuss our experimental results on the *Discover* dataset, and finally, section 6 concludes the paper with some future research directions.



## 2 Related Work

Lexical cohesion refers to the connectivity between two portions of text in terms of word relationships ([6]). Although there can be different kinds of relationships between two words, including synonymy (the same meaning) and hyponymy (where one word is a more specific instance of another), the simplest form is when two words are identical or share the same morphological root. Lexical cohesion is commonly modeled by the interconnectivity between sentences in terms of word overlaps or similarities ([10]).

Based on the sentence-level similarity, [12] defines a general similarity measure for the interconnectivity between two adjacent segments in a document as

$$sim_{between} = \frac{\sum_{i=1}^m \sum_{j=1}^n sim_{ij}}{m \times n} \tag{1}$$

where  $m$  and  $n$  are the numbers of sentences in two adjacent segments, and  $sim_{ij}$  is the similarity between sentences  $i$  and  $j$ .

Clearly, the similarity between two sentences that are close to each other should weigh more than that between two sentences that are further apart. Accordingly, [12] also offers a distance-based interconnectivity measure between two segments as

$$sim_{between} = \frac{\sum_{i=1}^m \sum_{j=1}^n w_{ij} sim_{ij}}{m \times n} \tag{2}$$

where  $w_{ij} = 1$  for  $|i - j| \leq 2$  or  $\frac{1}{\sqrt{|i-j|-1}}$ , otherwise.

Alternatively, we can follow Utiyama and Isahara’s approach in [11] and use a cost function to measure the strength for a segment partition of a document. Let  $W = w_1 w_2 \dots w_n$  be a document of  $n$  words and  $S = S_1 S_2 \dots S_m$  be a partition of  $m$  segments. The cost for this partition is defined as follows:

$$\begin{aligned} C(S) &= -\log P(W|S)P(S) \\ &= -\sum_{i=1}^m \sum_{j=1}^{n_i} \log P(w_j^i | S_i) + m \log n \\ &= -\sum_{i=1}^m \sum_{j=1}^{n_i} \log \frac{f_i(w_j^i) + 1}{n_i + k} + m \log n \\ &= \sum_{i=1}^m \left[ \sum_{j=1}^{n_i} \log \frac{n_i + k}{f_i(w_j^i) + 1} + \log n \right] \end{aligned} \tag{3}$$

Here,  $n_i$  denotes the length of segment  $S_i$  and  $w_j^i$  gives the  $j$ th word in segment  $S_i$ . In addition,  $f_i(w_j^i)$  stands for the frequency of word  $w_j^i$  in  $S_i$ , while  $k$  is the total number of unique words in a given document.

Intuitively, the cost function measures the strength of a segment partition: the lower the value, the weaker the interconnectivity between the segments, and

therefore, the better the choice for text segmentation. Since we are only interested in two-way partitions in our iterative approach, *i.e.*, splitting a portion of text into two segments at a time, we can simplify equation (3) into the following:

$$C(S) = \sum_{i=1}^2 \sum_{j=1}^{n_i} \log \frac{n_i + k}{f_i(w_j^i) + 1} \quad (4)$$

### 3 Iterative Text Segmentation

#### 3.1 Motivation for an Iterative Approach

Most existing text segmentation methods are aimed at finding a suitable linear segment partition for a given document, at either a topic or a subtopic level. To recover the underlying hierarchical organization of a naturally occurring document, these methods may need to be applied recursively: first, by breaking a document into a sequence of topics, and then, by breaking large segments into sequences of subtopics.

One problem with this approach is that each application of the method can be quite expensive. For example, in both [11] and [12], one needs to enumerate all possible partitions for a different number of segments in order to select the best possible segment partition. Such a process is expensive even using dynamic programming techniques to save the intermediate results.

Another problem is that a system needs to be tuned to model a typical / average topic or subtopic partition for human editors. As observed by Hearst in [7], wide variations typically exist among human editors about the underlying hierarchical organization of a document. Some are fine-grained while others are coarse-grained, and even for the same editor, some parts of the organization can be detailed while other parts are brief, depending on the editors background knowledge and interest. As a result, it will be difficult to have one size to fit all variations.

In this paper, we propose a new method for text segmentation that iteratively splits a portion of the text at its weakest point in terms of the interconnectivity between the two parts. Such two-way partitions are easy to implement and efficient to compute. In addition, we build a complete binary tree of the partitioned segments as the output of this iterative process. We treat it as a separate task to flatten the tree into a broad and shallow hierarchy so that we can adapt the results to different levels of details and user preferences or needs.

#### 3.2 Detailed Steps

Our method performs two-way partitions for a document or a portion of the text. Given a range of sentences, our method will try different split points using the interconnectivity measures described in equations (2) or (4) above so that we can find the weakest point to split the text into two segments. This is implemented as the *InterSim* function mentioned below. Intuitively, it should be easier and perhaps more reliable to break a portion of the text at its weakest point than

**Input:** *start* and *end* sentence positions, and the document similarity *index*

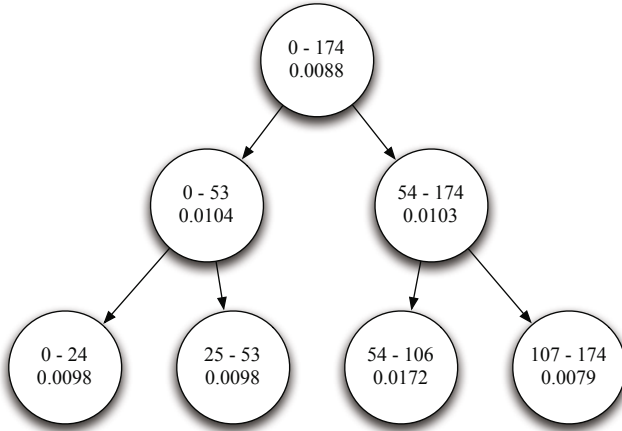
**Output:** the partitioned segments as a binary *tree*

```

begin
  set tree to empty;
  if (end - start) > MinSegmentSize then
    set bestSim to MaxValue;
    set bestSplit to -1;
    foreach position i between start and end do
      set sim to the result of InterSim(start, i, end, index);
      if sim < bestSim then
        bestSim = sim;
        bestSplit = i;
      end
    end
    end
    if bestSplit > 0 then
      set tree to a new node covering sentences start to end;
      add SplitSegment(start, i, index) to the children list of tree;
      add SplitSegment(i + 1, end, index) to the children list of tree;
    end
    return tree
  end
end
end

```

**Algorithm 1.** SplitSegment Function



**Fig. 1.** A Binary Tree of the Segmented Document Structure Created by *divSeg*

finding multiple split points as required by most of the existing methods for text segmentation.

Based on the *InterSim* function (either equation (2) or (4)), we will iteratively partition a portion of the text until it is too small to be split (less than or equal to the minimum segment size). We summarize all the related steps in the following algorithm for clarity. Note that variables are displayed in italics.

As can be seen from the *SplitSegment* function detailed in Algorithm 1, two-way partitions are both easy to implement and efficient to compute, since each time we split, the remaining segments are made smaller. This is a typical application of the divide-and-conquer principle. Our output is a binary tree of all the partitioned segments, which can later be flattened into a desirable hierarchical organization, given the user's segmenting needs (please refer to subsection 4.2 for details).

A portion of an example binary tree is shown in Figure 1, where each node contains the *start* and *end* sentence positions for the portion of text it covers, along with a score referring to the interconnectivity strength for splitting it into the two segments at the children level. For example, the left child of the root node covers sentences from 0 to 53 and the score for splitting it into two segments (sentences from 0 to 24 and the sentences from 25 to 53) is 0.0104.

### 3.3 Comparisons with Related Work

Our iterative approach for text segmentation relies on the interconnectivity measures between two segments defined in [11] and [12], but we differ in how these measures are used for text segmentation. Both [11] and [12] use dynamic programming to enumerate all possible partitions for a different number of segments and then apply an interconnectivity measure to select the best possible segment partition.

In [11], the prior information about the probability of segmentation  $S$ , which is modeled as  $P(S) = n^{-m}$ , helps get a reasonable number of segments. Given the variations among human editors about the level of details and preferences for the underlying hierarchical organizations of segments, it is difficult to see how such a general factor can be optimized for the segmentation process.

Although [12] considers many factors for text segmentation such as within-segment similarities and between-segment similarities along with segment lengths and sentence distances, it is not clear what is the rationale about the way these factors are combined in their implementation. Their cost function, where the  $\alpha$  and  $\beta$  parameters would have to be optimized on training data, is shown in equation (5) below.

$$C(S) = \alpha Sim_{within} - (1 - \alpha) Sim_{between} + \beta Ratios_{length} \quad (5)$$

In our approach, we iteratively break a portion of the text at the weakest points and record all the partitioned segments in a binary tree structure. We treat it as a post-processing step to flatten the binary tree into a broad and shallow hierarchy, which can be tuned to different levels of details and user preferences. In fact, the binary tree itself can simply be used as a visual illustration of the underlying hierarchical structure of a document, or it can be usefully applied to many other tasks such as information retrieval and text summarization.

## 4 Evaluating Hierarchical Organizations

Our new method for text segmentation produces a binary tree of all the partitioned segments along with their connectivity scores. On the other hand, a human-labeled segmentation structure is also hierarchical, typically made up of topics and subtopics. We take a two-step approach to evaluate the results of text segmentation. First, given two hierarchical organizations, we try to find the best possible match by comparing all possible linear partitions for the two hierarchical organizations. Second, we treat it as a separate step to flatten a binary tree into a suitable linear partition so that the result can be compared directly with a human-labeled structure, at either a topic- or subtopic-level.

### 4.1 Best Possible Match

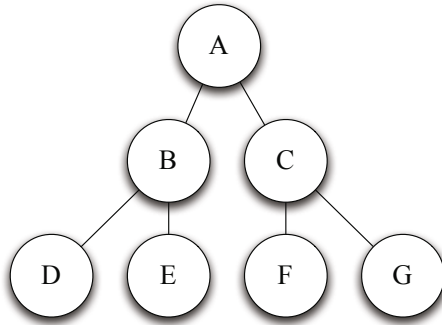
Most current systems for text segmentation are evaluated by  $Pk$  and / or *WindowDiff* measures. Both compute a degree of mismatches between two linear partitions of text.  $Pk$ , originally proposed by Beeferman et al. in [1], measures the rate of mis-matched segment boundaries via a moving window of size  $k$ , which is usually set to half of the average segment length in the reference partition. *WindowDiff* is an improved version of  $Pk$  proposed by Pevzner and Hearst in [8]. For each window of size  $k$ , *WindowDiff* not only decides whether there is a mismatch between the hypothesized partition and the reference partition, it also counts the difference of the number of segment boundaries in the given window between the two partitions. Thus, the results of *WindowDiff* are generally higher than those of  $Pk$ .

To measure the degree of mismatches between two hierarchical organizations, we propose the following method to find the best possible match between these two structures. First, we generate all linear partitions for each of the hierarchical organizations, from coarse levels to refined levels, and that cover the same portion of the text, as illustrated in Figure 2. Then, by comparing all pairs of the linear partitions from these two structures, we find the best possible match that has the lowest  $Pk$  or *WindowDiff* value between the corresponding linear partitions.

The best possible match represents the ideal case where a machine-generated partition matches a human labeled partition. It establishes an upper-bound for the results that could be achieved given a perfect converter from the binary tree representation to a flatly segmented document. In addition to providing us with a “best-case” view of our segmenter for a given human-labeling of a given dataset, this also allows us to measure how consistent the human labeled organizations are for a given document as compared to their statistical interconnectivity.

### 4.2 Flattened Linear Partition Match

Although our result of a binary tree captures most partitioned segments at different levels of details, many applications such as information retrieval and text summarization may require topic-level and / or subtopic-level partitions that are comparable to human labeled structures. We treat it as a separate step to flatten a deep and narrow tree into a broad and shallow hierarchy. Our



*Possible partitions:*  
 $[A], [B, C], [B, F, G], [D, E, C], [D, E, F, G]$

**Fig. 2.** All Possible Partitions for a Hierarchical Structure

example flattener traverses a binary tree, such as the one shown in Figure 1, in the depth-first order and examines the scores at each node. If the score at a particular node is below a pre-determined threshold, the flattener cuts the tree at that point and creates a segment for the left branch and continues the search for the right branch. Here, the score represents the interconnectivity strength at a split point for two segments at the children level and can be computed by either equation (2) or (4). With a threshold of 0.009, for example, the tree in Figure 1 will be flattened into two segments: sentences from 0 to 53 and sentences from 54 to 174.

To find appropriate threshold values for the flattening step, we use a training set of documents with human labeled segment structures and estimate the threshold values for topic and subtopic partitions, respectively. For flattening, we will generate topic partitions first, and then for each segment, we will continue generating subtopic partitions if needed.

By treating it as a separate step for flattening a binary tree into a flattened linear partition, we can adapt our results to different levels of details (either topics or subtopics or both) and user preferences (some may be detailed while others may be general).

## 5 Experimental Results

To demonstrate the effectiveness of our iterative method for text segmentation, we conducted experiments on a dataset of real-world magazine articles and compared our results with two existing methods for text segmentation.

### 5.1 Discover Magazine Dataset

For our experiments, we used a corpus of 265 articles collected from *Discover* magazine between the years 2000 and 2009. Each document was annotated by

**Table 1.** Statistics for the editor-segmented *Discover* Magazine dataset

Average number of topics per document	4.33
Average topic length in words	355.48
Average number of subtopics per document	10.31
Average subtopic length in words	115.40
Total number of word types in the dataset	20,491

an independent human editor for both topic and subtopic boundaries. The independent editor (who was unaware of our research) was given instructions for segmenting the document where the objective was to place major topic boundaries only when a prominent topic under discussion changes to some other prominent topic. Among other directives, a topic transition was defined as a major change in the subject matter. Subtopics were defined as being used to support the discussion of a topic; subtopics can only be nested within a major topic. We chose *Discover* magazine articles because each article has a reasonable length (between 1,000 and 2,000 words generally) and there are no clearly marked sections and subsections; there is therefore a need for automatic text segmentation in order to break the articles into topics and / or subtopics.

All documents are preprocessed by removing non-alphabetic tokens such as numbers and punctuation marks. We also remove stopwords using a common stopword list. Table 1 shows the statistics of this corpus after these preprocessing steps.

## 5.2 Hierarchical Evaluation

To evaluate the performance of our text segmentation method, we compared our results with those generated by two existing methods for text segmentation: C99 based on dotplots (2) and U00 based on language models (11). We chose these two systems because they are representative of the existing text segmentation methods, and their implementations are freely available on the Internet.

There are two versions of our iterative text segmentation method: *divSeg (IC)* uses equation (2) for computing the interconnectivity between two adjacent segments, while *divSeg (LM)*, based on language models, uses equation (4). We record all the results from both the *Pk* and *WindowDiff* measures so that we can get different perspectives about the performance.

Since our method produces a binary tree of all partitioned segments and the reference structures marked by human editors contain both topics and subtopics, we began by following the methodology outlined in subsection 4.1 and compared the best possible matches between two hierarchical organizations. Although C99 and U00 produce linear partitions as output, we also followed the best possible match method with these algorithms against the reference structure, which is typically hierarchical with both topics and subtopics.

As seen in Table 2, our segmentation method exceeds C99 and U00 considerably; both *Pk* and *WindowDiff* scores are decreased by an appreciable amount. Between the two versions of our own implementation, *divSeg (LM)* out-performs *divSeg (IC)* significantly, making it the overall winner in the comparisons. While

**Table 2.** Hierarchical Evaluation Results

Algorithm	C99	U00	divSeg (IC)	divSeg (LM)
Avg Pk	.32748	.37265	.2650	.20148
Avg WD	.41979	.39896	.2803	.22014

we concede that this comparison benefits *divSeg* due to the fact that our binary tree is more amenable to a “best-case” study, these results are given as an “upper-bound” and serve simply to demonstrate the potential of our method in a text segmentation application. Furthermore, the fact that our method is more amenable to this type of study is a benefit in itself; it demonstrates the dynamic nature of the fully adaptable object that our algorithm outputs. This flexibility provides for a diverse range of flattening algorithm approaches that each highlight a distinct level and preciseness of segmentation for a user’s needs. A direct comparison of our example flattener to C99 and U00 in a flat topic-level segmentation scenario follows.

### 5.3 Topic Level Evaluation

In this subsection, we present a real-world evaluation of our system by automatically flattening a deep and narrow binary tree into a suitable topic-level partition. While the *Discover* dataset includes both topic and subtopic annotations, we concentrate on topic-level segmentation in this work since the other segmentation systems are aimed at topic level partitions.

To train our flattener for an appropriate threshold, we randomly separated our dataset into a training set of 160 documents and a testing set of 105 documents. This separation was selected randomly and the documents selected for inclusion in each set were also chosen randomly. The flattener was then tested on the training documents to help find the threshold value that achieves the best *Pk* or *WindowDiff* scores. For topic-level partitions, this particular editor, and this particular dataset, the threshold value was found to be approximately 0.008. The average *Pk* and *WindowDiff* topic-level scores obtained on the testing set for our implementation and for C99 and U00 are shown in Table 3.

**Table 3.** Topic-Level Evaluation Results

Algorithm	C99	U00	divSeg (IC)	divSeg (LM)
Avg Pk	.4846	.5190	.4312	.4013
Avg WD	.7092	.6121	.4446	.4178

While not as impressive as the scores obtained in the best possible match evaluation, these results again show the advantage of our method. For both the *Pk* and *WindowDiff* metrics, our algorithm shows substantial, statistically significant – as determined by the Student’s t-test – improvements over the C99 and U00 algorithms. With further work on a more technically advanced flattening procedure, we are confident that these scores can be improved even further.



## 6 Conclusions and Future Work

In this paper, we proposed a novel method for text segmentation that iteratively splits a portion of the text until a binary tree of all partitioned segments is fully built. As illustrated in subsection 3.2, such a process is both easy to implement and efficient to compute. We followed a separate step to flatten the binary tree into a broad and shallow hierarchy in order to model human-labeled segment structures. Our experiments on the documents in the *Discover* dataset showed that the new iterative approach has the potential to generate much better segmentation results and the separate flattening step gives us the flexibility to adapt our results to different levels of details and user preferences in the final segment structures.

There are at least two major directions we can potentially take our method for text segmentation. First, it will be desirable to generalize the  $Pk$  and  $WindowDiff$  measures for determining the degrees of mismatches between hierarchical organizations, since the best possible match only establishes an upper-bound estimate when doing hierarchical experiments. Next, we can explore different ways of flattening a binary tree into a broad and shallow hierarchy so that we can better model the human labeled structures with different levels of details and user preferences. Finally, we will extend our testing to other domains and datasets such as the Wikipedia corpus which should allow for straightforward testing due to its pre-segmented format.

**Acknowledgements.** The authors would like to acknowledge the financial support from Ontario Centres of Excellence (OCE) through the OCE/Precarn Alliance Program. We would also like to thank the editor Juliette Zhang for labeling the topic and subtopic structures for the *Discover* magazine dataset.

## References

1. Beeferman, D., Berger, A., Lafferty, J.: Statistical models for text segmentation. *Mach. Learn.* 34(1-3), 177–210 (1999)
2. Choi, F.Y.Y.: Advances in domain independent linear text segmentation. In: Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference, pp. 26–33. Morgan Kaufmann Publishers, San Francisco (2000)
3. Cieri, C., Graff, D., Liberman, M., Martey, N., Strassel, S.: Large, multilingual, broadcast news corpora for cooperative research in topic detection and tracking: The tdt-2 and tdt-3 corpus efforts. In: Proceedings of Language Resources and Evaluation Conference (2000)
4. Eisenstein, J.: Hierarchical text segmentation from multi-scale lexical cohesion. In: NAACL 2009: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 353–361. Association for Computational Linguistics, Morristown (2009)

5. Eisenstein, J., Barzilay, R.: Bayesian unsupervised topic segmentation. In: EMNLP 2008: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 334–343. Association for Computational Linguistics, Morristown (2008)
6. Halliday, M.A.K., Hasan, R.: *Cohesion in English* (English Language). Longman Pub. Group, Harlow (1976)
7. Hearst, M.A.: Multi-paragraph segmentation of expository text. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, pp. 9–16. Association for Computational Linguistics, Morristown (1994)
8. Pevzner, L., Hearst, M.A.: A critique and improvement of an evaluation metric for text segmentation. *Comput. Linguist.* 28(1), 19–36 (2002)
9. Reynar, J.C.: *Topic Segmentation: Algorithms and Applications*. PhD thesis, University of Pennsylvania (1998)
10. Skorochod'ko, E.F.: Adaptive method of automatic abstracting and indexing. In: Proceedings of the IFIP, vol. 71, pp. 1179–1182 (1972)
11. Utiyama, M., Isahara, H.: A statistical model for domain-independent text segmentation. In: ACL 2001: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, pp. 499–506. Association for Computational Linguistics, Morristown (2001)
12. Ye, N., Zhu, J., Zheng, Y., Ma, M.Y., Wang, H., Zhang, B.: A dynamic programming model for text segmentation based on min-max similarity. In: Li, H., Liu, T., Ma, W.-Y., Sakai, T., Wong, K.-F., Zhou, G. (eds.) AIRS 2008. LNCS, vol. 4993, pp. 141–152. Springer, Heidelberg (2008)

# Improving Query Focused Summarization Using Look-Ahead Strategy

Rama Badrinath, Suresh Venkatasubramanian, and C.E. Veni Madhavan

Department of Computer Science and Automation,  
Indian Institute of Science, Bangalore, India  
{ramab,vsuresh,cevm}@csa.iisc.ernet.in

**Abstract.** Query focused summarization is the task of producing a compressed text of original set of documents based on a query. Documents can be viewed as graph with sentences as nodes and edges can be added based on sentence similarity. Graph based ranking algorithms which use ‘Biased random surfer model’ like topic-sensitive LexRank have been successfully applied to query focused summarization. In these algorithms, random walk will be biased towards the sentences which contain query relevant words. Specifically, it is assumed that random surfer knows the query relevance score of the sentence to where he jumps. However, neighbourhood information of the sentence to where he jumps is completely ignored. In this paper, we propose look-ahead version of topic-sensitive LexRank. We assume that random surfer not only knows the query relevance of the sentence to where he jumps but he can also look N-step ahead from that sentence to find query relevance scores of future set of sentences. Using this look ahead information, we figure out the sentences which are indirectly related to the query by looking at number of hops to reach a sentence which has query relevant words. Then we make the random walk biased towards even to the indirect query relevant sentences along with the sentences which have query relevant words. Experimental results show 20.2% increase in ROUGE-2 score compared to topic-sensitive LexRank on DUC 2007 data set. Further, our system outperforms best systems in DUC 2006 and results are comparable to state of the art systems.

**Keywords:** Topic Sensitive LexRank, Look-ahead, Biased random walk.

## 1 Introduction

Text summarization is the process of condensing a source text into a shorter version preserving its information content [1]. Generic multi-document summarization aims at producing summary from a set of documents which are based on the same topic. Query focused summarization is a particular kind of multi-document summarization where the task is to create a summary which can answer the information need expressed in the query [2]. After the introduction of query focused summarization as the main task in DUC [3] competitions, it has

<sup>1</sup> <http://duc.nist.gov>

become one of the important fields of research in both natural language processing and information retrieval.

In this paper, we concentrate on sentence extractive summarization which basically involves heuristic ranking of sentences present in the documents, and picking top-ranked sentences to the summary. Query focused summarization is harder task than generic multi-document summarization because it expects the summary to be biased towards the query. Usually, queries are complex and not direct questions. So the the summary generated by just picking the textual units which contain names or numbers would not suffice. Therefore it requires deep understanding of the documents to create an ideal summary. Further, query will have very few words. So the main challenge is to use this little information to pick important sentences which answer the question in the query.

Several methods have been developed for query focused summarization over the years. Graph based summarization methods based on Google's PageRank algorithm [3] have gained much attention due to their simplicity, unsupervised nature and language independence. For example, Lexrank [4] builds a weighted undirected graph by considering sentences as nodes and edges are added between the sentences based on cosine similarity measure. Then PageRank algorithm is applied to find salient sentences in the graph. Otterbacher et al. [5] proposed the idea of biased random walk called topic-sensitive LexRank for question answering. However, topic-sensitive LexRank can be easily extended to query focused summarization. Wan et al. [6] came up with an improved method. In their algorithm, instead of treating all relations between the sentences equally, within-document relationships and cross-document relations are differentiated and separate random walk models are used.

In the existing 'biased surfer models', sentences which contain query relevant words are given high scores by making the random walk biased towards these sentences. This is based on the assumption that random surfer knows the query relevance score of the sentence to where he jumps. However, the sentences which are indirectly related to the query are found during the course of the algorithm using the link structure of the similarity graph.

In our model, we try to find out sentences which are indirectly related to the query using the neighbourhood information. Specifically, we assume that random surfer not only knows the query relevance score of the sentence to where he jumps but we also include the option of looking N-step ahead from that sentence to learn more about it. Now we bias the random walk towards both indirect query relevant sentences and the ones which contain query relevant words. This results in generating better quality summaries.

The experiments on DUC 2006 and DUC 2007 data sets confirm that inclusion of look-ahead strategy yields better performance. We show that our method performs better than some of the recently developed methods and results are comparable to state of the art approaches.

The rest of the paper is organized as follows: In Section 2, we give a brief description of topic-sensitive LexRank and then we introduce our model. In

Section 3, we present experiments and results. Finally, we conclude and suggest possible directions for future research in Section 4.

## 2 Topic-Sensitive LexRank: A Revisit

Since we develop our model from topic-sensitive LexRank, we give a brief description of it in this section. Topic-sensitive LexRank uses concept of graph based centrality to rank the sentences. It consists of the following steps.

A similarity graph  $G$  is constructed using the sentences in the document set. Each sentence is viewed as a node. Stopwords are removed from both query and the sentences. Next, all the words are reduced to their root form through stemming and word ISF's (Inverse Sentence Frequency) are calculated by the following formula:

$$isf_w = \log\left(\frac{N_s + 1}{0.5 + sf_w}\right) \tag{1}$$

where,  $N_s$  is the total number of sentences in the cluster,  $sf_w$  is the number of sentences that the word  $w$  appears in.

Now, relevance of a sentence  $s$  to the query  $q$  is computed by the following formula:

$$rel(s|q) = \sum_{w \in q} \log(tf_{w,s} + 1) \times \log(tf_{w,q} + 1) \times isf_w \tag{2}$$

where  $tf_{w,s}$  and  $tf_{w,q}$  are the number of times  $w$  appears in  $s$  and  $q$ , respectively. Similarity between two sentences is calculated using cosine measure weighted by word ISF's.

$$sim(x, y) = \frac{\sum_{w \in x, y} tf_{w,x} tf_{w,y} (isf_w)^2}{\sqrt{\sum_{x_i \in x} (tf_{x_i,x} isf_{x_i})^2} \times \sqrt{\sum_{y_i \in y} (tf_{y_i,y} isf_{y_i})^2}} \tag{3}$$

The main aim of query focused summarization is to pick sentences which are relevant to the query. Therefore sentences which are similar to the query should get high score. But a sentence that is similar to other high scoring sentence in the graph should also get a high score. This is modelled by using a mixture model. Considering the entire graph, if  $p(s|q)$  denotes score of sentence  $s$  given query  $q$ , is determined as the sum of its relevance to query and the similarity to other sentences in the document cluster.

$$p(s|q) = d \frac{rel(s|q)}{\sum_{z \in C} rel(z|q)} + (1 - d) \sum_{v \in C} \frac{sim(v, s)}{\sum_{z \in C} sim(v, z)} p(v|q) \tag{4}$$

$d$  is known the as the damping factor which is a trade-off between the similarity of a sentence to the query and to the other sentences in the cluster.

Equation 4 can be explained using random walk model as follows. If we imagine a random surfer jumping from one node to another on the graph, at each step, the random surfer does one of the two things - with probability  $d$ , he randomly jumps to a sentence (random jump) with a probability proportional to its

relevance to the query; or with probability  $1 - d$  he follows an outlink to reach one of the neighbouring nodes (forward jump) with a probability proportional to the edge weight in the graph. Since we want to give high score to sentences which are similar to the query, usually  $d > 1 - d$  in Equation 4. Experimentally it is proved that  $d = 0.7$  gives best results [7].

Now we will introduce few key terms which help us to understand the topic-sensitive LexRank better.

**Direct query relevant sentence:** A sentence which has got at least one word overlapping with the query.

**Indirect query relevant sentence:** A sentence which does not have any query related words but it is in the vicinity of ‘direct query relevant’ sentences in the graph.

**N-step indirect query relevant sentence:** An ‘indirect query relevant’ sentence with at least one of the sentences which are N-hops away from the current sentence is ‘direct query relevant’ and further, none of the sentences which are at a distance less than N-hops are ‘direct query relevant’.

We assume that the similarity graph contains at least one ‘direct query relevant’ sentence. Topic-sensitive LexRank uses biased random walk to boost the scores of both direct and ‘indirect query relevant’ sentences in a single equation. In case of random jump, ‘direct query relevant’ sentences are preferred as random surfer knows the query relevance of the sentence to where he jumps. Therefore random jump boosts the score of only ‘direct query relevant’ sentences. Scores of ‘indirect query relevant’ sentences will not get affected as they have zero similarity with the query. On the other hand, forward jump is used to increase the scores based on sentence similarity. Basically through forward jump, sentences which are adjacent to other high scoring sentences end up getting high score too. Therefore, it helps ‘indirect query relevant’ sentences as they are very near to other high scoring sentences.

At the starting of the algorithm, set of ‘direct query relevant’ sentences is known. So the random surfer is purposely made to jump to those sentences in every random jump to increase their score. Whereas the set of ‘indirect query relevant’ sentences is not known. Therefore they depend on the forward jump of the random surfer from neighbouring high scored sentences to boost their score.

## 2.1 Topic-Sensitive LexRank with Look-Ahead Strategy

In our method we mainly concentrate on detecting ‘indirect query relevant’ sentences so that random surfer is made to choose both ‘direct query relevant’ and ‘indirect query relevant’ sentences during random jump.

For the analysis purpose, lets concentrate on ‘1-step indirect query relevant’ sentences i.e. sentences which have at least one ‘direct query relevant’ sentence as their neighbour. If we know the set of ‘1-step indirect query relevant’ sentences

before hand just like ‘direct query relevant’ sentences, then we can make the random surfer to jump to even ‘1-step indirect query relevant’ sentences during random jump. The advantage of this method is that, random jump which is happening in “every move of the random surfer”, will now prefer these sentences too. So ‘1-step indirect query relevant’ sentences need not have to wait for the forward jumps to boost their scores.

‘Direct query relevant’ sentences can be easily detected, as they will have non zero similarity with the query. But in order to detect ‘1-step indirect query relevant’ sentences, we have to make use of query relevance scores of the neighbours. Specifically, sum of query relevance scores of the neighbours will be non zero for ‘1-step indirect query relevant’ sentences.

Therefore to make the random surfer to jump to both ‘direct query relevant’ and ‘1-step indirect query relevant’ sentences in every random jump, we will define the modified query relevance function as follows

$$rel'(s|q) = \alpha \times rel(s|q) + \beta \times \sum_{s_i \in Ne(s,1)} rel(s_i|q) \tag{5}$$

where,  $Ne(s, k)$  returns sentences which are k-hop distant from  $s$ .

$\alpha$  and  $\beta$  are the parameters used to control the probability of random surfer jumping to ‘direct query relevant’ sentences and ‘1-step indirect query relevant’ sentences respectively.

In Equation 4, if we use the modified query relevance function defined in Equation 5, then resultant model can be viewed as topic-sensitive LexRank with “1-step look-ahead”. So now the random surfer not only knows the query relevance score of the sentence to where he jumps, but he also knows the query relevance score of its neighbours. Since ‘direct query relevant’ sentences are more important than ‘1-step indirect query relevant’ sentences, usually  $\alpha > \beta$ . Note that in Equation 5, ‘direct query relevant’ sentences can get additional advantage from second term as their neighbouring sentences could be other ‘direct query relevant’ sentences. So  $\alpha$  and  $\beta$  must be carefully chosen.

Generalizing this, for ‘N-step indirect query relevant’ sentence, sum of query relevance scores of sentences which are N-hops away from the current sentence will be non zero. So if we use the look ahead information, we can judge a sentence better in the sense that we can detect whether it is 1-step or 2-step or in general ‘N-step indirect query relevant’ sentence. Now with “N-level look ahead information”, we can make the random surfer to jump to both ‘direct query relevant’ sentences and all “K-step indirect query relevant sentences” in every random jump, where  $1 \leq K \leq N$ .

So topic-sensitive LexRank with “N-step look ahead” makes use of modified query relevance function which is defined as follows.

$$rel'(s|q) = \alpha \times rel(s|q) + \beta \times \sum_{s_i \in Ne(s,1)} rel(s_i|q) + \dots + \nu \times \sum_{s_i \in Ne(s,N)} rel(s_i|q) \tag{6}$$

where,  $\alpha \beta \dots \nu$  are the controlling parameters.

Note that in Equation 6, a ‘direct query relevant’ sentence can get additional advantage from rest of the N terms if it has other ‘direct query relevant’ sentences

in any of the  $N$  future levels. Similarly, a ‘ $K$ -step indirect query relevant’ sentence can get additional advantage from the rest  $(N-K)$  terms.

Finally, in Equation 4 we use the modified query relevance function defined in Equation 6 and we continue with biased random walk to rank the sentences. Theoretically, our model should work for any value of  $N$ . But since the sentences at different levels are not “conceptually” related (as we only look at word overlap to add edges), for higher values of  $N$  our model breaks down.

## 2.2 Redundancy Removal

Redundancy is a major problem in case of multi-document summarization as sentences from different documents may have similar content. Therefore it is essential to improve the diversity of the focused summary in order to increase the information coverage. In our model, we make use of the greedy algorithm proposed in 8 to impose diversity penalty on the sentences. The algorithm is as follows.

1. Define two sets,  $A = \phi$  and  $B = \{s_i | i=1,2..N\}$ , and initialize the score of each sentence to its graph based ranking score computed using Equation 4. i.e.  $Score(s_i) = p(s_i|q)$ ,  $i = 1,2..N$ .
2. Sort the sentences in  $B$  by their current scores in descending order.
3. Suppose  $s_i$  is the highest ranked sentence in  $B$ . Move  $s_i$  from  $B$  to  $A$ , and recalculate the scores of the remaining sentences in  $B$  by imposing redundancy penalty as follows. For each sentence  $s_j \in B$

$$Score(s_j) = Score(s_j) - \lambda \cdot sim(s_i, s_j) \cdot p(s_i|q) \quad (7)$$

where,  $\lambda$  is the penalty degree factor which is used to control penalty imposed on sentences.

$sim(s_i, s_j)$  is the similarity function defined in equation 3.

4. Go to step 2 and iterate until  $B = \phi$  or the iteration count reaches a predefined maximum number.

Finally, sentences in the set  $A$  are added to the summary in the same order.

## 3 Experiment and Evaluation

### 3.1 Experiment Setup

We conducted experiments on DUC 2006 and 2007 data sets. Query (Topic) focused summarization was the only task in DUC 2006 and it was the main task in DUC 2007. In DUC 2006 data set we had 50 document clusters and in DUC 2007 data set we had 45 document clusters. Each document cluster has 25 documents picked from AQUAINT corpus. Further, each document cluster contains a topic and 4 human generated summaries. The task is to synthesize a fluent, well-organized 250-word summary of the documents that answers the



need for information expressed in the topic. We combined the title and the topic for each document set and conducted the experiemnts.

We use ROUGE toolkit<sup>2</sup>(Recall Oriented Understudy for Gisting Evaluation) for evaluation. ROUGE compares system generated summary against human generated summaries to measure its quality. ROUGE gives a variety of different statistics:

- ROUGE-N(1-4): N-gram based co-occurrence statistics
- ROUGE-L: LCS (Longest Common Subsequence) based statistics
- ROUGE-W: Weighted-LCS based statistics
- ROUGE-SU: Skip-bigram plus unigram based co-occurrence statistics

In this paper, we show the results in terms of ROUGE-2 and ROUGE-SU4 scores which were considered as main criteria to rate the summaries under automatic evaluation in DUC 2007. For simplicity, we discuss the results of the proposed method with “1-step look-ahead” approach. So we use modified query relevance function defined in Equation 5 and hence final equation for the ranking process is as follows.

$$p(s|q) = d \left( \frac{\alpha \times rel(s|q) + \beta \times \sum_{s_i \in Ne(s,1)} rel(s_i|q)}{\sum_{z \in C} \left[ \alpha \times rel(z|q) + \beta \times \sum_{s_i \in Ne(z,1)} rel(s_i|q) \right]} \right) + (1-d) \sum_{v \in C} \frac{sim(v, s)}{\sum_{z \in C} sim(v, z)} p(v|q) \quad (8)$$

There are 4 parameters in our algorithm.  $d$  is the damping factor,  $\alpha$  and  $\beta$  are the weights given to ‘direct query relevant’ and ‘1-step indirect query relevant’ sentences respectively in Equation 8.  $\lambda$  represents penalty degree factor used to remove redundancy in Equation 7.

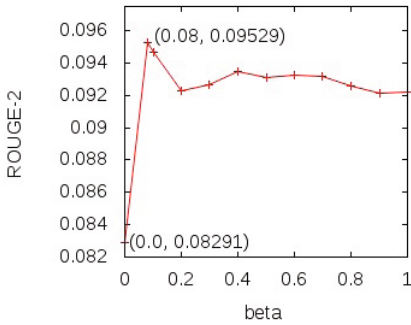
### 3.2 Experiments on DUC 2006 Data Set

In our experiments we used DUC 2006 data set to tune the parameters and then tested the performance of our model on DUC 2007 data set. From the experiments we picked the value of 0.2 for  $\lambda$  as it produced best results. We started off with  $d = 0.7$  for which topic-sensitive LexRank achieves maximum performance 7.

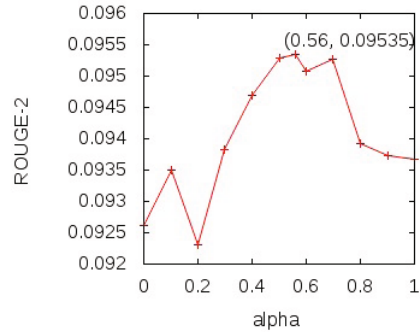
To estimate the values of  $\alpha$  and  $\beta$ , we use gradient search strategy. In the first step, we set  $\alpha$  to a value and  $\beta$  is varied within a range to observe the variation in performance. Next, we set  $\beta$  to the value for which we got best results. To find the appropriate value of  $\alpha$ , the experiment is repeated again with  $\alpha$  varying within a range.

In our experiments we first keep  $\alpha$  value fixed to 0.5 and  $\beta$  is varied from 0 to 1. Fig 1 demonstrates the influence of  $\beta$  on the performance of the model. Since

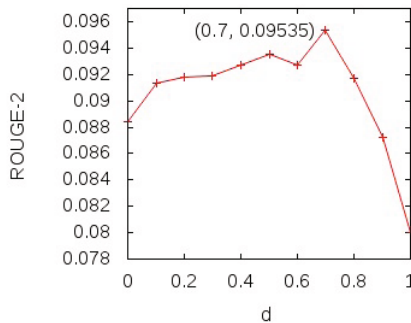
<sup>2</sup> ROUGE 1.5.5 is used, and the parameters are -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -l 250.



**Fig. 1.** ROUGE-2 vs.  $\beta$



**Fig. 2.** ROUGE-2 vs.  $\alpha$



**Fig. 3.** ROUGE-2 vs.  $d$

$\beta > 0$  indicates the addition of “look ahead information”, we can clearly see an abrupt increase in ROUGE-2 from 0.08291 to 0.09529 before and after the addition of “look ahead information” respectively. We can conclude that when  $\beta = 0.08$ , ROUGE-2 score reaches maximum and thereafter it decreases.

Drop in the performance with increase in  $\beta$  can be explained as follows. Second term in Equation 5 is the summation of query relevance scores of all the neighbours. This is quite a big value compared to query relevance score of the current sentence alone. So net effect of increase in  $\beta$  after 0.08 is that, second term completely masks the effect of first term. i.e. we are completely neglecting query relevance score of the current sentence. So the random surfer will not be able to differentiate between ‘direct query relevant’ sentences and other sentences. Because of this loss of information, performance decreases.

To find the appropriate value of  $\alpha$ , we set  $\beta$  to 0.08 and we repeat the experiment with  $\alpha$  varying from 0 to 1. Fig 2 shows the performance with different values of  $\alpha$ . Best ROUGE-2 score (0.09535) is obtained when  $\alpha$  reaches 0.56 which confirms our assumption that  $\alpha > \beta$ .

In order to test the effect of damping factor  $d$  on the ranking process, we repeated the experiment with  $d$  varying from 0 to 1 and keeping  $\alpha = 0.56$  and

$\beta = 0.08$ . From Fig 3, we can conclude that look-ahead version of topic-sensitive LexRank also achieves maximum performance at  $d = 0.7$ .

Now with the setting  $\lambda = 0.2$ ,  $d = 0.7$ ,  $\alpha = 0.56$  and  $\beta = 0.08$  we tested the performance of our model on DUC 2007 data set. We got ROUGE-2 score of 0.11983 and ROUGE-SU4 score of 0.17256.

### 3.3 Comparison with DUC Systems

Tables 1 and 2 show the comparison of our model with top 5 performing systems in DUC 2006 and DUC 2007 respectively. Our model is denoted by “1-step T-LR” which stands for “Topic sensitive LexRank with 1-step look-ahead”. In both the tables, scores are arranged in decreasing order of ROUGE-2. Last row of each table indicates baseline summaries which were created automatically for each document set by taking all leading sentences from the most recent document until a word length of 250 was reached.

**Table 1.** Comparison with DUC 2006 top 5 systems

Systems	R-2	R-SU4
<b>1-step T-LR</b>	<b>0.09535</b>	<b>0.15134</b>
S24	0.09505	0.15464
S12	0.08987	0.14755
S23	0.08792	0.14486
S8	0.08707	0.14134
S28	0.08700	0.14522
Baseline	0.04947	0.09788

**Table 2.** Comparison with DUC 2007 top 5 systems

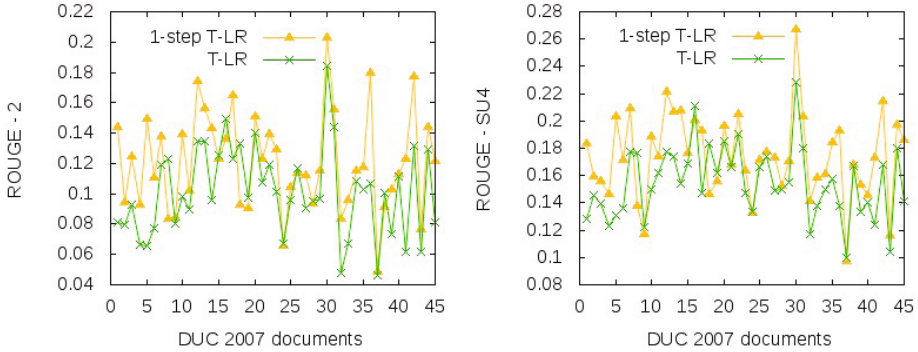
Systems	R-2	R-SU4
S15 (IIIT-H)	0.12448	0.17711
S29 (PYTHY)	0.12028	0.17074
<b>1-step T-LR</b>	<b>0.11983</b>	<b>0.17256</b>
S4	0.11887	0.16999
S24	0.11793	0.17593
S13	0.11172	0.16446
Baseline	0.06039	0.10507

In DUC 2006, the proposed approach is able to outperform all the top performing systems and stands at first position in ROUGE-2 score. In DUC 2007, we can see that our method stands at third position in ROUGE-2 score. System 15 (IIIT-H) [9] and System 29 (PYTHY) [10] which were positioned at the top in overall ROUGE evaluations in DUC 2007 are state of the art systems.

It should be noted that System 15 (IIIT-H) uses about a hundred of manually hand-crafted rules (which are language dependent) to reduce the sentences without losing much information. Even System 29 (PYTHY) uses certain sentence simplification heuristics. Though this technique increases the information content of the summary, this might affect the readability due to the fact that resulting sentences might be grammatically incorrect. Sentence simplification methods usually help in increasing ROUGE scores as we are removing unimportant words and hence making room for the informative ones. But this is done at the cost of generating ungrammatical sentences which are difficult to understand. To prove our point, IIIT-H system dropped to 22<sup>th</sup> position and PYTHY dropped to 21<sup>th</sup> position out of 30 submitted systems under “Grammaticality” in DUC 2007 evaluations. However, our method does not use any sentence simplification methods and hence summaries generated by our system do not suffer

**Table 3.** Comparison with existing methods on DUC 2007 data set

Systems	R-2	R-SU4	1-step T-LR Improvement
<b>1-step T-LR</b>	<b>0.11983</b>	<b>0.17256</b>	-
Adasum	0.11720	0.16920	(2.24%, 1.99%)
SVR	0.11330	0.16520	(5.76%, 4.46%)
HT	0.11100	0.16080	(7.96%, 7.31%)
Wiki	0.11048	0.16479	(8.46%, 4.72%)
<b>T-LR</b>	<b>0.09972</b>	<b>0.15300</b>	<b>(20.17%, 12.78%)</b>

**Fig. 4.** Per-topic comparison of Topic-sensitive LexRank (T-LR) with our system (1-step T-LR)

from grammaticality issues. Moreover, our method is able to produce informative summaries which are as good as the ones produced by state of the art systems which is evident from ROUGE scores.

### 3.4 Comparison with Existing Methods

In this section, we will compare the performance of our model with some of the recently developed systems. Description of the systems are as follows.

**Adasum** [11]: Employs a mutual boosting process to generate extractive summaries and optimize topic representation.

**SVR** [12]: Uses Support Vector Regression (SVR) to estimate the importance of sentences through set of pre-defined features.

**HT** [13]: Builds a hierarchical tree representation of the words present in the document set. A bottom-up algorithm is used to find significance of words and then sentences are picked using a top down algorithm applied on the tree.

**Wiki** [14]: Uses wikipedia as a source of knowledge to expand the query.

**T-LR** [5]: Topic-sensitive LexRank.

In Table 3 we can see that our system performs better than the recently published methods. Further, our model shows 20.17% improvement in ROUGE-2 score compared to topic-sensitive LexRank on DUC 2007 data set.

Fig 4 shows the per-topic comparison of Topic-sensitive LexRank (T-LR) with our model (1-step T-LR) in ROUGE-2 and ROUGE-4 scores on DUC 2007 data set. We can see that our method has performed well in almost all the document sets.

## 4 Conclusion and Future Work

In this paper, we present look ahead version of the topic-sensitive LexRank. Essentially we use look ahead strategy to find ‘indirect query relevant’ sentences and then we bias the random walk towards both ‘direct query relevant’ and ‘indirect query relevant’ sentences. Experimental results on DUC 2006 and DUC 2007 data sets confirms the idea of the proposed work and shows that performance of our model is comparable to state of the art approaches. Further, our model preserves linguistic quality of the generated summary unlike state of the art approaches. Our method does not depend on any language specific features and achieves good results without taking help of any external resources like WordNet/Wikipedia. We do not include any pre-processing steps like POS tagging or parsing of sentences which may consume time.

In future, we plan to extend our model to generic multi-document summarization. In query focused summarization we exactly know that sentences which are biased towards the query are potential candidates for the summary. But in generic multi-document summarization, we have to exploit the natural topic distribution in the documents to find out important sentences. So the main challenge is to figure out how to incorporate look-ahead strategy in this framework.

**Acknowledgement.** We acknowledge a partial support for the work, from a project approved by the Department of Science and Technology, Government of India.

## References

1. Barzilay, R., Elhadad, M.: Using lexical chains for text summarization. In: Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization, pp. 10–17 (1997)
2. Zhao, L., Wu, L., Huang, X.: Using query expansion in graph-based approach for query-focused multi-document summarization. *Inf. Process. Manage.* 45(1), 35–41 (2009)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* 30(1-7), 107–117 (1998)
4. Radev, D.R.: Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* 22 (2004)
5. Otterbacher, J., Erkan, G., Radev, D.R.: Using random walks for question-focused sentence retrieval. In: HLT 2005: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp. 915–922. Association for Computational Linguistics, Morristown (2005)

6. Wan, X., Yang, J., Xiao, J.: Using cross-document random walks for topic-focused multi-document. In: WI 2006: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 1012–1018. IEEE Computer Society, Washington, DC (2006)
7. Erkan, G.: Using biased random walks for focused summarization. In: Proceedings of the DUC 2006 Document Understanding Workshop, Brooklyn, NY, USA (2006)
8. Wan, X., Yang, J., Xiao, J.: Manifold-ranking based topic-focused multi-document summarization. In: IJCAI 2007: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 2903–2908. Morgan Kaufmann Publishers Inc., San Francisco (2007)
9. Pingali, P., Rahul, K., Varma, V.: Iiit hyderabad at duc 2007. In: Proceedings of the Document Understanding Conference. NIST, Rochester (2007)
10. Toutanova, K., Brockett, C., Gamon, M., Jagarlamudi, J., Suzuki, H., Vanderwende, L.: The pythy summarization system: Microsoft research at duc2007. In: DUC 2007: Document Understanding Conference, Rochester, NY, USA (2007)
11. Zhang, J., Cheng, X., Wu, G., Xu, H.: Adasum: an adaptive model for summarization. In: CIKM 2008: Proceeding of the 17th ACM Conference on Information and Knowledge Management, pp. 901–910. ACM, New York (2008)
12. Ouyang, Y., Li, W., Li, S., Lu, Q.: Applying regression models to query-focused multi-document summarization. *Inf. Process. Manage* (2010)
13. Ouyang, Y., Li, W., Lu, Q.: An integrated multi-document summarization approach based on word hierarchical representation. In: ACL-IJCNLP 2009: Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pp. 113–116. Association for Computational Linguistics, Morristown (2009)
14. Nastase, V.: Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation. In: EMNLP 2008: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 763–772. Association for Computational Linguistics, Morristown (2008)

# A Generalized Method for Word Sense Disambiguation Based on Wikipedia

Chenliang Li, Aixin Sun, and Anwitaman Datta

School of Computer Engineering,  
Nanyang Technological University, Singapore  
{lich0020,axsun,anwitaman}@ntu.edu.sg

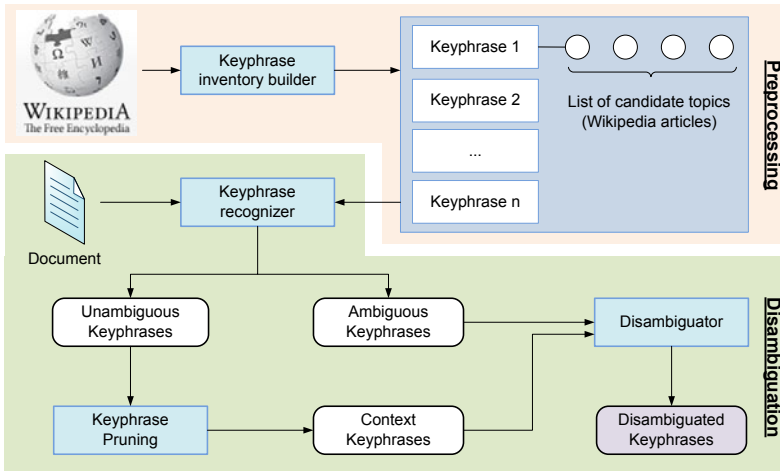
**Abstract.** In this paper we propose a general framework for word sense disambiguation using knowledge latent in Wikipedia. Specifically, we exploit the rich and growing Wikipedia corpus in order to achieve a large and robust knowledge repository consisting of keyphrases and their associated candidate topics. Keyphrases are mainly derived from Wikipedia article titles and anchor texts associated with wikilinks. The disambiguation of a given keyphrase is based on both the commonness of a candidate topic and the context-dependent relatedness where unnecessary (and potentially noisy) context information is pruned. With extensive experimental evaluations using different relatedness measures, we show that the proposed technique achieved comparable disambiguation accuracies with respect to state-of-the-art techniques, while incurring orders of magnitude less computation cost.

**Keywords:** Word Sense Disambiguation, Wikipedia, Context Pruning.

## 1 Introduction

Word sense disambiguation (WSD) is the problem of identifying the sense (meaning) of a word within a specific context. In our daily life, our brain subconsciously relates an ambiguous word to an appropriate meaning based on the context it appears. In natural language processing, word sense disambiguation is thus the task of automatically determining the meaning of a word by considering the associated context(s). It is a complicated but crucial task in many areas such as topic detection and indexing [7, 13], cross-document co-referencing [2, 18], and web people search [1, 12, 22]. Given the current explosive growth of online information and content, an efficient and high-quality disambiguation method with high scalability is of vital importance.

Two main approaches can be found in the literature that try to address the issue, namely *knowledge-based* methods and *supervised machine learning* methods. The former relies primarily on dictionaries, thesauri, or lexical knowledge bases, e.g., a sense inventory consisting of words/phrases and definitions of their possible senses. The Lesk algorithm [11] is the seminal algorithm of such kind, with the assumption that the words referring to the same meaning share a common topic in their neighborhood. Following this idea, a lot of works attempted



**Fig. 1.** The framework of keyphrase disambiguation based on Wikipedia

to identify the correct meaning for a word by maximizing the agreement between the dictionary definitions and the contextual terms of the given ambiguous word. Within the disambiguation process, a high-quality sense inventory is a critical factor that affects the performance. However, building such a large-scale, machine-readable lexical resource is tedious and laborious. Thus, the knowledge acquisition bottleneck is the main problem limiting the performance of such systems. The second method based on supervised machine learning attempts to derive a set of local and global contextual features from a manually sense-tagged dataset and to integrate these training examples into a machine learning classifier. Many machine learning techniques have been applied to WSD, and shown to be successful [6], [10], [17]. Nevertheless, machine learning methods too suffer from the knowledge acquisition bottleneck since they require substantial amounts of training examples.

In this paper, we propose a generalized method exploring the use of Wikipedia as the lexical resource for disambiguation. Wikipedia is the largest online encyclopedia and collaborative knowledge repository in the world with over 3.2M articles in English alone. It provides with a reasonably broad if not exhaustive coverage of topics, in comparison to many other knowledge bases. Previous study has found that the quality of Wikipedia articles is comparable to the editor-based encyclopedia [5]. Because of its massive scale of collaboration as well as usage, Wikipedia has become a fruitful resource in many research areas in recent years.

The proposed disambiguation framework is illustrated in Figure 1. Three key components, Wikipedia inventory, keyphrase identification and pruning, and sense disambiguator are developed in our work for disambiguation. Specifically, we build a word sense inventory by extracting the polysemy, synonym and hyperlinks encoded in Wikipedia. Each entry in the inventory is a keyphrase which refers to at least one Wikipedia article. To be detailed in Section 3.1, a keyphrase is either a Wikipedia article title, or the surface form (or anchor text) of a wikilink. Those



keyphrases, each of which refers to exactly one Wikipedia article, are unambiguous keyphrases. Some keyphrases are ambiguous; each of which refers to multiple Wikipedia articles (i.e., candidate topics/senses, shown in Figure 1). Given a document, the unambiguous keyphrases recognized from the document serve as context information to disambiguate the ambiguous keyphrases. In between, the keyphrase pruning helps identify the most important keyphrases in the context of the occurrence of the given ambiguous keyphrase for disambiguation, and it can largely filter out the noise and improve efficiency of the system. The disambiguator is the core component of our framework. It aims to balance the agreement between the context of the ambiguous keyphrase and the context of each candidate sense. Empirical evaluations based on a ground-truth dataset illustrate that our method outperforms other state-of-the-art approaches in terms of both effectiveness and efficiency. Moreover, since the Wikipedia inventory we create relies on the rich semantic information contained in Wikipedia, our approach avoids the traditional knowledge acquisition bottleneck and is applicable to any domain of varying size. It can be plugged into the existing works which require to address word sense disambiguation as well as potential applications.

Our approach is general enough in several senses: given rather exhaustive coverage of Wikipedia topics, the Wikipedia inventory is domain independent; given Wikipedia's growing popularity in other languages, our approach can be readily reused across different languages; and finally, the modular framework allows for using different relatedness measures suiting different application needs.

The rest of this paper is structured as follows: Section 2 reviews related works. Section 3 introduces our approach along with the individual components in the proposed framework. In Section 4 we present and discuss the experimental results. Finally, we conclude in Section 5.

## 2 Related Work

Many recent works explore Wikipedia to enhance text mining tasks, such as semantic relatedness measure [15, 19], text classification and clustering [4, 9, 21], and topic detection [7, 13, 14, 16]. Among these studies, we review the related works involving word sense disambiguation and semantic relatedness measures.

Strube and Ponzetto used Wikipedia for measuring semantic relatedness [19]. Their method searches the Wikipedia articles that contain the specific word in their titles, and measure the relatedness by taking the path length measure in the Wikipedia category hierarchy, text overlap, as well as their probability of occurrence. Milne and Witten developed a light-weight measure of semantic relatedness based on the Wikipedia links, called Wikipedia Link-based Measure (WLM) [15]. First, they identified the Wikipedia articles that related to the term; then, they compute the relatedness of two terms by their mapped Wikipedia articles as follow:

$$relatedness(a, b) = \frac{\log(\max(|A|, |B|)) - \log(|A \cap B|)}{\log(|W|) - \log(\min(|A|, |B|))} \quad (1)$$

where  $a$  and  $b$  are the two Wikipedia articles,  $A$  and  $B$  are the sets of all Wikipedia articles that link to  $a$  and  $b$  respectively, and  $W$  is the set of the entire Wikipedia articles. Due to its high accuracy and low cost, it is commonly used in existing works [8], [13], [16]. In our work, we employ *WLM* as an option to calculate the semantic relatedness between two Wikipedia articles efficiently. Since this method focuses on the hyperlinks within Wikipedia articles, we also investigate Dice and Jaccard measures over the hyperlinks for disambiguation in our study.

Wikify! [14] tries to annotate keyphrases in a document with Wikipedia topics where keyphrase disambiguation is a key step. Both knowledge-based and data-driven algorithms were used in Wikify!. The knowledge-based method, inspired by the Lesk algorithm [11], utilizes the occurrences of ambiguous keyphrases and the contextual information. However, the standalone method performed worse than the baseline method using the most common sense. The data-driven method learns classifier with a number of features, such as part-of-speech and the local contextual words. They then combined the two algorithms by using voting scheme. Most significantly, the method is computationally expensive since it extracts a training feature vector for each ambiguous keyphrase from all its occurrences in the whole Wikipedia.

Medelyan *et al.* [13] utilized both relatedness and commonness measures. For a given document, all keyphrases, each of which uniquely maps to one Wikipedia topic are identified as the context. The context is used to then disambiguate the keyphrases that each can map to more than one Wikipedia topic. In their work, relatedness to the context for each candidate topic of an ambiguous keyphrase is computed by *WLM*. For a candidate topic  $t$ , *Commonness* for a given keyphrase  $k$  is the priori probability of the keyphrase  $k$  referring to the candidate topic  $t$ , i.e.,  $P(t|k)$  [14]. With the two measures, a score is computed for each candidate topic  $t$  for a given keyphrase  $k$  using the following equation.

$$Score(t, k) = \frac{\sum_{c \in C} relatedness(t, c)}{|C|} \times P(t|k) \quad (2)$$

In this equation,  $C$  denotes the context of the keyphrase  $k$ . Observe that all context keyphrases in [13] are treated equally. Evaluated on 100 Wikipedia articles, the proposed method outperformed the most common sense baseline by a significant 2.4 percent in F-measure.

Naturally, some keyphrases are more related to the context than others especially when a document covers multiple topics. Milen and Witten [16] proposed to weigh the context keyphrases based on their relatedness to each other as well as their *keyphraseness*. Specifically, if the context is cohesive, then the relatedness measure becomes more relevant; while *commonness* is more useful when the context is diverse. Their empirical study showed that C4.5 classifier achieved the better performance than Medelyan *et al.*'s approach.

While the works from Medelyan *et al.*, Milne and Witten achieve a promising performance among the existing approaches to date, they rely on the context relatedness by taking all unambiguous keyphrases identified in the given document into account, which is not efficient. As a document often contains some noise,

i.e., webpages, not all unambiguous keyphrases are equally useful for expressing the thread of the document, and some of them may even lower accuracy besides wasting computational resources. Although Milen and Witten applied a weighting scheme to highlight the more semantic related context keyphrases, it inevitably incurs additional cost. In this work, we apply a pruning scheme picking the most important keyphrases for further processing. This non-trivial step filters out shallow keyphrases and significantly reduces noise, which leads to both better efficiency as well as accuracy. Moreover, existing methods were defined and evaluated by using a specific relatedness measure. Here, we develop a generalized algorithm that can be adaptive to different relatedness measures.

### 3 Disambiguation Framework

In this section, we provide concrete description of the three core components realized in order to achieve disambiguation, namely: *Wikipedia inventory*, *keyphrase identification and pruning*, and *disambiguator*, in that sequence respectively, following the order of their usage in our framework.

#### 3.1 Wikipedia Inventory

The Wikipedia inventory consists of keyphrases and their associated candidate topics. The keyphrases are from two sources, namely, Wikipedia article titles and anchor texts of wikilinks.

In Wikipedia, each article describes a single topic and is titled using the name which is most commonly used to refer to the topic<sup>1</sup>. Hence, the titles of Wikipedia articles are included in our Wikipedia inventory as keyphrases, each of which refers to the associated Wikipedia article as its candidate topic<sup>2</sup>. Note that, Wikipedia pages for administration or maintenance purposes (e.g., discussion, talk, user pages), are excluded, but the redirect pages are included. A redirect page in Wikipedia redirects the page title to the target article with the preferred title given the two titles referring to the same topic. Such redirection can help us deal with synonym (alternative names), abbreviations, spelling variations, and misspellings. Naturally, target article of the redirection is the candidate topic for the title of a redirect page as a keyphrase in the inventory.

Based on the Wikipedia policy, wikilinks (or hyperlinks) in Wikipedia should be created to relevant topics of the article, technical terms mentioned, or for proper names that are likely to be unfamiliar to readers<sup>3</sup>. Thus, the anchor texts and the linked articles of hyperlinks are semantic associations built by the wisdom of crowd of Wikipedia contributors. Note that, anchor text is the surface form of a hyperlink which may not always match the title of the linked article. Hence the anchor texts enrich the keyphrase inventory largely by polysemy, associative relatedness and social relatedness reflected by them [8]. The anchor

<sup>1</sup> <http://en.wikipedia.org/wiki/Wikipedia:TITLE>

<sup>2</sup> From now on, we use Wikipedia article, candidate sense, sense, candidate topic, Wikipedia topic equivalent interchangeably.

<sup>3</sup> <http://en.wikipedia.org/wiki/Wikipedia:Linking>

text and its linked article is added in our Wikipedia inventory as keyphrase and its candidate topic respectively.

Wikipedia disambiguation pages are designed to disambiguate a number of similar topics which may be referred to by a single ambiguous term. The titles of such pages are normally one of the ambiguous terms, followed by the tag **disambiguation**. The candidate topics are listed in the page, each with a short description about it. We adopt the heuristic by Turdakov and Velikhov [20] to extract the candidate topics from each disambiguation page. When an ambiguous term already exists in the inventory as a keyphrase, we update its list of candidate topics with the ones extracted from corresponding disambiguation page.

In summary, Wikipedia keyphrase inventory is created by taking Wikipedia article titles, processing redirected pages, parsing disambiguation pages and extracting of hyperlinks. In the inventory, if a keyphrase is associated with exactly one topic (or article), we call it unambiguous keyphrase. An ambiguous keyphrase is associated with more than one topic.

### 3.2 Keyphrase Identification and Pruning

We parse the input document and extract all keyphrases that are also present in the inventory, with preference for longer ones. For instance, given a sentence “The Java Sea is ...”, we extract a keyphrase *java sea* instead of *java*. For the unambiguous keyphrases extracted, their associated Wikipedia topics are obtained directly from the inventory. These Wikipedia topics help us understand the topics covered by the document, and provide context to determine the sense of the ambiguous keyphrases extracted.

However, a document may cover very diverse topics. Thus, not all identified unambiguous keyphrases are equally important for disambiguation. While the related keyphrases can help identify the correct sense of an ambiguous keyphrase, the unrelated ones may hurt the disambiguation accuracy and incur additional computational cost. This calls for an appropriate pruning scheme for both effectiveness and efficiency.

We use the *keyphraseness* measure to quantify the importance of a keyphrase as in [7], [14]. For a given unambiguous keyphrase, *keyphraseness* is a priori probability that a keyphrase is used as anchor text, no matter where it appears. Based on this measure, we select the top  $M$  keyphrases with the highest *keyphraseness* values to form *context keyphrases*. The ambiguous keyphrases identified from the document are then disambiguated using the context keyphrases. In our experiments, we shall evaluate the impact of  $M$  on the effectiveness and efficiency of disambiguation.

### 3.3 Disambiguator

For a given ambiguous keyphrase  $k$ , not all context keyphrases are equally important for disambiguation as some are more semantically related to  $k$  than others. For example, keyphrase *Albert Einstein* appears in the Wikipedia article *Google Search*<sup>4</sup> as an example for the introduction of Google Doodle feature.

<sup>4</sup> [http://en.wikipedia.org/wiki/Google\\_Search](http://en.wikipedia.org/wiki/Google_Search)

Obviously there is very little relatedness (if any) between the genius in science and the search engine giant. Nevertheless, due to its high keyphraseness value, *Albert Einstein* is often selected as one of the context keyphrases.

Since each keyphrase (or one of its candidate topics) refers to one Wikipedia article, the computation of relatedness between two keyphrases (or candidate topics) can therefore be reduced to the problem of computing relatedness between their associated Wikipedia articles. A few measures have been reported in the literature to measure semantic relatedness between two Wikipedia articles, mainly based on wikilinks, such as Dice, Jaccard, and WLM [15] measures (see Section 2). As a generic framework, our proposed method can use any such measure and in our following discussion we use  $Relatedness(k, k')$  to denote the relatedness between two keyphrases  $k$  and  $k'$  (or candidate topic  $t$ ).

Recall that a document may cover many diverse topics, which is often reflected by its  $M$  context phrases. That is, some context phrases from  $M$  may not be strongly related to the other context phrases. Similar to that in [16], a context keyphrase is weighted by its relatedness to all other context keyphrases, shown in the following equation. In this equation,  $C$  denotes the set of context keyphrases and  $|C| \leq M$ .

$$Weight(k, C) = \frac{\sum_{k' \in C \setminus k} Relatedness(k, k')}{|C| - 1} \tag{3}$$

With the defined weight, the relatedness between a candidate topic  $t$  to the entire context  $C$  is computed in Equation 4. Similar contextual similarity has been adopted in [2], [3], [11], [14].

$$Relatedness(t, C) = \frac{\sum_{k \in C} Weight(k, C) \times Relatedness(t, k)}{\sum_{k \in C} Weight(k, C)} \tag{4}$$

As discussed in Section 2, *commonness* is the priori probability of a keyphrase referring to a specific topic. Existing works already show the effectiveness of commonness measure. In our framework, we balance the relatedness and commonness using an exponential factor  $c$ . Given a keyphrase  $k$  to be disambiguated, let  $C_k$  be the set of candidate topics of  $k$ . We assign topic  $t_o$  as the disambiguated topic to  $k$  which maximizes both relatedness and commonness with the pre-specified parameter  $c$ , shown in Equation 5.

$$t_o = \arg \max_{t \in C_k} (Relatedness(t, C)^c \times P(t|k)) \tag{5}$$

Thus, our framework involves two parameters:  $M$  for the size of the context, and  $c$  for balancing the relatedness and commonness. A smaller  $M$  keeps the more useful topics for disambiguation and improves the efficiency, with the risk of filtering away helpful topics as well. A larger  $M$ , on the other hand, may bring in more useful topics as well as noise, and certainly is costlier computationally. As for the scaling factor  $c$ , it gives the flexibility of adjusting the impact of relatedness measure based on various relatedness definitions (e.g., Jaccard and WLM). In the following section, we illustrate the impact of the two parameters empirically.

## 4 Experiments

We conducted two sets of experiments. In the first, we evaluate the disambiguation accuracy of the proposed technique and the impact of varying the two parameters  $M$  and  $c$  on the three types of relatedness measures, namely, *Dice*, *Jaccard* and *WLM*. In the second set of experiments, we compare the proposed technique with three state-of-the-art methods and two baseline methods. Next we report our findings.

### 4.1 Dataset and Performance Metric

We used the English Wikipedia dump released on 30 January, 2010<sup>5</sup> to build the keyphrase inventory. In this dump, there are 3,246,821 articles and 266,625,017 hyperlinks among them. The resulting inventory consists of 6,168,269 unambiguous keyphrases and 526,081 ambiguous keyphrases respectively. For the latter, each keyphrase refers to 4.22 candidate topics on an average.

All evaluated disambiguation method assigns each ambiguous keyphrase  $p$  to exactly one candidate topic  $t$ . We report the *accuracy* of the assignments, i.e., the ratio of the correct assignments for all ambiguous keyphrases involved in the evaluation<sup>6</sup>. The correct assignments are predetermined by human annotations (wikilinks which have been made collaboratively) in our experiments.

### 4.2 Evaluation of the Proposed Method

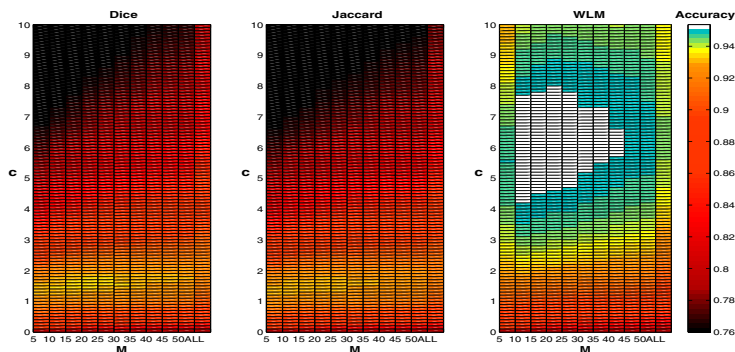
To evaluate the disambiguation accuracy of the proposed method and the impact of the parameter settings, we randomly selected 500 articles from the Wikipedia dump, such that, each selected article contained at least 50 unambiguous keyphrases. Such a selection criterion allowed us to evaluate a relatively large range of  $M$  values. Recall that our proposed method involves two parameters  $M$  and  $c$ , and a relatedness measure.  $M$  determines the number of related keyphrases involved in the computation and  $c$  balances the commonness and relatedness measure.

The selected 500 articles contained 15,298 ambiguous keyphrases in total. Figure 2 reports the disambiguation accuracy of the proposed methods by varying  $M$  and  $c$  on the three relatedness measures.  $M$  was varied from 5 to 50 with a step of 5, and *All* took all unambiguous keyphrases into account.  $c$  was varied from 0 to 10 with a step of 0.1. Note that when  $c = 0$ , our method reduces to the ‘most common sense’ scenario. We made the following observations on the experimental results.

- Parameter  $c$  significantly affected the results for all relatedness measures. For *Dice* and *Jaccard*, best accuracies were achieved when  $c = 1.5$  for a fairly large span of  $M$  values from 10 to 40. For *WLM*, the best accuracies were achieved when  $c$  was in the range of 5 to 7 and  $M$  was between 10 and 40.

<sup>5</sup> <http://download.wikimedia.org/enwiki/20100130/>.

<sup>6</sup> Note that as each ambiguous keyphrase cannot have more than one sense in a given context, the accuracy reported here is the same as both precision and recall.



**Fig. 2.** Accuracy of varying  $M$  and  $c$  with Dice, Jaccard and WLM

**Table 1.** Relatedness distribution using Dice, Jaccard and WLM

Relatedness	Mean	Std	CV
Dice	0.0158	0.0201	1.2709
Jaccard	0.0081	0.0106	1.3074
WLM	0.4174	0.1583	0.3793

- A larger  $M$  did not necessarily lead to better accuracy. In particular, accuracies dropped for all settings when  $M = All$ . This is consistent with what we have discussed earlier, that not all unambiguous keyphrases are useful for disambiguation. Many of them may bring in more noise than benefit. The other implication of obtaining high accuracy for relatively small values of  $M$  is that, even very few unambiguous keyphrases provide adequate clues for disambiguation.

To better understand the impact of  $c$  on the three relatedness measures, as a case study, we calculated the pair-wise relatedness between the Wikipedia article *Google* and all its 235 out-going neighbors, using Dice, Jaccard and WLM respectively. Table 1 reports the mean, standard deviation (std) and coefficient of variation (CV) of these 235 pair-wise relatedness. Observe that the relatedness values by Dice and Jaccard are widely scattered; while WLM generates a narrow dispersion of relatedness values. This is consistent with the previous observation that a larger  $c$  obtains a better disambiguation ability with WLM. The experimental results and the case study also illustrate that our method can generalize well for different settings.

### 4.3 Comparison with Other Methods

In this set of experiments, we compare our method with three state-of-the-art methods and two baseline methods for both effectiveness and efficiency. Specifically, we compared our method with the methods reported in Milen and Witten

**Table 2.** Statistics on datasets

Dataset	#articles	#unambiguous	#ambiguous	#candidates
Training	500	59,027	15,298	707,016
Validation	100	13,442	3,800	178,306
Evaluation	200	24,872	7,614	354,592

**Table 3.** Disambiguation accuracy and execution time on the evaluation set

Method	Accuracy(%)	Time(second)
Random sense	18.34*	14
Most common sense	78.28*	42
Medelyan <i>et al.</i>	86.07*	5,438
M&W with C4.5	86.25*	5,810
M&W with bagged C4.5	85.59*	5,877
Dice(M=5,c=1.5)	92.01*	137
Dice(M=10,c=1.5)	92.65*	265
Dice(M=15,c=1.5)	92.50*	392
Jaccard(M=5,c=1.5)	91.99*	136
Jaccard(M=10,c=1.5)	92.58*	266
Jaccard(M=15,c=1.5)	92.46*	392
WLM(M=5,c=6.0)	93.63*	140
WLM(M=10,c=6.0)	94.17	273
WLM(M=15,c=6.0)	<b>94.19</b>	399

(M&W)<sup>7</sup> [16], and Medelyan *et al.* [13]. The former builds machine learning classifiers to disambiguate the keyphrases and the latter maximizes the balance between commonness and relatedness using equal weight (See Section 2). To build the classifiers, we used C4.5 and Bagged C4.5 using Weka library<sup>8</sup>. The two baseline methods are *Random sense* and *Most common sense* which simply assign topics to ambiguous keyphrases randomly and to the most common sense respectively.

We used the dataset of 500 articles that was used in the previous section (Section 4.2) for classifier training. The trained classifiers are validated using another set of randomly selected 100 articles. For a fair comparison, all methods were evaluated on another set of 200 randomly selected articles which has no overlap with the articles used in training, validation. The statistics of the three datasets are reported in Table 2.

The disambiguation accuracy and execution time of the evaluated methods are reported in Table 3. Note that, for *Random sense*, the result is averaged over 10 runs. For the proposed method, we report the performance using 9 sets of parameter settings on relatedness measure,  $M$  and  $c$ , respectively. The parameters were set according to the findings in Section 4.2.

<sup>7</sup> Two classifiers with the best performance in their work are evaluated here: C4.5 and bagged C4.5.

<sup>8</sup> <http://www.cs.waikato.ac.nz/ml/weka/>



**Effectiveness.** Overall, the proposed method with WLM achieved the best performance among all methods. Specifically, the best accuracy 94.19% was achieved with *WLM* and  $M = 15$ ,  $c = 6.0$ . The symbol \* indicates the change is significant according to the paired *t*-test at the level of  $p < 0.001$ , compared to the best accuracy. The methods with Dice and Jaccard yield competitive accuracies. M&W with C4.5 classifier performed marginally better than Medelyan *et al.*. While classifier bagging improved the accuracy by 0.3% in [16], it degraded the performance by 0.66% in our experiments. All these methods, on the other hand, significantly outperformed the two baselines. Specifically, most common sense delivered an accuracy of 78.28%, and random guess had a mere 18.34% accuracy.

**Efficiency.** Table 3 also reports the execution time by each method evaluated, ignoring the time taken for data loading and classifier training. All experiments were conducted on the same workstation with a 2.40GHz Xeon quad-core CPU and 24GB of RAM. Observe that our method outperformed the state-of-the-art methods significantly in terms of efficiency. With  $M = 15$  and 5, our method was 14 and 40 times faster than Medelyan *et al.* and M&W, respectively. Moreover, by setting  $M$  to 5 instead of 15, the proposed method speed up 2.8 times with less than 1% of drop in accuracy, for all three relatedness settings.

## 5 Conclusion

Word sense disambiguation is an essential ingredient needed to address in many applications in the areas of Natural Language Processing, Information Retrieval and others. The large scale and high quality knowledge in Wikipedia enables a domain independent knowledge repository for word sense disambiguation. In this paper, we propose a general framework (which can accommodate diverse relatedness measures, is domain independent, and potentially can be applied for other languages) to utilize Wikipedia for word/keyphrase sense disambiguation using both commonness and relatedness measures. We show that pruning of unnecessary or potentially noisy context make the disambiguation process orders of magnitude faster than existing methods while achieving comparable (if not better) disambiguation accuracy.

**Acknowledgments.** This work is supported in part by the Agency for Science, Technology and Research (A\*STAR) SERC Grant No: 072 134 0055.

## References

1. Artiles, J., Gonzalo, J., Sekine, S.: Weps 2 evaluation campaign: overview of the web people search clustering task. In: Web People Search Evaluation Workshop (WePS), WWW Conference (2009)
2. Bagga, A., Baldwin, B.: Entity-based cross-document coreferencing using the vector space model. In: Int'l Conf. on Computational Linguistics, pp. 79–85 (1998)

3. Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: EMNLP-CoNLL, pp. 708–716 (2007)
4. Gabrilovich, E., Markovitch, S.: Overcoming the brittleness bottleneck using wikipedia: enhancing text categorization with encyclopedic knowledge. In: AAAI, pp. 1301–1306 (2006)
5. Giles, J.: Internet encyclopaedias go head to head. *Nature* 438 (December 2005)
6. GlioZZo, A., Giuliano, C., Strapparava, C.: Domain kernels for word sense disambiguation. In: ACL, pp. 403–410 (2005)
7. Grineva, M., Grinev, M., Lizorkin, D.: Extracting key terms from noisy and multi-theme documents. In: WWW, pp. 661–670 (2009)
8. Han, X., Zhao, J.: Named entity disambiguation by leveraging wikipedia semantic knowledge. In: ACM CIKM, pp. 215–224 (2009)
9. Hu, X., Zhang, X., Lu, C., Park, E.K., Zhou, X.: Exploiting wikipedia as external knowledge for document clustering. In: ACM KDD, pp. 389–396 (2009)
10. Lee, Y.K., Ng, H.T.: An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In: EMNLP, pp. 41–48 (2002)
11. Lesk, M.: Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: SIGDOC, pp. 24–26 (1986)
12. Mann, G.S., Yarowsky, D.: Unsupervised personal name disambiguation. In: HLT-NAACL, pp. 33–40 (2003)
13. Medelyan, O., Witten, I.H., Milne, D.: Topic indexing with wikipedia. In: AAAI Workshop on Wikipedia and Artificial Intelligence (2008)
14. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: ACM CIKM, pp. 233–242 (2007)
15. Milne, D., Witten, I.H.: An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: AAAI Workshop on Wikipedia and Artificial Intelligence (2008)
16. Milne, D., Witten, I.H.: Learning to link with wikipedia. In: ACM CIKM, pp. 509–518 (2008)
17. Pedersen, T.: A decision tree of bigrams is an accurate predictor of word sense. In: NAACL, pp. 1–8 (2001)
18. Ravin, Y., Kazi, Z.: Is hillary rodham clinton the president?: disambiguating names across documents. In: Workshop on Coreference and its Applications (CorefApp), pp. 9–16 (1999)
19. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: AAAI, pp. 1419–1424 (2006)
20. Turdakov, D., Velikhov, P.: Semantic relatedness metric for wikipedia concepts based on link analysis and its application to word sense disambiguation. In: *SYRCoDIS*. CEUR Workshop Proceedings, vol. 355 (2008)
21. Wang, P., Domeniconi, C.: Building semantic kernels for text classification using wikipedia. In: ACM KDD, pp. 713–721 (2008)
22. Yoshida, M., Ikeda, M., Ono, S., Sato, I., Nakagawa, H.: Person name disambiguation by bootstrapping. In: ACM SIGIR, pp. 10–17 (2010)

# Representing Document Lengths with Identifiers

Raffaele Perego, Fabrizio Silvestri, and Nicola Tonellotto

ISTI, National Research Council of Italy (CNR), Pisa, Italy

**Abstract.** The length of each indexed document is needed by most common text retrieval scoring functions to rank it with respect to the current query. For efficiency purposes information retrieval systems maintain this information in the main memory. This paper proposes a novel strategy to encode the length of each document directly in the document identifier, thus reducing main memory demand. The technique is based on a simple document identifier assignment method and a function allowing the *approximate* length of each indexed document to be computed analytically.

## 1 Introduction

Modern Information Retrieval Systems (MIRs) need to process users' queries over huge indexes both efficiently and effectively. Typically a query must be matched against an index built over the whole collection of documents that the search engine searches through. During the query processing phase, each term contained in the query is looked up in the index, and the list of all documents containing it is retrieved (a.k.a posting list). Eventually, these lists are combined and the documents within these lists are ranked by means of a scoring function measuring their relevance for the query [2]. A key issue to scale MIRs up to large document collections is the amount of memory consumed. In fact a MIRS has several sources of memory consumption: ad-hoc data structures for efficient matching techniques, software caches to allow fast access to frequent queries results and frequently used posting lists, information on document attributes for ranking purposes, etc.

In this paper we propose a novel strategy that allows memory demand to be reduced by means of a simple document identifier assignment method allowing the *approximate* length of each indexed document to be computed analytically. The knowledge of the lengths of the documents answering a given query is needed for document scoring purposes, and lengths are thus commonly cached in the main memory for fast access. Our proposal provides to encode this information directly in the document identifiers stored in the posting lists of the inverted index, thus saving the main memory used to code them explicitly.

It has been shown that assigning document identifiers by document length enhances remarkably the compressibility of the indexes [1]. In this paper, we show that this ordering can be exploited also to eliminate the space needed to store document lengths. We report the results of detailed experiments conducted with the 2009 TREC Web Track dataset showing that the small approximation error incurred by our technique has a negligible impact on retrieval effectiveness.

## 2 Document-Length Implicit Encoding

The length of a document is an important feature considered by most text retrieval scoring functions. Consider the popular BM25 ranking model [3]. Given a query  $Q$  containing terms  $q_1, \dots, q_n$ , the BM25 score for a document  $D$  in the collection is computed according to the following formula:

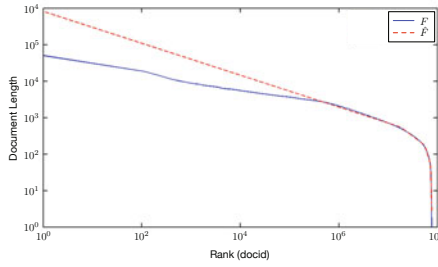
$$\text{score}_{\text{BM25}}(D, Q) = \sum_{i=1}^n \left( \text{IDF}(q_i) \frac{f(q_i, D)(k_1 + 1)}{f(q_i, D) + k_1 + (1 - b + b \frac{|D|}{\text{avgdl}})} \right)$$

where  $f(q_i, D)$  is  $q_i$ 's term frequency, i.e. the number of occurrences of  $q_i$  in  $D$ ,  $\text{IDF}(q_i)$  is the *Inverse Document Frequency* of  $q_i$  in the collection considered,  $k_1$  and  $b$  are tuning parameters (usually set to 2.0 and 0.75, respectively [3]),  $\text{avgdl}$  is the average length of the documents in the collection and  $|D|$  is the number of terms contained in  $D$ . To evaluate the above scoring function for a query and each document in the indexed collection, several data must be accessed. In particular, we have to scan the posting lists associated to all the query terms, and we need to know the length  $|D|$  of each document contained in these posting lists, to normalize the weight of a term in the current document with respect to the number of terms  $D$  contains. Since, the lengths of all scored documents have to be accessed, MIRSs resort to an ad-hoc data structure, e.g., a simple array  $\text{dl}$  of integers, that for each document  $D$ , returns its length  $\text{dl}[D]$ . For efficiency reasons, the array  $\text{dl}$  is kept in main memory, but we assert that this design strategy is far from being the best one.

Let us do a *back-of-the-envelope* calculation to estimate the amount of memory needed to store *just* this information in memory. Let us assume to manage a document collection of about  $2^{27} \sim 130M$  documents, i.e., a size that can be managed efficiently by a single modern computing server. Each document length can be stored using an unsigned integer that in modern CPUs occupies 8 Bytes. We consider, to be conservative, 4 Bytes for each integer. Therefore, the  $\text{dl}$  array requires  $2^2 \cdot 2^{27} = 2^{29} \sim 0.5$  GBytes of memory, becoming 1GB if we store integers using 64 bit integers. We can save some space if we encode  $\text{dl}$  using some sort of integer encoding method [2]. Another drawback of storing  $\text{dl}$  as an in-memory array is that elements of  $\text{dl}$  are usually scanned in the order in which the document identifiers are stored in the posting lists. Since posting lists may be huge, temporal locality may be poor, although some spatial locality may be exploited by prefetching mechanisms of modern memory hierarchies.

The above analysis motivates the main matter of this paper: a document identifier assignment strategy based on sorting documents on the basis of their lengths that has the beneficial effect of allowing the definition of a *function* which computes the *approximate* length of each indexed document.

Figure 1 shows the log-log distribution  $F$  of document lengths of CW09B. To achieve this plot the documents were sorted by decreasing order of length, and the resulting data were decimated linearly (by taking 1 value every 100 samples), and shifted by one to deal with zero-length documents. We assert that by sorting documents by decreasing order of length and assigning them identifiers



**Fig. 1.** Document length distribution  $F$  and an approximation  $\hat{F}$  using 4 segments

**Table 1.** Fitting parameters and resulting RMSE for four different approximations

n	$a_i$	$b_i$	RMSE
1	-0.92	21.38	0.52
2	-0.68, -7.05	17.64, 128.60	0.2299
3	-0.59, -2.609, -35.99	16.35, 50.78, 640.20	0.1286
4	-0.48, -1.12, -4.86, -56.30	14.76, 25.17, 90.21, 999.60	0.0695

according to their rank, we can approximate the distribution  $F$  with a function  $\hat{F}$ , that allows us to estimate the length of each document. In order to devise the approximating function  $\hat{F}(docid)$  we considered piecewise linear functions defined in the log-log plane of Figure 1(right). Let thus  $d = \log(docid)$ , and  $l = \log(length)$ . We try to approximate the function  $l = F(d)$  with the function  $l = \hat{F}(d)$ . This is carried out considering segments of line  $l = a_i d + b_i$  in the  $l - d$  plane. If we select just the number  $n$  of lines used in the approximation, it is easy to show that a generic approximation function  $\hat{F}$  can be analytically expressed as  $\min(a_1 d + b_1, \dots, a_i d + b_i, \dots, a_n d + b_n)$ . In this way, fixed  $n$  and by using non-linear least square minimization, we can approximate quite precisely the shape of the rank-length distribution (see Figure 1). The proposed non-linear approximation has two advantages: firstly, it is simple enough to be rapidly evaluated and secondly, it allows to automatically determine the best points of intersection between two segments. In Table 1 we report the fitting parameters and the Root Mean Square Error (RMSE) measured for four approximations, using respectively one, two, three or four segments in the rank-length log-log plane.

### 3 Experiments

Experiments were performed by using the BM25 weighting model with default parameters, and the CW09B collection. In our experiments, we used the Terrier IR platform. The CW09B corpus was indexed by applying the Porter’s English stemmer and by removing standard stopwords. Positional information was not stored in the index, and each posting consists of only the Elias-Gamma encoded  $docid$  d-gap, and the Elias-Unary encoded frequency. To evaluate the technique proposed in this paper, we first evaluate its impact on the size of the inverted

**Table 2.** StatMAP with and without doc. length approximation

	original	4 seg.	3 seg.	2 seg.	1 seg.
statMAP	0.1245	0.1235	0.1271	0.1230	0.1370

**Table 3.** Query length distribution and response time on CW09B indexes

query length	num. queries	response time (sec)		benefit
		random	doc length	
1	332	1.389	1.036	25.4%
2	311	2.620	2.196	16.2%
3	204	3.022	2.649	12.3%
4	81	4.065	3.631	10.7%

index. We thus built two different indexes for CW09B: in the first case document identifiers were assigned randomly, while in the second case by decreasing document lengths. The first index occupies  $22.43GB$ , while the second only  $19.60GB$ , with a reduction of 12.6% proving the beneficial effect of our technique on index compressibility.

The TREC 2009 Web Track queries with the category B prels relevance judgments<sup>1</sup> were used to measure effectiveness on the top 1000 documents retrieved. The statMAP metrics measured for valid topics when the proposed length approximation strategies are exploited or not (original) are reported in Table 2. By looking at the results reported, it is clear that the proposed technique does not impact negatively retrieval effectiveness. In one case, when the worst approximation (in terms of total RMSE) was used we even obtained a effectiveness boost of about 10%.

The last test conducted aims at evaluating the response times of the Terrier search engine when using or not our technique. We considered the first 1000 queries of a real-world web query log (by MSN). The query length distribution and the average response times measured are reported in Table 3. We can see that response times measured on the smallest index, obtained by assigning identifiers by decreasing document length, are always lower.

## 4 Conclusions

We have proposed an approximation method for computing the length of documents during the query evaluation phase by exploiting the document identifier assigned by decreasing order of document length. Experiments showed that the effectiveness, in terms of MAP, is not affected in a significant way, and that the smaller resulting index allows lower query response times to be achieved.

<sup>1</sup> <http://trec.nist.gov/data/web09.html>

## References

1. Büttcher, S., Clarke, C.L.A., Cormack, G.V.: Information Retrieval: Implementing and Evaluating Search Engines. The MIT Press, Boston (2010)
2. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
3. Robertson, S.E., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval* 3(4), 333–389 (2009)

# Free-Text Search versus Complex Web Forms

Kien Tjin-Kam-Jet, Dolf Trieschnigg, and Djoerd Hiemstra

University of Twente, Enschede, The Netherlands  
{tjinkamj,trieschn,hiemstra}@cs.utwente.nl

**Abstract.** We investigated the use of free-text queries as an alternative means for searching ‘behind’ web forms. We conducted a user study where we evaluated our prototype free-text interface in a travel planner scenario. Our results show that users prefer this free-text interface over the original web form and that they are about 9% faster on average at completing their search tasks.

## 1 Introduction

The internet contains a large amount of information that is only accessible through complex web forms. Journey planners, real estate websites, online auction and shopping websites, and other websites commonly require the user to fill out a form consisting of a number of fields in a graphical interface. The user should first interpret the form and then translate his information need to the appropriate fields. Filling out these forms can be slow because they require mixed interaction with both the mouse and keyboard. A natural language interface (NLI) allows the user to enter his information need in a single textual statement. Rather than navigating between and entering information in the components of the web form, the user can focus on formulating his information need in an intuitive way. NLIs require or assume syntactically well-formed sentences as input, in essence restricting the range of textual input. However, describing all possible natural language statements and dealing with query ambiguity can be a time-consuming process [1,2,3,4]. Therefore, we introduce a free-text interface (FTI) which allows the user to *freely input text* without any restrictions. This work is a stepping stone for further investigation of a single textual interface to access the deep web [5]. Ideally, we wish to use these techniques to build a distributed search system which can search multiple resources, including information behind complex web forms, simultaneously. Our contribution is that we demonstrate that users can search faster with an FTI than with a complex web form, and that they prefer the FTI over the complex web form.

## 2 Experiment and Results

We developed a prototype FTI [6], consisting of a single search box (see Fig. 2), and compared it to an existing travel-planner web form (see Fig. 1). Six information items can be specified in the form: the departure and arrival locations, an optional via location, the time, the date, and a flag indicating whether the



**Fig. 1.** A complex web form, based on the Dutch Railways site

**Fig. 2.** Trajectvinder ('Route Finder'): an FTI, tailored to the complex web form

date and time are for arrival or departure. In our experiment we try to answer the following questions:

*i)* do people prefer to use an FTI over a complex web form? *ii)* is searching by means of an FTI faster than searching by means of a complex web form? *iii)* is there much variation in the query formulations? *iv)* are people consistent in their query formulations? *v)* what are the positive and negative aspects of the FTI? and *vi)* why is the FTI better, or worse, than the complex web form?

## 2.1 Experimental Setup

**Experimental procedure.** The experiment consisted of an offline part, an online part, and a questionnaire. During the offline part, the subjects first provided background information (e.g. age, study). Then, they wrote down their 'most recent travel question' if they could remember it. Next, an information need was shown as a route on a map, along with a desired date and time. The subjects were asked to fill out the complex web form on paper based on this information need. Likewise, but based on a different information need, they filled out the FTI on paper. Finally, the subjects were shown a filled out complex web form, and they reformulated that into a question suitable for the FTI. We aimed to collect query formulations with as little bias to the question as possible. That is why we asked the subjects to formulate a query both from memory, and based on graphical instead of textual descriptions of the information need. During the online part, the subjects first familiarized themselves with the complex interface of the existing travel planner site. Then, they searched for 5 specific train routes and wrote down the departure and arrival times. We recorded the total time to find all routes. Each route was described textually, with a different order of the information items (i.e. the date, time, and locations), and with different wordings (e.g. *ten past one*, or *13:10*). Next, the subjects familiarized themselves with the FTI. After that, they searched for 5 specific routes and wrote down the departure and arrival times, and we recorded the total search time. All questions in the questionnaire, except for the open questions and explanatory questions, were answered on a five-point Likert scale. The subjects indicated whether they thought the FTI was easy to use, if they could find results faster using the FTI, and whether the results of the FTI were correct. They indicated whether or not the FTI was nicer and better, and explained why they thought so. There were

two open questions, asking the subjects to indicate the most negative and the most positive aspects of the system. Finally, they indicated which system they preferred.

**Analysis.** We tested whether the task completion times of the FTI differed significantly ( $p < 0.05$ ) from those of the complex web form, using the Paired Samples T-Test. We also tested whether the five-point Likert scale values differed significantly from neutral (i.e. the number ‘3’), also using the T-Test. Further, we evaluated the query formulation consistency by looking at the order of the information items. Each item was first replaced by a symbol as follows. We replaced the ‘from’ (location) with A, ‘to’ with B, ‘via’ with V, the ‘date’ with D, and the ‘time’ with T. For example, the input “from Amsterdam via Haarlem to The Hague, tomorrow at 10am.” was represented as AVBDT. We then measured the correlation between the subject’s query formulation and the task description using Kendall’s  $\tau$ . Lastly, for each subject, we measured the average Kendall’s  $\tau$  over the combinations of the subject’s formulations.

## 2.2 Results

**The subjects.** A total of 17 subjects (11 male, 6 female) participated in the study. The age distribution ranged from 21 to 66 (median: 27, mean: 32); most subjects were between the age of 21 and 33. The background of the subjects ranged from (under)graduate students in various studies to people working in healthcare, consultancy, and IT-software development. Participation (including the questionnaire) took around 30 minutes on average for each subject.

**The questionnaire.** Comparing the free-text interface (FTI) against the complex web form, the subjects indicated on a five-point Likert scale whether the FTI was: *faster* (**2.4**), *nicer* (**1.8**), *better* (**2.5**), and *preferred* (**2.0**). The numbers in parentheses are the average scores, where ‘1’ indicates full agreement, and ‘5’ denotes the opposite. All results differed significantly ( $p < 0.05$ ) from neutral, except for the third aspect. On average, the subjects felt that they could search a little faster using the FTI than using the complex web form. This was supported by the times measured for the web form and the FTI, with **7.3** and **6.7** minutes on average, respectively. The subjects were significantly ( $p = 0.032$ ) faster, by about 9%, when using the FTI instead of the complex form.

**Pros and cons.** The subjects listed the most negative and most positive aspects of the FTI. The following **negative** aspects were mentioned: 24% of the subjects indicated that there was no example or short manual (forcing the subjects to ‘just type in something, and it worked’); 18% indicated that the interface was too simple, e.g. it lacked pictures; and 12% disliked that they had to ‘click-through’ to obtain the same results as with the complex web form. The following **positive** aspects were mentioned: 41% of the subjects liked how the system ‘understood’ dates like tomorrow and Tuesday, and written time like ‘ten past nine’; 41% liked that you only had to type (without clicking on menus); 35% mentioned

the query-suggestions as a useful feature; and 18% appreciated the fact that the input order of information items (e.g. time, date, places) did not matter.

**Consistency.** When considering only the order of the information items<sup>1</sup> in a query, there were 17 different query formulations. The three most frequent online query formulations were: ABDT 41%, ABVDT 15%, and, tied at third place with 6%, were ABTD, DTABV, and TABVD.

The mean Kendall's  $\tau$  between the online task descriptions and the query formulations was **0.42**. The task with the highest average  $\tau$  (0.96) was sequenced ABDT, the other tasks were BADT (0.67), TABVD (0.39), DTABV (0.09), and TBAD (-0.02). Two subjects always followed the same information order of the descriptions and had an average  $\tau$  of 1.0 (though they used different wordings). Three subjects had an average  $\tau$  between 0.6 and 1.0, and the rest of the seventeen subjects had an average less than or equal to 0.3.

The mean Kendall's  $\tau$  for the (within subjects) online query formulations was **0.64**. Six subjects always formulated their questions in the same order and had an average  $\tau$  of 1; six subjects averaged between 0.7 and 0.9; and, five subjects had an average  $\tau$  less than 0.2.

Overall, the subjects were highly consistent in their query formulations individually; however, there was considerable query variation between subjects. Further, the task descriptions had little effect on the subjects' query formulations; the moderate correlation (0.42) is most probably an artifact caused by subjects consistently formulating their queries as ABDT. This explains the high correlations between the query formulations and the two tasks ABDT and BADT.

### 3 Conclusion

We conducted a user study to compare a free-text interface (FTI) with a complex web form in a travel planner scenario. Our results showed that the subjects could search 9% faster when using the FTI instead of the complex form, and that this finding is significant. Furthermore, they preferred the FTI over the original web form. The results also showed that the subjects were highly consistent in their individual query formulations, and that there was considerable query variation between subjects, even in such a relatively simple scenario.

**Acknowledgments.** This research was supported by the Netherlands Organization for Scientific Research, NWO, grants 639.022.809 and 612.066.513.

### References

1. Sun, J., Bai, X., Li, Z., Che, H., Liu, H.: Towards a wrapper-driven ontology-based framework for knowledge extraction. In: Zhang, Z., Siekmann, J.H. (eds.) KSEM 2007. LNCS (LNAI), vol. 4798, pp. 230–242. Springer, Heidelberg (2007)
2. Kaufmann, E., Bernstein, A.: Evaluating the usability of natural language query languages and interfaces to semantic web knowledge bases. In: Web Semantics: Science, Services and Agents on the World Wide Web (2010)

<sup>1</sup> i.e. the 'date' (D), 'time' (T), and the 'from' (A), 'to' (B), and 'via' (V) locations.

3. Papakonstantinou, Y., Gupta, A., Garcia-Molina, H., Ullman, J.D.: A query translation scheme for rapid implementation of wrappers. In: Ling, T.-W., Vieille, L., Mendelzon, A.O. (eds.) DOOD 1995. LNCS, vol. 1013, pp. 161–186. Springer, Heidelberg (1995)
4. Appelt, D.E., Onyshkevych, B.: The common pattern specification language. In: Proceedings of a Workshop on Held, pp. 23–30. Association for Computational Linguistics, Baltimore (1996)
5. Madhavan, J., Ko, D., Kot, L., Ganapathy, V., Rasmussen, A., Halevy, A.: Google's deep web crawl. Proc. VLDB Endow. 1(2), 1241–1252 (2008)
6. Tjin-Kam-Jet, K., Trieschnigg, D., Hiemstra, D.: Onebox: Free-text interfaces as an alternative to complex web forms. Technical Report TR-CTIT-11-01, Centre for Telematics and Information Technology University of Twente, Enschede (2011)

# Multilingual Log Analysis: LogCLEF

Giorgio Maria Di Nunzio<sup>1</sup>, Johannes Leveling<sup>2</sup>, and Thomas Mandl<sup>3</sup>

<sup>1</sup> Department of Information Engineering, University of Padua, Italy  
dinunzio@dei.unipd.it

<sup>2</sup> Centre for Next Generation Localisation (CNGL), Dublin City University, Ireland  
jleveling@computing.dcu.ie

<sup>3</sup> Information Science, University of Hildesheim, Germany  
mandl@uni-hildesheim.de

**Abstract.** The current lack of recent and long-term query logs makes the verifiability and repeatability of log analysis experiments very limited. A first attempt in this direction has been made within the Cross-Language Evaluation Forum in 2009 in a track named LogCLEF which aims to stimulate research on user behaviour in multilingual environments and promote standard evaluation collections of log data. We report on similarities and differences of the most recent activities for LogCLEF.

## 1 Introduction

Log data containing interactions between users and information systems is important for different research communities. Computer science researchers could study and analyze new algorithms via a common benchmark search log, learn about user information needs and query formulation approaches. Social scientists could investigate the use of language in queries as well as discrepancies between user interests as revealed by their queries versus their interests expressed in face-to-face surveys. Advertisers could use interaction logs to better understand how users navigate to their pages and improve keyword advertising campaigns [1].

Recently, researchers have been addressing the problem of the availability and use of log data: how log files should be made publicly available to researchers, whether log data should be gathered for specific tasks, whether there is value in general log data, and how additional information can be gathered and correlated with query log data [2]. The current lack of recent and long-term data makes the verifiability and repeatability of experiments very limited. It is practically impossible to find two works on the same dataset unless by the same author, or at least one of the authors worked for a commercial search engine company.

## 2 LogCLEF

A first attempt to release a collection of log data with the aim of verifiability and repeatability was done within the Cross-Language Evaluation Forum (CLEF) in 2009 in a track named LogCLEF which is an evaluation initiative for the analysis of queries and other user activities [3]. An important long-term aim of the track is

**Table 1.** Log file resources at CLEF

Year	Origin	Size	Type	Year	Origin	Size	Type
2009	Tumba!	350K	queries	2010	TEL	2.6M	records
2009	TEL	1.87M	records	2010	TEL	1.5 GB (zipped)	activity log
			activity log	2010	DBS	5 GB	web server log
							web server log

to stimulate research on user behavior in multilingual environments and promote standard evaluation collections of log data. In the first two LogCLEF editions, different data sets have been distributed to the participants: search engine query and server logs from the Portuguese search engine Tumba! and from the German EduServer (Deutscher Bildungsserver (DBS)); and digital library systems query and server logs from The European Library (TEL). Table 1 summarizes the log resources and the relative sizes.

In particular, the analyses of the TEL logs are challenging given the nature of the the service. TEL is a free service that offers access to the resources of 48 national libraries of Europe in 35 languages. It aims to provide a vast virtual collection of material from all disciplines and offers interested visitors simple access to European cultural heritage. Resources can be both digital (e.g. books, posters, maps, sound recordings, videos) and bibliographical and the quality and reliability of the documents are guaranteed by the 48 collaborating libraries.

*LogCLEF 2009 participation and results.* Four groups participated in LogCLEF 2009. A thorough analysis of query reformulation, query length and activity sequence was carried out by Ghorab et al. [4]. The group showed that many query modification operations concern the addition or the removal of stopwords. These actions only have an effect for the language collection in which the word is a stop word. The ultimate goal is the understanding of the behavior of users from different linguistic or cultural backgrounds. The application of activity sequences for the identification of communities is also explored. The analysis revealed the most frequent operations as well as problems with the user interface of TEL. Lamm et al. analyzed sequences of interactions within the log file. These were visualized in an interactive user interface which allows the exploration of the sequences [5]. In combination with a heuristic success definition, this system lets one identify typical successful activity sequences. This analysis can be done for users from one top level domain. A few differences for users from different countries were observed but more analysis is necessary to reveal if these are real differences in behavior. In addition, issues with the logging facility were identified.

*LogCLEF 2010 participation and results.* In 2010, seven groups participated (five of which were newcomers). The major topics of interest at LogCLEF 2010 were named entities in queries, language identification (LI) of queries, determining successful searches, and comparing search behaviour between web search and search in TEL data. Bosca et al. [6] experimented on LI in queries. They found that LI is a difficult task because of missing context in queries and that named

entities can lead to misclassifications. For example, “Mozart” may be classified as a German query, but can also be a query in many other different languages. They concluded that LI for queries should be different from LI for documents. The experiments were performed on a manually annotated subset of 100 queries. Stiller et al. [7] analyzed and manually annotated a subset of 510 queries from the TEL data. They found that more than half of the queries are for named entities, which has a huge impact on correct LI of queries. The query language could often not even be manually determined or disambiguated, because many proper nouns are not translated between different languages. They report that seven of the ten most frequent queries contain named entities and that 167 out of 279 named entity queries are ambiguous. Takaku et al. [8] performed an analysis of search sessions and click ranks. They viewed sessions as sequences of actions and durations and compare actions with web search log actions. Leveling et al. [9] investigated the relation between query language, interface language and user IP address. They showed that these aspects correlate and this information can be used to automatically generate a ranking of document collections that better reflects user preferences. In addition they examined query performance indicators for web search and applied them to queries in sessions to find out if performance of user queries increases over time. They found that there are only few consistent changes in consecutive queries on the same topic. However, the first query in a session seems to indicate behaviour in the remainder of the search session: long initial queries seem to be improved by removing terms, while initially short queries will be expanded, Lana-Serrano et al. [10] defined successful queries as queries with results and user interactions on these results. They reported that choosing the native language as the interface language does not affect the success rate of queries. Verberne et al. [11] investigated search behaviour for users of the TEL portal in comparison to ad-hoc searchers using MSN services. The queries do not differ much in average length, but they differ in the topics of interest and in the diversity of languages (mono- vs. multilingual search). In contrast to the TEL data, Web search logs contain a high fraction of navigational and transactional queries while the most frequent TEL queries contain named entities. They also investigated intra-session search behaviour. Perea-Ortega et al. [12] performed a brief analysis of the TEL data, reconstructing user sessions. They analyze TEL queries with a focus on multilingual search and report that nine major European languages cover 95% of all sessions, with 84% of the queries in English.

### 3 Conclusions

LogCLEF has been an active laboratory for researchers in the field of multilingual log analysis where common needs and problems were identified: language identification, named entity recognition in queries, classification of queries, and definition of success of a search. Given the scarcity of resources, LogCLEF organisers promised to continue to support the community by sharing resources and knowledge on log analysis such as annotated data, open source code, documents and articles, which can serve as gold standards or training data for future evaluations. The next edition of LogCLEF will be held as a lab at CLEF 2011.

## Acknowledgments

This work has been partially supported by the PROMISE network of excellence (contract n. 258191) project, as part of the 7th Framework Program of the European Commission.

## References

1. Korolova, A., Kenthapadi, K., Mishra, N., Ntoulas, A.: Releasing search queries and clicks privately. In: Quemada, J., León, G., Maarek, Y.S., Nejdl, W. (eds.) WWW, pp. 171–180. ACM, New York (2009)
2. Clough, P., Berendt, B.: Report on the TrebleCLEF query log analysis workshop 2009. SIGIR Forum 43, 71–77 (2009)
3. Mandl, T., Agosti, M., Di Nunzio, G.M., Yeh, A.S., Mani, I., Doran, C., Schulz, J.M.: LogCLEF 2009: The CLEF 2009 multilingual logfile analysis track overview. In: [14], pp. 508–517
4. Ghorab, M.R., Leveling, J., Zhou, D., Jones, G.J.F., Wade, V.: Identifying common user behaviour in multilingual search logs. In: [14], pp. 518–525
5. Lamm, K., Mandl, T., Koelle, R.: Search path visualization and session performance evaluation with log files. In: [14], pp. 538–543
6. Bosca, A., Dini, L.: Language identification strategies for cross language information retrieval. In: [13], p. 100
7. Stiller, J., Gäde, M., Petras, V.: Ambiguity of queries and the challenges for query language detection. In: [13], p. 99
8. Takaku, M., Egusa, Y., Saito, H., Kando, N., Terai, H., Miwa, M.: CRES at LogCLEF 2010: Towards understanding the user behaviors through an analysis of search sessions, search units and click ranks. In: [13], p. 103
9. Leveling, J., Ghorab, M.R., Magdy, W., Jones, G.J.F., Wade, V.: DCU-TCD@LogCLEF 2010: Re-ranking document collections and query performance estimation. In: [13], p. 101
10. Lana-Serrano, S., Villena-Román, J., Cristóbal, J.C.G.: DAEDALUS at LogCLEF 2010: Analyzing the success of search queries. In: [13], p. 102
11. Verberne, S., Hinne, M., van der Heijden, M., Hoenkamp, E., Kraaij, W., van der Weide, T.P.: How does the library searcher behave? A contrastive study of library search against ad-hoc search. In: [13], p. 99
12. Perea-Ortega, J.M., Ráez, A.M., Cumbreñas, M.A.G., López, L.A.U.: SINAI at LogCLEF 2010. In: [13], p. 100
13. Braschler, M., Harman, D., Pianta, E. (eds.): CLEF 2010 Labs and Workshops. Abstracts Notebook Papers, CLEF 2010, September 22–23. University of Padua, Italy (2010)
14. Peters, C., Di Nunzio, G.M., Kurimo, M., Mostefa, D., Peñas, A., Roda, G. (eds.): CLEF 2009. LNCS, vol. 6241. Springer, Heidelberg (2010)



# A Large-Scale System Evaluation on Component-Level

Jens Kürsten and Maximilian Eibl

Chemnitz University of Technology, Germany

{jens.kuersten,eibl}@informatik.tu-chemnitz.de

**Abstract.** This article describes a large-scale empirical evaluation across different types of English text collections. We ran about 140,000 experiments and analyzed the results on system component-level to find out if we can select configurations that perform reliable on specific types of corpora. To our own surprise we observed that a specific set of configuration parameters achieved 95% of the optimal average MAP across all collections. We conclude that this configuration could be used as baseline reference for evaluation of new IR approaches on English text corpora.

## 1 Introduction

The Cranfield paradigm [1] has been adopted as default setting since the early beginnings of information retrieval (IR) evaluation. In general it is used to provide system rankings to identify the best performing set of IR components over a given set of queries and corresponding relevance assessments on a specific data collection. However, it has been recently discussed whether a ranking of systems is the best way to assess a new idea and its implementation or not [2].

Amongst others [2] the main drawback of the paradigm is the evaluation on system level. In practice publicly available tools like Lemur [3], Lucene<sup>1</sup>, Terrier [4], Xapian<sup>2</sup>, Zettair [5] and others are an excellent development in order to allow thorough analysis of systems and component-level configurations.

Due to the complexity and number of components in current IR systems improving retrieval performance is usually limited to optimize a single component (by developing, implementing and evaluating it) and leaving out the remaining components or at least fix them to a certain extent. Therefore it is almost impossible to draw generally valid conclusions.

To our knowledge the Grid@CLEF pilot task in 2009 [6] was the first attempt to focus on evaluating key components of textual IR systems. The task defined four retrieval steps: tokenization, stop word removal, decompounding, and stemming. Its main focus was multilingual retrieval. In this paper, we follow this inspiring idea and study three key components of IR systems with a large number of commonly used configurations: stemmers, ranking algorithms and feedback models.

---

<sup>1</sup> <http://lucene.apache.org/>

<sup>2</sup> <http://xapian.org/>

## 2 Experimental Setup

In our experiments we investigate various publicly available algorithms covering three major IR system components (see Table 1). The most widely implemented morphological operators are stemmers which lemmatize words from documents in a pre-indexing and user queries in a pre-retrieval stage. Stemming techniques and their effect on retrieval performance have been studied extensively for many European languages and collections in [7]. We decided to include 4- and 5-grams from the n-gram stemming approach because it has been shown that both perform best on English collections [7]. All stemming algorithms we used in our study are listed in the 1<sup>st</sup> column of Table 1 along with their original references.

Nowadays, it is almost impossible to give a complete overview about existing ranking algorithms. A comprehensive classification of commonly used probabilistic models is given in [8]. Again, we selected only a few models to limit the parameter space. In order to simplify the analysis of the results we form three main categories (see 2<sup>nd</sup> column in Table 1). These are: (a) traditional models; (b) type 1 models; (c) type 2 models. According to the theoretic foundations in [8] type 1 models are probabilistic and parameter-free, while type 2 models depend on parameters. Actually, BM25 belongs to the group of type 1 models. But we assigned it to the group of traditional models because it is widely used in IR evaluation.

Pseudo-relevance feedback (PRF) is our last topic of interest. Depending on both collections and queries it is a rather unstable technique. Hence, recent works focused on selective PRF mechanisms that rely on different techniques of query performance prediction [9,10] and risk estimation [11,12] to consistently improve retrieval performance. In this work we apply two models from the Divergence from Randomness (DFR) framework implemented in Terrier [4] in a non-selective way. We compare different configurations for the number of documents and terms to baseline experiments without PRF.

Adding up all variations of different implementations for the tested components leads to a series of exactly 11,895 experiments per collection. The total number results from the combination of five different stemming approaches, 13 ranking algorithms in a set of baseline experiments without PRF and a large series of runs with the two feedback models by altering values for the amount of PRF documents (seven samples: 3, 6, 9, 12, 15, 20 and 30) and PRF terms (13 samples: 5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90 and 100).

**Table 1.** IR system components used in this study

Stemmer	IR model	PRF model
4-Gram, 5-Gram	<i>traditional:</i> TF-IDF, BM25, Lucene	None
Porter, Krovetz	<i>type 1:</i> DFR, DFR_BM25,	Kullback-Leiber (KL)
UeaLite	DLH, DPH, BB2, IFB2, In_Exp, PL2	Bose-Einstein (BE)
	<i>type 2:</i> HiemstraLM, DirchletLM	

### 3 Evaluation

Detailed descriptions of all collections that were used for this work are given in [13]. The queries were automatically constructed from title and description fields of the topic sets. Thorough analysis and interpretation of about 15 million document lists resulting from testing 11,895 experiments on a dozen collections is a challenging task. For that reason we aim to answer the questions, which system configurations achieve strong performance and how reliable performance remains across collections.

Table 2(a) lists best performance (in terms of MAP) for single system configurations using all queries along with best performance when the best system configuration is selected on a per-query basis (MAP\*). In Table 2(b) we report parameters of configurations that achieve highest averaged MAP across all corpora. To obtain a metric that represents this fact, we use an arithmetically averaged mean average precision (AAMAP).

**Table 2.** (a) best performance of system configurations by corpus, (b) top-10 system configurations across all tested corpora in terms of AAMAP compared to optimal AAMAP

Corpus	MAP	MAP*	Stemmer	IR model	PRF(d,t)	AAMA
LIB1	0.3776	0.5992	-	-	-	0.3765
LIB2	0.4187	0.5923	Porter	DLH13	KL(6,30)	0.3578
LIB3	0.3674	0.6621	Porter	TF_IDF	KL(9,80)	0.3518
LIB4	0.4183	0.5688	Krovetz	DLH13	KL(6,20)	0.3503
SPTR	0.3203	0.6175	Porter	In_expB2	KL(6,40)	0.3501
MM1	0.5309	0.5973	Porter	DFR_BM25	KL(6,100)	0.3497
MM2	0.2916	0.4812	Porter	BM25	KL(6,100)	0.3495
MM3	0.2781	0.4219	Porter	DFR	KL(3,20)	0.3480
NEWS1	0.3306	0.5431	Porter	IFB2	KL(6,40)	0.3464
NEWS2	0.3112	0.5600	UeaLite	DLH13	KL(6,100)	0.3450
NEWS3	0.5864	0.7812	Porter	BB2	KL(6,10)	0.3448
NEWS4	0.2876	0.5264				

$$\text{MAP}(Q, C) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} (\text{Precision}(R_{jk}, C)) \quad (1)$$

$$\text{AAMAP}(C) = \frac{1}{|C|} \sum_{i=1}^{|C|} \text{MAP}(Q_i, i) \quad (2)$$

AAMAP allows an intuitive interpretation of cross-collection performance and is formulated in mathematical terms by extending the MAP definition from [14] as presented in equations (1) and (2). The optimal AAMAP (see 1<sup>st</sup> line in Table 2(b)) results from averaging the MAP values for the best system configuration per collection. The best configuration across all collections involves the Porter stemmer, the DLH13 ranking and the KL feedback model and achieves 95% of the optimum. In addition to that, those parameters seem to work consistently well across the tested corpora: Porter's stemming algorithm appears in eight, the KL feedback model in all and the DLH13 ranking model in three out of the reported top-10 configurations.

## 4 Result Discussion

We presented the setup and results of a large-scale empirical study on cross-collection text retrieval in English. In our point of view the most surprising outcome of the experiments was a single system configuration that achieved about 95% averaged MAP across all tested collections. This configuration involves the system components Porter stemmer, DLH13 ranking algorithm and KL PRF model. Furthermore, we observed that each of the elements of this particular configuration appeared highly frequent among the best configurations. We conclude that it might be worth to consider these parameters as baseline for comparison when new system components need to be evaluated.

## References

1. Cleverdon, C.W.: Report on the testing and analysis of an investigation into the comparative efficiency of indexing systems. Tech. report, Cranfield, USA (1962)
2. Hanbury, A., Müller, H.: Automated Component-Level Evaluation: Present and Future. In: Agosti, M., Ferro, N., Peters, C., de Rijke, M., Smeaton, A. (eds.) CLEF 2010. LNCS, vol. 6360, pp. 124–135. Springer, Heidelberg (2010)
3. Ogilvie, P., Callan, J.: Experiments using the Lemur toolkit. TREC 10, 103–108 (2001)
4. Ounis, I., Lioma, C., Macdonald, C., Plachouras, V.: Research directions in terrier: a search engine for advanced retrieval on the Web. *Novatica/UPGRADE Special Issue on Next Generation Web Search*, 49–56 (2007)
5. Billerbeck, B., Cannane, A., Chattaraj, A., Lester, N., Webber, W., Williams, H.E., Yiannis, J., Zobel, J.: RMIT University at TREC 2004 (2004)
6. Ferro, N., Harman, D.: CLEF 2009: Grid@CLEF pilot track overview. In: Peters, C., Di Nunzio, G.M., Kurimo, M., Mostefa, D., Penas, A., Roda, G. (eds.) CLEF 2009. LNCS, vol. 6241, pp. 552–565. Springer, Heidelberg (2010)
7. McNamee, P., Nicholas, C., Mayfield, J.: Addressing morphological variation in alphabetic languages. In: ACM SIGIR 2009, pp. 75–82 (2009)
8. Amati, G.: Frequentist and bayesian approach to information retrieval. In: Lalmas, M., MacFarlane, A., Rüger, S.M., Tombros, A., Tsikrika, T., Yavlinsky, A. (eds.) ECIR 2006. LNCS, vol. 3936, pp. 13–24. Springer, Heidelberg (2006)
9. Zhou, Y., Croft, W.B.: Ranking robustness: a novel framework to predict query performance. In: ACM CIKM 2006, pp. 567–574 (2006)
10. Hauff, C., Azzopardi, L., Hiemstra, D.: The combination and evaluation of query performance prediction methods. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 301–312. Springer, Heidelberg (2009)
11. Lafferty, J., Zhai, C.: Document language models, query models, and risk minimization for information retrieval. In: ACM SIGIR 2001, pp. 111–119 (2001)
12. Collins-Thompson, K.: Reducing the risk of query expansion via robust constrained optimization. In: ACM CIKM 2009, pp. 837–846 (2009)
13. Kürsten, J., Eibl, M.: Vergleich von IR Systemkonfigurationen auf Komponentenebene. In: Internationales Symposium der Informationswissenschaft 2011 (to appear, 2011)
14. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval, p. 160. Cambridge University Press, Cambridge (2009)

# Should MT Systems Be Used as Black Boxes in CLIR?\*

Walid Magdy and Gareth J.F. Jones

Centre for Next Generation Localisation,  
School of Computing, Dublin City University, Dublin 9, Ireland  
{wmagdy, gjoness}@computing.dcu.ie

**Abstract.** The translation stage in cross language information retrieval (CLIR) acts as the main enabling stage to cross the language barrier between documents and queries. In recent years machine translation (MT) systems have become the dominant approach to translation in CLIR. However, unlike information retrieval (IR), MT focuses on the morphological and syntactical quality of the sentence. This requires large training resources and high computational power for training and translation. We present a novel technique for MT designed specifically for CLIR. In this method IR text pre-processing in the form of stop word removal and stemming are applied to the MT training corpus prior to the training phase. Applying this pre-processing step is found to significantly speed up the translation process without affecting the retrieval quality.

## 1 Introduction

Cross-language information retrieval (CLIR) is concerned with searching a collection of documents that are in a different language from the user's query. Two main techniques have been used for the translation step in CLIR; bilingual dictionaries and machine translation (MT) systems [4]. In recent years, MT has become the most commonly used technique in CLIR due to the increasing availability of high quality free MT systems, such as Google translate<sup>1</sup>, Bing translate<sup>2</sup>, and Yahoo Babel Fish<sup>3</sup>. In addition, some open source statistical MT (SMT) libraries are available for research purposes, such as MaTrEx [7] and Moses [1].

Since the MT approach usually provides a high quality translation for queries that consequently leads to high retrieval effectiveness in CLIR close to that of monolingual information retrieval (IR), it has been always used as a black box for the translation process in CLIR. Less attention was directed toward the fact that MT and IR have two different perspectives in measuring the quality of a sentence. MT focuses on generating translations that are semantically, morphologically, and syntactically correct. While IR focuses on retrieving documents that match the query on the conceptual level regardless of the surface form of words.

---

\* This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (CNGL) project at Dublin City University.

<sup>1</sup> <http://translate.google.com/>

<sup>2</sup> <http://www.microsofttranslator.com/>

<sup>3</sup> <http://babelfish.yahoo.com/>

In this paper we open the black box of the MT system and we present a novel technique for using it in a much more efficient way for the purpose of CLIR. The approach introduced utilizes the fact that the surface form and the sentence structure is generally unimportant in standard IR application, To do this, the workflow of the MT process is adapted to focus only on the conceptual meaning of text and neglect its structure. The new setup of the MT system is demonstrated to be five times faster than the standard MT techniques in both the training and decoding phases when tested on the cross language patent search task from the CLEF-IP 2010. The retrieval effectiveness using the new technique of translation is proven to be statistically indistinguishable from results obtained using standard MT.

## 2 New Approach for Using MT in CLIR

An overlooked issue in CLIR systems when MT is used for translation is that MT systems take significant time selecting proper sentence structure for the output, which is unused later by the IR system. Conventional MT focuses on generating a translation that is human readable, therefore it seeks to select the proper pronouns, verb tenses, and word ordering for the translated text. This requires a huge amount of processing power and time for executing an effective algorithm for selecting the proper words since pronouns and verb tenses are generally found to be the most confusing terms in any translation. On the other hand, IR cares more about the conceptual meaning of the word regardless to its surface form and tense. In addition, all the pronouns are considered insignificant to the translated text for IR purposes and are filtered out of the query and the documents prior to their entry into the IR system.

The basic idea in our new approach is to train the MT system for translation of topics or documents in CLIR using training data pre-processed for IR. The pre-processing of IR data uses the standard stages performed by most of the IR systems, which includes case folding, stopword removal, and stemming. These three operations aim to improve retrieval efficiency and effectiveness by removing insignificant words and matching different surface forms of words. While these are standard processes in IR, for MT, applying these operations in the pre-processing stage would appear to be destructive of the quality of the translated sentence. However, since the objective of the translation process here is retrieval effectiveness, the quality of the text structure of the translated content is unimportant. Our hypothesis is that training an MT system using corpora pre-processed for IR can lead to similar or possibly improved translated text from the IR perspective, which consequently can lead to better retrieval effectiveness. In addition, the training of the MT system and subsequent translation is expected to be much faster and more efficient, since a large portion of the text which represents the stopwords will be removed, and the remaining content will be normalized creating a smaller vocabulary, and that a smaller processed training corpus can be as effectively as a larger unprocessed corpus for translation in CLIR.

## 3 Experimental Investigation

To demonstrate the effectiveness and efficiency of the proposed approach, a cross language search in patent retrieval task was used. The main objective is to find

relevant documents in an English collection of patents that are related to French patent applications. The data comes from the CLEF-IP 2010 task [5], where 134 French patent topics are used to search a collection of 1.35M patents that consist of English text only. Since the patent collection comes from the European patent office (EPO) most of the patents in the collection have some parts translated into three languages: English, French, and German. For the MT experiments, 8.1M parallel sentences in English and French were extracted from the collection.

For the CLIR baseline run, the 8.1M parallel sentences were used to train the MaTrEx MT system [7] without pre-processing (referred to later as “ordinary MT”), and then using the output MT model to translate the 134 French patent topics. Since the translated text is in its full form, standard IR pre-processing was applied to the translated text to filter out English stopwords and to stem the words.

The same training data set was used to train the MaTrEx MT system again, but after pre-processing the data to remove stop words, apply case folding, and stem words for both languages in the parallel corpora (referred to later as “processed MT”). The output MT model was used to translate the French topics after applying the same IR pre-processing prior to the translation. Hence the translated text output in this case is in the form of stemmed English words with no stop words.

Queries were constructed from the translated patent topics based on the best runs submitted to the CLEF-IP 2010 [5], where most of sections in the patent topics were used to formulate the query as described in [3]. The time taken for translating these long queries was found to be very long (30 mins per topic using an Intel Xeon quad-core processor, 2.83GHz, 12MB cache, and 32GB RAM), which motivates the need for a more efficient translation process to reduce the translation time. The indri search toolkit was used for indexing and searching the collection [6].

Retrieval effectiveness is measured using two scores; mean average precision (MAP) and the recently introduced patent retrieval evaluation score (PRES) [2]. PRES is an evaluation score designed for recall-oriented retrieval tasks where the objective is to find all possible relevant documents at the highest possible ranks. Significance is tested using Wilcoxon test with  $p$ -value 0.05. The time for training the MT systems and decoding (translating) the patent topics were calculated.

## 4 Results

Table 1 reports the results for the CLEF-IP 2010 CLIR task when using the ordinary MT vs. the processed MT as the translation process. The retrieval effectiveness results were found to be statistically indistinguishable between using the translation techniques when compared using either MAP or PRES. This result shows that processing the query text by removing stop words and stemming will lead to the same retrieval results regardless of whether it is applied before or after the translation process. The other results in Table 1 show the main benefit of applying the new “processed MT” approach, which is the MT processing time. It can be seen that the processed MT is much faster than the ordinary MT, since it is more than five times faster in both the training and decoding (translation) phases. These results confirm that adapting the MT system for IR use to be much more efficient than using it as a black box while maintaining the retrieval effectiveness.

**Table 1.** Retrieval effectiveness and processing time compared when using ordinary MT vs. processed MT for the CLEF 2010 cross language patent search task

		Ordinary MT	Processed MT
<b>Retrieval Effectiveness</b>	MAP	0.085	0.084
	PRES	0.413	0.419
<b>Processing Time</b> ( <i>hh:mm:ss</i> )	Training (8M sentences)	221:31:28	44:11:16
	Decoding (134 patents)	68:18:21	13:29:39

## 5 Conclusion and Future Work

This paper studied the use of MT systems in CLIR with the objective of discovering whether there is a way of training MT systems specifically for IR instead of using them as black boxes. We presented an efficient technique for training MT systems for the purpose of CLIR by re-ordering the workflow of the CLIR steps to apply the standard IR pre-processing prior to the translation process instead of after it, and to train the MT system in the same fashion by processing the parallel corpus before the MT training. Testing the suggested approach on a cross language patent search task showed the new translation process to be five times faster than the ordinary MT system while preserving the same retrieval quality.

For future work, the approach needs to be further tested on different language pairs. In addition, the performance of the new MT approach is required to be investigated when only limited amount of MT training corpus is available.

## References

1. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open Source Toolkit for Statistical Machine Translation. In: ACL 2007 (2007)
2. Magdy, W., Jones, G.J.F.: PRES: a score metric for evaluating recall-oriented information retrieval applications. In: SIGIR 2010 (2010)
3. Magdy, W., Jones, G.J.F.: Applying the KISS Principle for the CLEF-IP 2010 Prior Art Candidate Patent Search Task. In: CLEF 2010 (2010)
4. Oard, D.W., Diekema, A.R.: Cross-Language Information Retrieval. In: Williams, M. (ed.) ARIST (1998)
5. Piroi, F.: CLEF-IP 2010: Retrieval Experiments in the Intellectual Property Domain. In: CLEF 2010 (2010)
6. Strohmman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: ICIA (2004)
7. Stroppa, N., Way, A.: MaTrEx: DCU Machine Translation System for IWSLT 2006. In: IWSLT (2006)



# Video Retrieval Based on Words-of-Interest Selection

Lei Wang, Dawei Song, and Eyad Elyan

School of Computing, The Robert Gordon University, United Kingdom  
{l.wang4, d.song, e.elyan}@rgu.ac.uk

**Abstract.** Query-by-example video retrieval is receiving an increasing attention in recent years. One of the state-of-art approaches is the Bag-of-visual Words (BoW) based technique, where images are described by a set of local features mapped to a discrete set of visual words. Such techniques, however, ignores spatial relations between visual words. In this paper, we present a content based video retrieval technique based on selected Words-of-Interest (WoI) that utilizes visual words spatial proximity constraint identified from the query. Experiments carried out on a public video database demonstrate promising results of our approach that outperform the classical BoW approach.

**Keywords:** Bag-of-Words, Content based video retrieval, Words-of-Interest.

## 1 Introduction

The rapid growth of video resources available on the internet requires effective content based video retrieval technology. A scenario that has attracted increasing attention is “query by a visual example”, where a user submits a video as a query to search against a database of video for best matches. A state of the art approach is the Bag of visual Words (BoW) model that has been widely applied in content based image/video retrieval [1]. In BoW, the images are described by a set of local feature descriptors that are mapped into discrete visual words [1].

However, BoW ignores inter-word spatial relationships [5], assumes that visual words are generated independently, and assigns all visual words with equal prior importance. A problem is that visual words associated with users interests are always messed up with redundant information. The mixture of Words-of-Interest (WoI) and other words in given query would lead to irrelevant results.

Recently, several techniques have been proposed to address above problems. Cao et al. [5] utilized spatial coherence of neighboring visual words to enhance the Latent Topic Model for object classification. Liu and Chen [3] proposed a method to represent a video by regional characteristics, namely Object-of-Interests (OoI) extracted. However, the offline OoI extraction may exclude relevant information, because the interests of user are very hard to determine prior to online search. Zhang et al. [2] argue that selected Descriptive Visual Word (DVP) through supervised training improve the performance of image retrieval and object recognition.

In this paper, we propose a novel approach based on the selected WoI according to the spatial proximity imposed by given query. The WoI selection is based on assumptions that a salient visual word tends to co-occur with and close to the other important

ones. We rank the importance of the visual words based on these assumptions and select the WoI without supervised learning and training data.

The rest of the paper is organized as follows. In Section 2, WoI generation algorithms and the WoI based video retrieval are presented. The experiment and results are discussed in Section 3. Finally, Section 4 concludes the paper.

## 2 Words-of-Interest Generation and Video Retrieval

The BoW representation is adopted, and local feature descriptors are extracted using the Speed Up Robust Feature algorithm (SURF) [4]. The descriptors are grouped to generate a vocabulary  $\text{Voc} = \{w_j\}, j = 1 \dots N_w$  using K-means clustering.

A video  $V$  is represented as a set of key frames  $\{f_c\}$ , and each  $f_c$  is a weighted vector of visual words:  $f_c = \{tf_j\}, j = 1 \dots N_w$ . In this paper, the weights of visual words in the  $f_c$  is Term Frequency (TF) [2]. Similarly, the given query is represented as a set of  $\{f_q\}$ . The distance  $|f_c - f_q|$  is computed for frame level similarity match.

### 2.1 Spatial Proximity Matrix Generation from Query

According to the assumption that the WoI co-occur closely with each other within the query, we propose to use a weighted distance  $\hat{d}$  to capture the spatial proximity among the visual words pair within the  $l^{\text{th}}$  key frames of the query:

$$\hat{d}_{i,j}^l = \frac{1}{N_i * N_j} \sum_{k=1}^{N_i} \sum_{m=1}^{N_j} d_{k,m} / \sigma_k, \quad (1)$$

where  $d_{k,m}$  is the Euclidean pixel distance between the  $k^{\text{th}}$  instance of a visual word  $w_i$  and  $m^{\text{th}}$  instance of  $w_j$ ,  $\sigma_k$  is the scale factor obtained from the SURF [4] descriptor, and  $N_i$  and  $N_j$  indicate the number of instances of  $w_i$  and  $w_j$  respectively. The proximity of  $w_i$  and  $w_j$  in the given query  $q$  is computed as:

$$s_{i,j}^q = N_q / \sum_{l=1}^{N_q} \exp(\hat{d}_{i,j}^l), \quad (2)$$

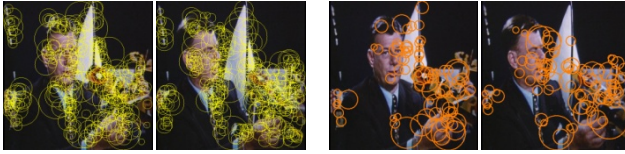
where  $N_q$  is the number of the key frames in the query  $q$ . For all visual words pairs, a  $N_w \times N_w$  spatial proximity matrix  $M_q$  is generated and each entry is computed by equation (2). It will be used for WoI selection in the next section.

### 2.2 WoI Selection and WoI Based Similarity Match

Our WoI selection algorithm is formalized as visual words ranking based on two criteria: i) the WoI co-occur more frequently in the query, and ii) the WoI are of a greater spatial proximity with each other. Figure 1 is an example of selected WoI.

Let  $r_i$  be average spatial proximity of a visual word  $w_i$  to other words:

$$r_i = \sum_{j=1}^{N_w} S_{i,j}^q * p(j|q), \quad p(j|q) = \frac{tf_j}{N_q} \quad (3)$$



**Fig. 1.** original BoW instances (yellow circles), selected WoI instances (orange circles)

where  $p(j|q)$  is the conditional probability that  $w_j$  occurs in the query  $q$ ,  $tf_j$  is the term frequency of the visual word  $w_j$  and  $N_q$  is the number of instances of the visual words in  $q$ . We define a proximity rank vector  $R_w \equiv \{r_i\}, i = 1 \dots N_w$  and the initial values of the  $r_i$  are computed as (3).

If a visual word is of greater proximity, the visual words co-occurring closely with it are more important, and we propose a recursive ranking algorithm as follows:

**Algorithm 1.** visual word proximity ranking

**Input:** spatial Matrix  $M_q$ , term frequency vector  $F_q$ , maxiter(maximum iteration steps)

**Output:** Visual word rank  $R_w$  due to the given query

**Begin:**  $R_w^1 = F_q / \text{sum}(F_q)$ ;

For  $i = 1:\text{maxiter}$

$R_w^{i+1} = M_c \times R_w^i$ ; Normalize( $R_w^{i+1}$ );

If  $|R_w^{i+1} - R_w^i| < \epsilon$

$R_w^{\text{maxiter}} = R_w^{i+1}$ , break;

End

$R_w = \text{sort}(R_w^{\text{maxiter}})$ ;

In the algorithm 1, the visual word  $w_j$  ranking on top of  $R_w$  is selected as WoI:  $WoI = \{w_j\}, j = id_1 \dots id_k \dots id_{N_{woi}}$ , where  $id_k$  is the index of  $k^{\text{th}}$  WoI in the visual vocabulary  $Voc$ , and  $N_{woi}$  is the number of the selected WoI.

A frame level similarity based on the WoI is measured by the distance:

$$\bar{D} = |f_c - f_q| + \alpha_{woi} * |\hat{f}_c - \hat{f}_q|, \quad (4)$$

where  $\hat{f}_c$  represents the term frequency of WoI in the key frame of the video and  $\hat{f}_q$  represents the key frame of the query,  $\alpha_{woi}$  is weighting factor.

Finally, the more key frames a video has which are similar to those in the query, the higher the video is ranked in the retrieved results.

### 3 Experiment and Results

To evaluate the WoI based video retrieval, we select a dataset from TRECVID 2002, which consists of 3000 videos, 6 query topics and 18 queries. The evaluation is based on common criteria used in the information retrieval community: Precision, Recall and Mean Average Precision (MAP).

Figure 2 compares the precision and recall performances of the WoI and BoW based video retrieval.

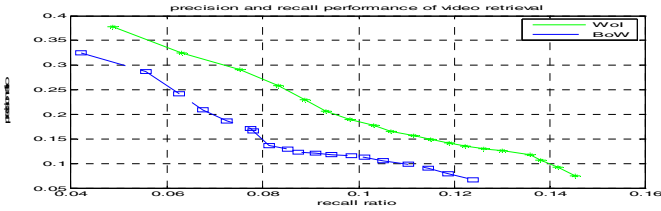


Fig. 2. Precision and recall comparison of WoI and BoW

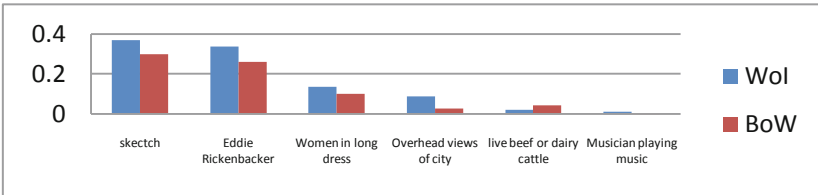


Fig. 3. AP comparison of WoI and BoW for several topics

It is clear that when the approaches based on the WoI and BoW perform the same on recall ratio, the WoI based approach clearly outperforms the BoW on precision ratio.

Figure 3 shows that for 5 out of 6 topics, our WoI based approach outperforms the classic BoW on the average precision (AP).

## 4 Conclusions and Future Work

In this paper, we propose a technique for selecting relevant visual words of users interest based on the discovery of the spatial proximity between the visual words within the query. The experimental results show that the selection is effective and the generated WoI could improve query-by-example video retrieval performance compared with the classical BoW approach.

Our future work will be focused on developing a more descriptive WoI selection method by incorporating temporal information existing between frames.

## References

1. Sivic, J., Zisserman, A.: Efficient visual search for objects in videos. *Proceedings of the IEEE* (2008)
2. Zhang, S., Tian, Q., Hua, G., Huang, Q., Li, S.: Descriptive Visual Words and Visual Phrases for Image Applications. In: *ACM MM* (2009)
3. Liu, D., Chen, T.: Video retrieval based on object discovery. In: *CVIU* (2009)
4. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: SURF: Speeded Up Robust Features. In: *CVIU*, pp. 346–359 (2008)
5. Cao, L., Fei-Fei, L.: Spatially coherent latent topic model for concurrent object segmentation and classification. In: *ICCV* (2007)

# Classic Children’s Literature - Difficult to Read?

Dolf Trieschnigg<sup>1</sup> and Claudia Hauff<sup>2</sup>

<sup>1</sup> DB group

<sup>2</sup> HMI group,

University of Twente, Enschede, The Netherlands

{trieschn,hauffc}@ewi.utwente.nl

**Abstract.** Classic children’s literature such as *Alice in Wonderland* is nowadays freely available thanks to initiatives such as Project Gutenberg. Due to diverging vocabularies and style, these texts are often not readily understandable to children in the present day. Our goal is to make such texts more accessible by aiding children in the reading process, in particular by automatically identifying the terms that result in low readability. As a first step, in this poster we report on a preliminary user study that investigates the extent of the vocabulary problem. We also propose and evaluate a basic approach to detect such difficult terminology.

## 1 Introduction

Many classic works of children’s literature like *Alice in Wonderland* are nowadays freely available thanks to initiatives such as Project Gutenberg<sup>[1]</sup> (PG), a digital library which contains mostly public domain books. Many of these books were written more than one hundred years ago. Due to the diverging vocabularies, style and wording of the texts, they are often not readily understandable to children today. Our goal is to breach this vocabulary gap in order to make these classic works more accessible to children today.

Consider, for instance, the extract from *The Three Musketeers* in Figure <sup>[2]</sup> underlined are (multi-word) terms that we hypothesize to be unknown by most children today. Though the extract is short, five words are easily recognizable as unusual in today’s English. The table in Figure <sup>[3]</sup> gives an indication of the vocabulary mismatch between works of different time periods. We downloaded between 5 and 10 books from PG’s children’s literature category for each 25 year time period starting with 1775-1800 and ending with 1900-1925. The unigram term distribution over all books of a time period was then calculated. Reported is the Jensen-Shannon divergence (JSD) <sup>[3]</sup> between those distributions. The larger the divergence, the more the term distributions of different time periods differ. As can be expected, with increasing difference in time periods, the divergence in the vocabulary increases. While this experiment only considers unigrams and not multi-word terms (or even the style of writing), it already shows that on the single term level alone large differences occur.

---

<sup>1</sup> <http://www.gutenberg.org/>

To investigate the extent of the vocabulary gap in children’s literature, we performed a user study where subjects were asked to tag the terms in paragraphs of children’s books they were unfamiliar with or considered unusual. Those tagged terms were then used as a gold standard and a mechanism was developed to automatically detect them. We have two specific applications in mind: (i) a system that takes digital books from PG or other sources as input and generates an enhanced book in which difficult terminology is automatically linked to a definition, and, (ii) a system that aims to acquaint children with older vocabulary. While it would also be possible to create a system where children would interact with every word they do not know, it requires a lot of user input (such as clicks) and makes the reading process more tedious, in particular, if the number of unknown words is high.

	1800	1825	1850	1875	1900	1925
1800	0	0.25	0.26	0.33	0.34	0.40
1825		0	0.25	0.34	0.35	0.42
1850			0	0.25	0.26	0.30
1875				0	0.25	0.29
1900					0	0.26
1925						0

“The citizens always took up arms readily against thieves, wolves or scoundrels, often against nobles or Huguenots, sometimes against the king, but never against cardinal or Spain. [...] the citizens, on hearing the clamor, and seeing neither the red-and-yellow standard nor the livery of the Duc de Richelieu, rushed toward the hostel of the Jolly Miller.”

**Fig. 1.** The table shows the divergence in term distributions derived from books written in various time periods (1800 indicates the time period 1775-1800, and so on). The book extract on the right is from *The Three Musketeers* by Alexandre Dumas père (written in 1844). Underlined are terms we consider mostly unknown to children today.

To the best of our knowledge, no existing work deals specifically with the automatic detection of hard terminology in classic books. More remotely related work can be found in the area of readability metrics for texts as a whole [14] and text simplification [2].

This poster is organized as follows: in Sec. 2 we present our user study. The detection approach is outlined in Sec. 3. Future work is discussed in Sec. 4.

## 2 User Study

For the user study, ten children’s books were selected, ranging in time from *Robinson Crusoe* (1719) to *The Adventures of Paddy the Beaver* (1917). From each book, three paragraphs were randomly selected. The paragraph length ranges from 134 to 268 words. We recruited 30 subjects (11 female, average age: 31.8) from different research groups of several universities. All subjects are non-native English speakers; twenty-five participants judged their knowledge of English between 3-5 on a 1-5 Likert scale (where 5 indicates excellent). In this preliminary study, we make the simplifying (and debatable) assumption that terms that are tagged by good non-native English speakers are likely to be unknown to young native English speakers as well. The subjects were asked to tag the terms they were either unfamiliar with or deemed “difficult, rare or unusual”.

We deliberately kept the description vague to get an understanding of what kind of terms the subjects consider difficult. The subjects were asked to tag three or more paragraphs; the average number of tagged paragraphs per subject was 4.1 and every paragraph was tagged by at least three subjects.

A (multi-word) term is added to the gold standard if at least half of all subjects that were presented the paragraph tagged it. Slight differences in tagging (e.g. one subject tags *stoop*, another tags *stoop down*) were manually cleaned up. This resulted in 39% of all the terms tagged by one or more subjects to be included in the gold standard. This number suggests that the agreement between the subjects was not always high. Some subjects tagged more than others, some tagged whole sentences. In the latter case, the sentence structure was deemed unusual. On average, 3.9 terms ( $\sigma = 2.1$ ) were tagged in each paragraph. The minimum number (1 tag) was found in a paragraph from *The Adventures of Paddy the Beaver* (written in 1917), the maximum number (10 tags) was found in a paragraph from *The Legend of Sleepy Hollow* (written in 1820). The percentage of unique terms tagged in a paragraph varied between 0.9% and 8.3%, the average being 3.4%. The percentage of tagged terms varied considerably for the different paragraphs of each book. We did not observe a significant correlation between the year of writing and the percentage of unknown terms in a paragraph. Examples of tagged terms are *heathens*, *bonnets*, *garments*, *vicarage*, *horse balls*, *hillock* and *oratory*.

### 3 Detecting Difficult Terminology

We implemented a basic algorithm for the detection of difficult terminology backed up by definitions in Wiktionary<sup>2</sup>. The dictionary was compiled as follows. Wiktionary entries, limited to English nouns, verbs, adjectives and adjectives were extracted from a dump of Wiktionary pages. Stopwords and entries belonging to manually composed lists of inappropriate categories (such as *English basic words*, and *Sex*) were removed. Plural nouns and different verb forms were linked to their singular and simple present tense form, respectively. The filtered dictionary consists of 261,243 unique terms with 340,577 definitions. As an indicator of the difficulty of the terms, we determined the inverse collection frequency (ICF) of a term  $t$  (which is treated as a set of one or more words):  $ICF(t) = \min_{w \in t} \log \frac{n}{\max(cf(w), 1)}$ , where  $n$  is the total number of words in the collection and  $cf(w)$  is the number of times the word  $w$  appears in the collection. Our intuition was that more difficult terminology has a higher ICF. We used a dump of Simple Wikipedia and a selection of 114 Gutenberg books (mainly novels, written between 1719-1920) as two sources to determine ICF.

Difficult terminology is detected in a two step process. (i) Finding candidates: a candidate set of difficult terms is detected by scanning the text for terms in the dictionary. In case overlapping terms are detected in the same string, the longest term is preferred. (ii) Filtering: candidate terms with an ICF below a predetermined threshold (based on training data) are discarded.

The collection obtained from the preliminary user study was used to evaluate our algorithm. 93% of the tagged terms also appeared in our dictionary.

<sup>2</sup> <http://www.wiktionary.org/>

In particular, some hyphenated terms were missed. We evaluated the detection performance in terms of precision (P), recall (R) and F-measure (F1) with and without ICF filtering based on 11-fold cross-validation. The fold (3 paragraphs) was used for training the optimal threshold, yielding the highest F1, the remaining paragraphs were used for testing. The reported metrics indicate the average over 11 folds. As we expected, no filtering resulted in low precision (P:0.10, R:0.89, F1:0.17). ICF filtering clearly improved precision at a smaller loss of precision, resulting in an optimal F-measure of 0.46 (based on Wikipedia: P:0.37, R:0.66, F1:0.24; based on Gutenberg books: P:0.44, R:0.53, F1:0.46). Given the relatively low inter annotator agreement (only 39% of the terms was tagged by multiple subjects), we think this is a reasonable first performance. The false positives of the Simple Wikipedia filtering revealed its limited coverage of typical story words (verbs such as *remarked*, and *nodded*). In contrast, the ICF from the Gutenberg books sometimes falsely filters nowadays uncommon terms. A combination of filtering from both sources might overcome these errors. Further improvements might come from additional features, such as number of syllables in the terms and using multiple ICF thresholds for different parts of speech.

Finally, we applied the detection mechanism with the Gutenberg corpus on each of the ten children's books selected for our user study. The percentage of *unique* terms detected as difficult varied between 8.8% (*The Adventures of Paddy the Beaver*, 1917) and 30% (*Tarzan of the Apes*, 1914). In the latter case, the five most often occurring difficult terms are *cruiser*, *gorilla*, *locket*, *bugs*, *primeval*.

## 4 Summary and Future Work

In this poster we reported the results of a user study that investigates the amount of unknown/unusual vocabulary found in classic children's books. The results show that the vocabulary gap is significant and needs to be addressed in a system that attempts to make classic children's books accessible to young readers. Our attempt at an automatic detection mechanism yielded reasonably good results.

In this preliminary study we relied on non-native English speakers in lieu of young native speakers, assuming that both types of users would tag similar words. This assumption needs to be tested in a further user study with children. Such a study would also give insight into how large the vocabulary gap is for different age groups.

**Acknowledgements.** This research was supported by the Netherlands Organization for Scientific Research, NWO, grant 612.066.513.

## References

1. Collins-Thompson, K., Callan, J.: Predicting reading difficulty with statistical language models. *JASIST* 56(13), 1448–1462 (2005)
2. De Belder, J., Moens, M.F.: Text simplification for children. In: *Towards Accessible Search Systems Workshop*, pp. 19–26 (2010)
3. Lin, J.: Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory* 37(1), 145–151 (1991)
4. Petersen, S.E., Ostendorf, M.: A machine learning approach to reading level assessment. *Computer Speech and Language* 23(1), 89–106 (2009)



# Applying Machine Learning Diversity Metrics to Data Fusion in Information Retrieval

David Leonard, David Lillis, Lusheng Zhang,  
Fergus Toolan, Rem W. Collier, and John Dunnion

School of Computer Science and Informatics,  
University College Dublin, Ireland  
{david.leonard,david.lillis,lu-sheng.zhang,  
fergus.toolan,rem.collier,john.dunnion}@ucd.ie

**Abstract.** The Supervised Machine Learning task of classification has parallels with Information Retrieval (IR): in each case, items (documents in the case of IR) are required to be categorised into discrete classes (relevant or non-relevant). Thus a parallel can also be drawn between classifier ensembles, where evidence from multiple classifiers are combined to achieve a superior result, and the IR data fusion task.

This paper presents preliminary experimental results on the applicability of classifier ensemble diversity metrics in data fusion. Initial results indicate a relationship between the quality of the fused result set (as measured by MAP) and the diversity of its inputs.

## 1 Introduction

Data fusion is a technique for combining the ranked lists of documents returned by multiple Information Retrieval (IR) systems in an attempt to improve performance. One rationale for the success of data fusion is similarity between the relevant documents and diversity among the non-relevant documents returned by the component systems [1]. Similarly, in the area of Supervised Machine Learning (SML), diversity with respect to the errors committed by component classifiers in ensembles has received much attention. In particular, it has been proposed that the accuracy of and diversity between these systems are necessary and sufficient conditions to improve the performance of the combined system on classification learning tasks [2]. In an attempt to formalise such a relationship, Kuncheva devised 10 metrics to characterise diversity among classifiers [3]. This paper presents initial work in adapting one of these metrics to data fusion in order to explore the question of an accuracy/diversity trade-off in IR.

## 2 Background

In Machine Learning, classification is the problem of selecting the correct class for a data point from a discrete set of class labels. Multi-classifier systems combine the outputs of an ensemble of classifiers in an attempt to yield more accurate

classification performance. A question that arises across all the different incarnations of multi-classifier techniques is whether there are certain characteristics of the individual component classifiers that guarantee improved performance of the combined system.

The primary experimental work in the field of diversity and its relationship to accuracy in multi-classifier systems has been carried out by Kuncheva [34]. In an attempt to quantify diversity, ten metrics were formalised, which operate by studying the relationship between classifier outputs at the so-called “oracle” level (i.e. for each data point a classifier scores 1 if it classifies it correctly and 0 otherwise). These scores are then aggregated across the entire dataset of training points to obtain a measure of diversity. An example of such a metric is the entropy measure,  $E$ , which is given by:

$$E = \frac{1}{N} \sum_{j=1}^N \frac{1}{(L - \lfloor L/2 \rfloor - 1)} \min \left\{ \sum_{i=1}^L y_{j,i}, L - \sum_{i=1}^L y_{j,i} \right\} \quad (1)$$

where  $L$  is the number of classifiers in the ensemble,  $N$  is the total number of data points in the training dataset and  $y_{j,i}$  is the value (zero or one), that the  $i$ th classifier received on the  $j$ th data point.  $E$  varies between 0 and 1 where 0 indicates no difference and 1 represents the highest possible diversity.

### 3 Mapping

At the heart of the current work is the mapping of the metrics between the domains of SML and IR. As mentioned above, in the Machine Learning context the metrics operate at the oracle level (i.e. the output of each classifier is either correct or incorrect), classifiers may agree or disagree and be right or wrong in this respect. Analogously, in the IR context a document may be relevant or non-relevant and similarly IR systems may agree or disagree about this. If, in response to a query, an IR system returns a document then this may be considered as evidence that it considers it to be relevant and likewise the absence of a document in a result set is an affirmation that the system views it to be non-relevant.

One key difference between these scenarios is the notion of ordering. As designed, the metrics operate on unordered sets of outputs. However, the ranked lists returned by IR systems impose an ordering relation between the documents. The strategy outlined above takes a global measurement of diversity without taking this into consideration. It can be argued that this is not necessary because the accuracy (as measured by IR evaluation metrics such as MAP), of the individual IR systems have already taken ranking information into account. A second, related, point is the implicit assumption of an arbitrary cut-off point in the ranked lists, beyond which the systems no longer consider documents to be relevant.

## 4 Experimental Work

At this early investigative stage of the research the emphasis is on a) the ability of the metrics to capture diversity between document result sets and b) whether a relationship between diversity, accuracy and combined performance may be postulated. A number of decisions were made with respect to the parameters and scope of the experiment to investigate this.

Using inputs from the TREC 2004 Web Track, 51 queries were chosen for which there were between 8 and 147 relevant documents. Many of the available queries only had one relevant document associated with them. In this case, all systems would frequently return that relevant document somewhere in the 1000 documents they returned, resulting in no diversity being found between them. Avoiding queries for which there were very few relevant documents available avoids causing bias in the metrics in this way. In order to reduce the impact of differing-length result sets, 35 inputs were chosen that tended to return 1000 documents (the TREC maximum). The performance of these systems (measured by MAP) varied widely. Teams of 5 systems were fused using SlideFuse [5]. The performance (or ‘accuracy’ in SML terms) was measured by MAP. Diversity was quantified using the entropy measure presented in Section 2.

The entropy measure was calculated on a per query basis. For each of the  $N$  judged relevant documents, if a system included it in its result set it was given a value of 1 and otherwise it was given a value of 0 for that system. These intermediate statistics were then averaged across all queries resulting in a single value for the entropy of the fused system. The average MAP score for the inputs and the combined MAP score (of the fused output) were similarly aggregated across the query set.

### 4.1 Results

The first 250,000 combinations of the 35 candidate systems into ensembles of size 5, were fused using SlideFuse. Statistics pertaining to performance and diversity were gathered, from which the compound metrics were derived. A subsequent plot of these results was not promising, failing to reveal a pattern between the 3 variables at a global level across the entire spectrum of possibilities for accuracy and diversity. The original hypothesis proposed that the accuracy of and diversity between the component systems are necessary and sufficient conditions to improve the performance of the combined system. To investigate this relationship, the results were sorted with respect to the average MAP score and the top 5000 highest performing or most accurate combinations set aside. Correlation between each set of variables was measured using Pearson’s correlation coefficient,  $r$ . Again, there did not appear to be a clear relationship between either the average and combined MAP, ( $r = 0.19$ ), or between entropy and average MAP ( $r = -0.12$ ). There does, however, appear to be a pattern with respect to entropy and the combined MAP score ( $r = 0.80$ ). A plot of this relationship is shown in fig. 1. With reference to this figure it is clear that the combined MAP scores are higher for systems with a larger, hence more diverse, entropy value.

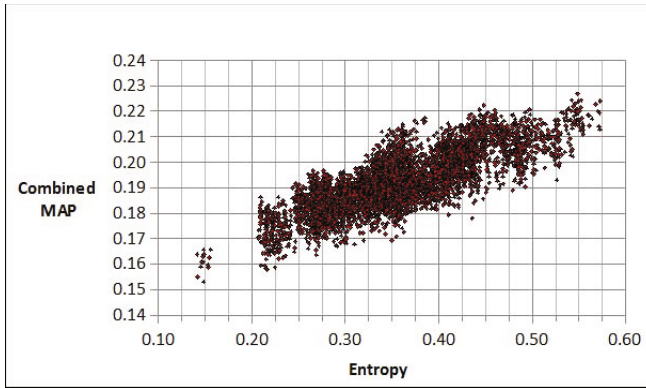


Fig. 1. A plot of Entropy versus Combined MAP for the top 5000 systems

## 5 Conclusions and Future Work

Based on the preliminary experimental work carried out to date there is evidence of a possible relationship between the combined performance of fused result sets, the average performance of the input result sets and the diversity between the relevant documents returned in the component result sets, as measured using the entropy metric.

In future work, it is proposed to investigate the suitability of metrics other than entropy for capturing diversity between document result sets, the role of diversity between non-relevant or unjudged documents and whether there is a particular cut-off point where the metrics should be applied (e.g. the top 100 rather than 1000 documents). It will also be necessary to ascertain whether such a result generalises to fusion techniques other than Slidefuse and, if so, determine the characteristics of the family of fusion techniques to which the diversity metrics are applicable.

## References

1. Lee, J.H.: Analyses of multiple evidence combination. *SIGIR Forum* 31, 267–276 (1997)
2. Dietterich, T.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) *MCS 2000*. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
3. Kuncheva, L., Whitaker, C.: Ten measures of diversity in classifier ensembles: limits for two classifiers. In: *IEEE Workshop on Intelligent Sensor Processing*, Birmingham, UK (2001)
4. Shipp, C., Kuncheva, L.: Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion* 3(2), 135–148 (2002)
5. Lillis, D., Toolan, F., Collier, R., Dunnion, J.: Extending Probabilistic Data Fusion Using Sliding Windows. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008*. LNCS, vol. 4956, pp. 358–369. Springer, Heidelberg (2008)

# Reranking Collaborative Filtering with Multiple Self-contained Modalities

Yue Shi, Martha Larson, and Alan Hanjalic

Multimedia Information Retrieval Lab, Delft University of Technology,  
Mekelweg 4, 2628CD Delft, Netherlands  
{y.shi,m.a.larson,a.hanjalic}@tudelft.nl

**Abstract.** A reranking algorithm, Multi-Rerank, is proposed to refine the recommendation list generated by collaborative filtering approaches. Multi-Rerank is capable of capturing multiple self-contained modalities, i.e., item modalities extractable from user-item matrix, to improve recommendation lists. Experimental results indicate that Multi-Rerank is effective for improving various CF approaches and additional benefits can be achieved when reranking with multiple modalities rather than a single modality.

**Keywords:** Recommender systems, collaborative filtering, reranking, multiple modalities, self-contained modalities.

## 1 Introduction

As one of the most popular and successful recommendation techniques, collaborative filtering (CF) provides personalized recommendations to users [1]. Little investigation has been devoted, however, to techniques that take a two-step approach, i.e., generate an initial recommendation using CF and then refine it via reranking. The lack of attention is surprising, given the advantages of a reranking approach. The reranking step is independent of the initial recommendation and, for this reason, reranking is flexible and can be used jointly with any technique that provides an initial recommendation list. Moreover, reranking makes it possible to improve performance by exploiting additional information/modalities whose potential is not fully exhausted when generating the initial recommendation.

Reranking is attractive for practical application, since it is possible to achieve improvements with low-cost reranking modalities i.e., modalities that do not rely on external resources or require intensive computational effort to extract. Additionally, reranking is able to preserve the underlying strength of the CF approach used to generate the original list. These properties have been established by [6] which combined the concept of Bayesian reranking from video search [7] with item representations, called self-contained modalities, that are derived from the user-item matrix and independent of external information. This work showed that reranking using self-contained modalities is able to improve recommendation from item-based CF [2]. To our knowledge, this is the only existing work on reranking applied to CF.

We propose a novel reranking approach for CF, Multi-Rerank, which not only maintains the benefits of self-contained reranking, but also succeeds in exploiting

multiple modalities simultaneously for reranking. We demonstrate the effectiveness of Multi-Rerank combined with several well-known CF approaches used to generate the initial results list. Finally, we experimentally verify that Multi-Rerank can capture benefits from multiple modalities and achieve substantial improvement in performance over reranking with single modality. The benefit is shown to be consistent over user types with profiles of different lengths.

## 2 The Multi-rerank Algorithm

Given an initial ranking list  $r^{init}=[r_1^{init}, r_2^{init}, \dots]$  which contains initial ranking scores for items, and given multiple modalities  $w^{(k)}$ ,  $k=1, 2, \dots, K$ , a loss function is defined as:

$$L(r_1, r_2, \dots, r_Z) = \frac{1}{2} \sum_{k=1}^K \alpha_k \sum_{p=1}^Z \sum_{q=1}^Z w_{pq}^{(k)} (r_p - r_q)^2 + \sum_{p=1}^Z (r_p - r_p^{init})^2 \quad (1)$$

where  $Z$  denotes the number of items in an initial list, and  $w_{pq}^{(k)}$  denotes the similarity between items  $p$  and  $q$  with respect to the  $k$ th modality. The first term of the loss function indicates the closeness of the items as measured using the different reranking modalities. The second term indicates the distance between a candidate reranked list and the initial list. Minimizing Eq. (1) produces a reranked list in which items similar with respect to the reranking modality are grouped together, but at the same time faithfulness to the initial list is retained. Since the loss function is convex w.r.t.  $r$ , it can be solved in a closed form – space constraints preclude presentation of the derivation here. Note that the  $\alpha_k$  are tradeoff parameters, which control the contributions from different modalities.

We adopt the self-contained collection-level modalities for reranking used by [6]. An exponential function is used to encode the similarity between items with respect to three different modalities:

$$w_{pq}^{(1)} = \exp(-\|Nr(p) - Nr(q)\|), \quad w_{pq}^{(2)} = \exp(-\|Ar(p) - Ar(q)\|), \quad w_{pq}^{(3)} = \exp(-\|NHR(p) - NHR(q)\|).$$

$Nr(\cdot)$  is the number of ratings for each item,  $Ar(\cdot)$  is the average rating of each item, and  $NHR(\cdot)$  is the number of highest ratings for each item.

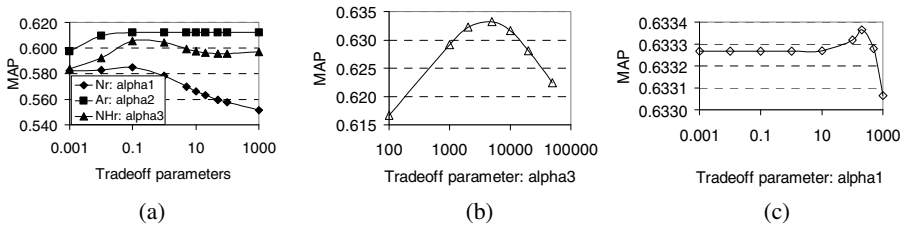
## 3 Experimental Evaluation

In this section, we provide details about our experimental setup and the results of a number of experiments we carried out to evaluate different aspects of the Multi-Rerank algorithm. The research questions that need to be answered are as follows. First, is Multi-Rerank generally effective for different CF approaches? Second, does Multi-Rerank outperform single-modality reranking?

The main experiments are performed the EachMovie (EM) data set (<http://www.cs.cmu.edu/~lebanon/IR-lab.htm>). The dataset consists of ca. 2.8 million ratings (integers scaled from 1 to 6) from 61265 users and 1623 items, where each user rated at least one item, and each item has been rated at least once. The sparseness of the dataset is around 97.2%.

We randomly select 10, 20 and 50 rated items for each user for training, i.e., with different user profile length (UPL), and use the remaining rated items in the user profile for testing. For each condition, users with less than 20, 30, or 60 rated items are removed in order to ensure we can evaluate on at least 10 rated items per user. We report the average performance attained across all users and ten runs of this procedure. Note that all the tradeoff parameters are tuned based on a validation set, which is another randomly split set under the condition of UPL=50. The validation set is mutually exclusive with the test sets. We consider the items that have been assigned high ratings by users to be relevant. Specifically, we regard the items rated by 5 or 6 as relevant in our experiments. Performance is reported in terms of the precision at top-N list (P@N) and mean average precision (MAP) [3]. Three well-known CF approaches are used for generating initial recommendation, i.e., item-based CF (ItemCF) [2] as a traditional CF approach, random walk with restarts (RWR) [4] and probabilistic matrix factorization (PMF) [5]. Note that model parameters that are involved in those approaches are also tuned to be optimal based on the validation set.

**Impact of Tradeoff Parameters.** We use the validation set to investigate impact of tradeoff parameters in Multi-Rerank. Due to the limit of space, we only demonstrate the impact of tradeoff parameters on ItemCF w.r.t MAP. Similar phenomena can be observed when reranking initial recommendations from RWR and PMF, and when using other metrics. As shown in Fig. 1(a), any of the three modalities can be used for improving initial recommendation from ItemCF, and the modality of *Ar* is more effective than the other two. Meanwhile, we can also observe the optimal tradeoff parameter for each single modality. In order to further investigate the impact of multiple modalities, we first fix the tradeoff parameter with the optimal value for the best single modality, i.e.,  $\alpha_2=50$  for *Ar* in this case, and then vary the tradeoff parameter for the second best single modality, i.e.,  $\alpha_3$  for *NHr* in this case. It can be seen from Fig. 1(b) that the second modality is capable of providing additional improvement over the best single modality. Similarly, when fixing the tradeoff parameters with the optimal values for the combination of *Ar* and *NHr*, we can further investigate the impact of the third modality, as shown in Fig. 1(c). The additional contribution from *Nr* is small, which is probably due to the fact the single modality of *Nr* is already the weakest. Note that it is not guaranteed that each modality is useful for improving any initial recommendation. We also find that the modality of *Nr* is not able to improve initial recommendations from RWR and PMF. In those cases, Multi-Rerank tunes the corresponding tradeoff parameters to 0.



**Fig. 1.** (a) The impact of tradeoff parameters for each modality on the performance of single-modality reranking ItemCF. (b) The impact of  $\alpha_3$  when the best single modality, i.e., *Ar*, is fixed with  $\alpha_2=50$ . (c) The impact of  $\alpha_1$  when the combination of the best two modalities, i.e., *Ar* and *NHr*, are fixed with  $\alpha_2=50$  and  $\alpha_3=5000$ .

**Performance.** In this subsection, we present the performance of Multi-Rerank when different CF approaches are used for initial recommendation and also for user profiles of different lengths. Results are shown in Table 1. Consistent with the results of [6], reranking with a single modality, achieves significant (Wilcoxon sign-rank significance test with  $p < 0.05$ ) improvement over initial recommendations: ca. 10% for ItemCF, ca. 5% for RWR, and ca. 3% for PMF. When reranking combines multiple modalities additional significant improvement (ca. 1%-4%) can be achieved over the performance achieved by using the best single modality. Note that the improvement is also consistent across all users with different profiles.

**Table 1.** The performance of Multi-Rerank for different CF approaches

	UPL=10		UPL=20		UPL=50	
	MAP	P@5	MAP	P@5	MAP	P@5
Initial: ItemCF	0.4876	0.4879	0.5349	0.5181	0.5704	0.5558
Multi-Rerank: +Nr	0.5170	0.5773	0.5580	0.5952	0.5857	0.6147
+Ar	0.5318	0.6151	0.5783	0.6441	0.6128	0.6791
+NHr	0.5347	0.6178	0.5761	0.6351	0.6070	0.6617
+Ar+NHr	0.5552	0.6699	0.6010	0.6966	0.6351	0.7270
+Ar+NHr+Nr	<b>0.5554</b>	<b>0.6713</b>	<b>0.6012</b>	<b>0.6982</b>	<b>0.6352</b>	<b>0.7282</b>
Initial: RWR	0.5791	0.6051	0.5837	0.6227	0.5802	0.6419
Multi-Rerank: +Ar	0.6037	0.6734	0.6067	0.6841	0.6033	0.6807
+NHr	0.6159	0.6592	0.6146	0.6601	0.6113	0.6682
+Ar+NHr	<b>0.6238</b>	<b>0.6830</b>	<b>0.6223</b>	<b>0.6866</b>	<b>0.6176</b>	<b>0.6888</b>
Initial: PMF	0.6249	0.6533	0.6385	0.6540	0.6985	0.7439
Multi-Rerank: +Ar	0.6451	0.7029	0.6602	0.7120	0.7100	0.7730
+NHr	0.6250	0.6533	0.6391	0.6540	0.6986	0.7440
+Ar+NHr	<b>0.6456</b>	<b>0.7036</b>	<b>0.6625</b>	<b>0.7172</b>	<b>0.7107</b>	<b>0.7757</b>

**Discussion and Conclusion.** We have presented Multi-Rerank, a novel reranking model for CF approaches. It exploits multiple low-cost self-contained modalities from the user-item matrix to improve recommendations generated by CF approaches. Experimental results from a large data set show that Multi-Rerank contributes to improving recommendation across various conditions, e.g., different initial recommendation approaches and different user profiles. The results of our experiments also demonstrate Multi-Rerank is able to exploit multiple modalities to achieve improvement beyond what is possible when only a single reranking modality is used.

**Acknowledgements.** PetaMedia Network of Excellence EC FP7 grant n° 216444.

## References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE* 17(6), 734–749 (2005)
2. Deshpande, M., Karypis, G.: Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems* 22(1), 143–177 (2004)



3. Herlocker, J., Konstan, J., Terveen, L.G., Riedl, J.: Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems* 22(1), 5–53 (2004)
4. Konstas, I., Stathopoulos, V., Jose, J.M.: On social networks and collaborative recommendation. In: *SIGIR 2009*, pp. 195–202 (2009)
5. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: *NIPS 2008*, vol. 20 (2008)
6. Shi, Y., Larson, M., Hanjalic, A.: Connecting with the collective: Self-contained reranking for collaborative recommendation. In: *Connected Multimedia 2010*, pp. 9–14 (2010)
7. Tian, X., Yang, L., Wang, J., Yang, Y., Wu, X., Hua, X.-S.: Bayesian video search reranking. In: *Multimedia 2008*, pp. 131–140 (2008)

# How Far Are We in Trust-Aware Recommendation?

Yue Shi, Martha Larson, and Alan Hanjalic

Multimedia Information Retrieval Lab, Delft University of Technology,  
Mekelweg 4, 2628CD Delft, Netherlands  
{y.shi,m.a.larson,a.hanjalic}@tudelft.nl

**Abstract.** Social trust holds great potential for improving recommendation and much recent work focuses on the use of social trust for rating prediction, in particular, in the context of the Epinions dataset. An experimental comparison with trust-free, naïve approaches suggests that state-of-the-art social-trust-aware recommendation approaches, in particular Social Trust Ensemble (STE), can fail to isolate the true added value of trust. We demonstrate experimentally that not only trust-set users, but also random users can be exploited to yield recommendation improvement via STE. Specific users, however, do benefit from use of social trust, and we conclude with an investigation of their characteristics.

**Keywords:** Recommender systems, social trust, trust-aware recommendation.

## 1 Introduction

Collaborative recommendation makes use of a user-item matrix to generate rating predications. Trust-aware recommendation [2-5] additionally exploits a social trust matrix encoding the *trust set* selected by each user. If users can be assumed to choose like-minded users for their trust sets, social trust is valuable source of additional information about user preference and stands to benefit recommendation significantly. Research on trust-aware recommendation [2-5] has focused on the Epinions dataset [5], the only publicly available dataset that contains both user-assigned ratings and user-selected trust sets. A recent state-of-the-art approach, Social Trust Ensemble (STE) [2], makes use of both social trust and the successful probabilistic matrix factorization (PMF) framework. The formulation of STE is expressed as follows:

$$L(U, V) = \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^n I_{ij}^R \left( R_{ij} - g \left( \alpha U_i^T V_j + (1 - \alpha) \sum_{k \in T(i)} S_{ik} U_k^T V_j \right) \right)^2 + \frac{\lambda_U}{2} \|U\|_F^2 + \frac{\lambda_V}{2} \|V\|_F^2$$

$U$  and  $V$  are the latent features of users and items,  $R$  is the user-item rating matrix,  $S$  is the user-user trust matrix and  $g(\cdot)$  is a logistic function.  $T(i)$  denotes the set of users trusted by  $i$ . The trade-off parameter,  $\alpha$ , balances the contribution of the target user with that of the users in the target user's trust set.  $\lambda_U$  and  $\lambda_V$  are regularization parameters, and  $I_{ij}$  is an indicator function, equal to 1 when  $R_{ij} > 0$  and 0 otherwise. Variations of STE have also been proposed, e.g., SoRec [4], which uses relational learning to factorize both user-item rating matrix and user-user trust matrix, and RWT [3], which uses user-user trust relationship as a regularization term in addition to PMF. All of those approaches focus on the added value from social trust for recommendation. However, more recent research started studying the influence of social relationships

on recommender systems yet further. Liu et al. [1] find that social friendship leads to hardly any significant improvement over a basic matrix factorization model. Shi et al. [7] find the long tail property of social networks can restrict the effectiveness of trust-aware recommendation. Here, by investigating a popular social recommendation approach, STE, we aim to uncover the basic value of social trust and identify users for which trust could be most helpful. Note that in this work we are investigating trust relationships established directly by users, rather than estimated by models as in [6]. Our first experiment compares trust-aware and trust-free approaches. Our second experiment compares the contributions of trust-sets to STE with the contributions of randomly chosen users. We conclude with an analysis of the characteristics of users that do benefit from social trust and an outlook.

## 2 Experimental Analysis and Discussion

We carry out a series of rating prediction experiments on the widely-used Epinions data [5], containing user ratings (scale of 1-5) and a user social trust network (user-selected trust sets). The user-item matrix contains ca. 665K ratings assigned by ca. 49K users to ca.140K items (sparseness of 99.9%). We adopt the experimental protocol used in [2]. Results are reported using Mean Absolute Error (MAE) averaged over five iterations with randomly selected test and training sets. We test two differently proportioned splits: 80%-train/20%-test (“Test20”) and 20%-train/80%-test (“Test80”). Coverage, the percentage of ratings that can be predicted by a given approach, cf. [5], is also reported. A good result has a low MAE and high coverage.

**Table 1.** Trust-free vs. trust-aware rating prediction

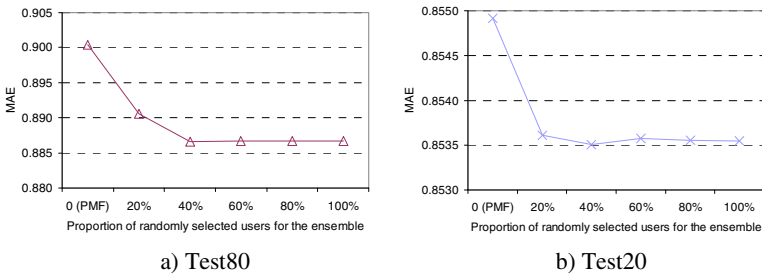
		Test80		Test20	
		Cov.	MAE	Cov.	MAE
<b>GlobAvRat</b>	no trust	100%	0.918	100%	0.917
<b>ItemAvRat</b>	no trust	74%	0.863	87%	<b>0.824</b>
<b>ItemAvRat+</b>	no trust	100%	<b>0.857</b>	100%	<b>0.824</b>
<b>TrustSetAvRat</b>	trust	10%	0.897	24%	0.852
<b>PMF</b>	no trust	100%	0.900	100%	0.855
<b>STE</b>	trust	100%	0.883	100%	0.850

Our first experiment, reported in Table 1, compares naïve, trust-free approaches and trust-aware approaches. The naïve “no trust” approaches exploit average ratings calculated on the user-item matrix: GlobAvRat uses the collection-wide average item rating across all users and all items (i.e., same prediction for every item). ItemAvRat uses the collection-wide average of the rating for a particular item (i.e., an item-specific prediction). Because there are a number of items that receive no user ratings (i.e., within the training set), ItemAvRat does not achieve full coverage, impeding a direct comparison with approaches that do. For this reason, we introduce a final trust-free approach, ItemAvRat+, which predicts with ItemAvRat and falls back to GlobAvRat in cases in which ItemAvRat does not exist. The trust-aware approaches make item-rating predictions for a user by incorporating information from that user’s trust set. TrustSetAvRat is the simplest trust-aware approach and uses the average rating assigned to an item by a user’s trust set as that item’s predicted rating. STE, the

state-of-the-art trust-aware approach described above that fuses influence from users’ trust sets with a PMF framework. For comparison, we also report predictions of plain (i.e., trust-free) PMF. We adopt the experimental settings of [2]; specifically, we use five latent dimensions for PMF and STE.

Note that our results in Table 1 are consistent with values reported in the literature, in particular PMF and STE in [2], and “TrustAll” in [5] (which is equivalent to ItemAvRat). Consistently with the observations of [2], STE out-performs both trust-free PMF and the simple trust-aware approach TrustSetAvRat. Surprisingly, however, there are two trust-free approaches, ItemAvRat and ItemAvRat+ (neither investigated by [2]) that beat both TrustSetAvRat and STE. These results cause us to question the extent to which improvement achieved by trust-aware recommendation can be directly attributed to the specific properties of social trust and lead to the question: *How far are we in social-trust-aware recommendation?* Specifically, we wonder whether trust-aware recommendation approaches have succeeded in isolating the added value of social trust, or whether they achieve improvement, at least in part, by other means.

We perform an experiment aimed at determining the specific value of social trust. We take STE and modify the trust set during the process of rating prediction. Specifically, we investigate the hypothesis that an arbitrary user from the collection is just as valuable as a member of the trust set for purposes of the ensemble. Effectively, we test a (ST)E algorithm, removing the social trust aspect of STE. We discard the users original trust set and instead sample users randomly from the general population.



**Fig. 1.** Performance of (ST)E when trust set is replaced with users drawn randomly from the general user population

It can be seen in Fig. 1 that for both the Test20 and Test80 splits, performance initially improves and then stabilizes. Surprisingly, even when no real trust is used, (ST)E still outperforms PMF (cf. Tab. 1). This result suggests that the original STE algorithm achieves at least part of its improvement by admitting, through the backdoor as it were, benefits unrelated to social trust. We believe that in STE integrating the trust set has the effect of introducing information about the collection with a background smoothing effect. However, (ST)E still does not achieve the performance level of the original STE (cf. Tab. 1), suggesting again that the trust set has specific properties important for recommendation.

In a final experiment, we take a closer look at the user-level differences between our best trust-free approach (ItemAvRat+) and our best trust-aware approach (STE). Focusing on the Test80 case, we divide users into two groups. We put users in the “no need for social trust” group (amounts to ca. 60% of users), if the trust-free approach

(ItemAvRat+) makes a better MAE prediction. We put users in the “need trust” group, (amounts to ca. 40% of users) if the trust-aware approach (STE) makes a better MAE prediction. We calculate some basic statistics of the users in these groups separately and report results in Table 2.

**Table 2.** User characteristics related to the usefulness of trust

Average user characteristics	no need	need trust
Number of ratings assigned by user	3.4	3.5
Size of user's trust set	11.7	12.5
Mean item rating assigned by user	3.99	4.09
Stan. dev. of ratings assigned by user	0.72	0.65

We notice that the two groups do not differ substantially with respect to how many ratings they assign or how many users they select to be in their trust sets, suggesting that user activity levels are not effective predictors of whether or not users benefit from trust-aware predictions. The mean of the rating users assign is also approximately the same for the two groups. However, the difference in the standard deviation of the ratings assigned by the two groups is relatively striking. One possible explanation is that the difference reflects the fact that different users focus on rating different categories of items (e.g., rate predominantly music vs. office supplies) and that collection level averages such as used by ItemAvRat+ make the best predictions for some categories (and therefore some users) but not for others.

**Conclusion.** We are apparently not very far along the road to effectively exploiting social trust in recommendation. Trust-free approaches beat state-of-the-art trust aware approaches such as STE. Moreover, the contribution of trust sets seems at least in part attributable to the contribution of a random user. In sum, effectively exploiting social trust for recommendation seems more difficult than expected, but our analysis supports its potential, given appropriate use of comparison with trust-free approaches.

**Acknowledgements.** The research leading to these results was carried out within the PetaMedia Network of Excellence and has received funding from the European Commission's 7th Framework Program under grant agreement n° 216444.

## References

1. Liu, N.N., Cao, B., Zhao, M., Yang, Q.: Adapting neighborhood and matrix factorization models for context aware recommendation. In: CAMRa 2010, pp. 7–13 (2010)
2. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: SIGIR 2009, pp. 203–210 (2009)
3. Ma, H., Lyu, M.R., King, I.: Learning to recommend with trust and distrust relationships. In: RecSys 2009, pp. 189–196 (2009)
4. Ma, H., Yang, H., Lyu, M.R., King, I.: SoRec: Social recommendation using probabilistic matrix factorization. In: CIKM 2008, pp. 931–940 (2008)
5. Massa, P., Avesani, P.: Trust-aware recommender systems. In: RecSys 2007, pp. 17–24 (2007)
6. O'Donovan, J., Smyth, B.: Trust in recommender systems. In: IUI 2005, pp. 167–174 (2005)
7. Shi, Y., Larson, M., Hanjalic, A.: Towards understanding the challenges facing effective trust-aware recommendation. In: RSWeb 2010, pp. 40–43 (2010)

# Re-ranking for Multimedia Indexing and Retrieval

Bahjat Safadi and Georges Quénot

Laboratoire d'Informatique de Grenoble  
BP 53 - 38041 Grenoble Cedex 9, France  
{Bahjat.Safadi, Georges.Quenot}@imag.fr

**Abstract.** We proposed a re-ranking method for improving the performance of semantic video indexing and retrieval. Experimental results show that the proposed re-ranking method is effective and it improves the system performance on average by about 16-22% on TRECVID 2010 semantic indexing task.

**Keywords:** Multimedia Indexing and Retrieval, Re-ranking.

## 1 Introduction

Semantic indexing and retrieval for multimedia databases has been a very active research field over the past few years. The global goal of multimedia indexing is to automatically describe documents containing images, sounds or videos. Nevertheless, allowing users to retrieve them within large collections or to easily navigate in these collections is still a very hard task.

Semantic indexing is generally achieved by supervised learning where a system is trained on positive and negative samples of a target concept (the development set) for producing a model which is then used for predicting the likeliness of new samples to contain this concept (the test set). This likeliness is often computed homogeneously to a probability for each data sample to contain the concept. Retrieval can then be done by ranking the samples according to the probability score. Such ranking is initially done with a score independently for each sample using only information from the development set. It is often possible to improve the indexing or retrieval performance by re-scoring the samples using the initial scoring information on the whole test collection. Recently, several ways of re-ranking methods have been proposed and developed, below we reviewed some of these methods.

Context fusion [12]: the results of different searching models (concept-based search model, text-based search model and query by example) are used to re-rank the ranked lists, in fact here the focus is on the fusion of different model outputs. This method needs to train new classifiers on new descriptors. Since we also use in our work the fusion of the outputs from multiple models, we took this as a baseline approach.

Classification-based re-ranking [3]: the initial results of a baseline system are used to discover the co-occurrence patterns between the target semantics and extracted features. This is very similar to “*learning to rank*” [4], which is based on training a ranking model which can precisely predict the ranking lists in the dataset. In [3] the authors used the top-ranked and bottom-ranked samples respectively, as pseudo-positive and pseudo-negative examples to train a new classification model for ranking, and the classification

margin for a target concept is regarded as its (new) re-ranked. The use of SVM as the classification model, leads to the method called *RankSVM*[4].

Ordinal re-ranking, in [6]: the author re-ranks an initial results by using the co-occurrence patterns via the ranking functions. The final score is the weighting combination of the original score and the re-ranked scores. They adopted a training method to train the Re-ranking algorithm on some concepts, and the re-ranking algorithm was applied to re-rank the remaining concepts.

In the case of video collections, the retrieval units are often not the whole videos themselves (which are generally too coarse grain for the user needs) but the video shots composing the videos. Our contribution in this paper, is to re-rank the video shots according to their scores, which were obtained from the classifiers, according to the video knowledge and nature. Our work is similar to the work in [5], where the authors re-ranked the previous results with video knowledge which is described as the mean score value of the current shots in the same video. The mean scores were adopted to be fused with the original scores.

The paper is organized as follows: Our re-ranking method is presented in section 2. Section 3 describes the experimental results, while section 4 presents our concluding remarks.

## 2 Re-ranking Method

In this paper, we propose an unsupervised method for re-ranking video shots according to a query or a concept. Our hypothesis is that videos have rather homogeneous contents and that the presence of a given concept in a video depends a lot on the nature of the video itself. Scores (here homogeneous to probabilities) are computed independently for all video shots as their likeliness to contain a target concept using classifiers (or networks of classifiers) that were trained on the development set. The re-ranking is actually done by a re-scoring which is done in two steps. First, we compute a global score for each video for containing the target concept; this score is computed from the scores of all the shots within the video. Then, we re-evaluate the score of each shot according to the global score of the video it belongs to.

The test collection contains a set of videos  $V = (v_1, v_2, \dots, v_m)$ ,  $m$  being the number of videos in the collection. Each video  $v_i$  composed of a sequence of shots  $v_i = (s_{i1}, s_{i2}, \dots, s_{in_i})$ ,  $n_i$  being the number of shots of  $v_i$ .

For each shot  $s_{ij}$ , an initial classification score  $x_{ij}$  is computed from supervised learning on the development set. Many options are possible for the computation of a global score  $x_i$  for a video  $v_i$  from the shots that it contains. We tried several formulas and found that the following one which is a generalization of the mean of the shot scores was the most effective:

$$z_i = \left( \frac{\sum_{j=1}^{n_i} (x_{ij})^\alpha}{n_i} \right)^{1/\alpha}, \quad (1)$$

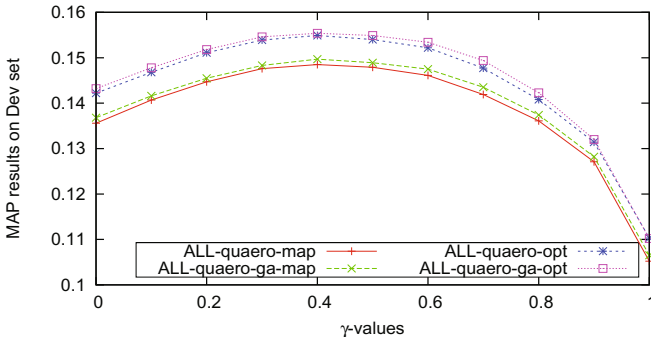
where  $\alpha$  is the parameter that has to be tuned by cross-validation within the development collection. Then, we update the score of each shot according to its previous score and the global score of the video it belongs to. Again, many options were possible and we chose a weighted multiplicative fusion:

$$x'_{ij} = x_{ij}^{1-\gamma} \times z_i^\gamma, \quad (2)$$

where  $\gamma$  is a parameter that controls the “strength” of the re-ranking. It also has to be tuned by cross-validation within the development collection.

### 3 Experiments

We conducted our experiments on TRECVID 2010, where 130 concepts are provided with ground truth labels in a training set. The evaluation is done by calculating the Mean Average Precision (MAP) on only 30 concepts that were chosen by NIST. We evaluated the re-ranking method on four different initial classification results, which we have submitted to TRECVID 2010, including different fusion strategies such as weighted and direct optimized weighted fusion, also the combination of the two fusion types with genetic fusion. These fusion strategies were applied on score vectors obtained by training different systems on 45 different descriptors (audio and visual descriptors) which have been produced by various partners of the IRIM project of the GDR ISIS [7].



**Fig. 1.** Tuning of the  $\gamma$  parameter: re-ranking the results after fusion with four different runs; MAP on the 130 concepts on the development set of TRECVID 2010

As it is shown, in order to get the best performance from our system, we need to tune the  $\gamma$  parameter in our re-ranking algorithm; this tuning was conducted on the development set using four different runs that had different fusion strategies. Fig. 1 shows the results of our re-ranking method on the development set using different values for  $\gamma$ . Each one of the curves indicates the MAP on the validation set after re-ranking the fused results with different values of  $\gamma$ . Here the baseline, for each run (curve), is given when  $\gamma = 0$ . As we can see, for the four runs, the best performance is reached when  $\gamma = 0.4$ ; here the re-ranking is applied directly on the fused results.

We made initial experiments with  $\alpha = 1$  (regular mean) and then tried other  $\alpha$  values. We found a small improvement with higher values with an optimal value close to 2, corresponding to a root mean square. We then applied the proposed method on the TRECVID 2010 test set with  $\gamma = 0.4$  and  $\alpha = 2$ . Table. 1 shows the gain on the MAP using our re-ranking method on four different initial scoring methods. As we can see, our proposed re-ranking method can significantly improve the performance: on this collection the gain is up to 22%.



**Table 1.** Results obtained on the test set; MAP on the 30 concepts of our four submitted results for TRECVID 2010

Run	Baseline	Re-rank	Gain(%)
ALL_quaero_map	0.0476	0.0577	21
ALL_quaero_ga_map	0.0479	0.0584	22
ALL_quaero_opt	0.0485	0.0563	16
ALL_quaero_ga_opt	0.0484	0.0568	17

## 4 Conclusion

We proposed a re-ranking method which improves the performance of semantic video indexing and retrieval. Experimental results show that the proposed re-ranking method is effective and that it can improve the performance of our system on average by about 16-22% on TRECVID 2010 semantic indexing task.

## Acknowledgments

This work was partly realized as part of the Quaero Program funded by OSEO, French State agency for innovation.

## References

1. Jiang, W., et al.: Context-based concept fusion with boosted conditional random fields. In: IEEE ICASSP (2007)
2. Liu, J., Lai, W., Hua, X.S., Huang, Y., Li, S.: Video Search Re-Ranking via Multi-Graph Propagation. In: Proc. 15th Int'l Conf. Multimedia, pp. 208–217 (2007)
3. Kennedy, L., Chang, S.-F.: A reranking approach for context-based concept fusion in video indexing and retrieval. In: ACM CIVR, pp. 333–340 (2007)
4. Herbrich, R., Graepel, T., Obermayer, K.: Support vector learning for ordinal regression. In: ICANN, pp. 97–102 (1999)
5. Wang, F., Merialdo, B.: Eurecom at TRECVID 2009 High-Level Feature Extraction. In: TRECVID 2009 Workshop Agenda, Gaithersburg, MD USA (November 2009)
6. Yang, Y.-H., Hsu, W.H.: Video search reranking via online ordinal reranking. In: IEEE International Conference on Multimedia and Expo, pp. 285–288 (2008)
7. Gorisse, D., Precioso, F., Gosselin, P., Granjon, L., Pellerin, D., Rombaut, M., Bredin, H., Koenig, L., Lachambre, H., Khoury, E.E., Vieux, R., Mansencal, B., Zhou, Y., Benois-Pineau, J., Jégou, H., Ayache, S., Safadi, B., Tong, Y., Thollard, F., Quénot, G., Benoît, A., Lambert, P.: IRIM at TRECVID 2010: High Level Feature Extraction and Instance Search. In: TREC Video Retrieval Evaluation Workshop, Gaithersburg, MD USA (November 2010)

# Combining Query Translation Techniques to Improve Cross-Language Information Retrieval

Benjamin Herbert, György Szarvas\*, and Iryna Gurevych

Ubiquitous Knowledge Processing Lab, Computer Science Department,  
Technische Universität Darmstadt,  
Hochschulstr. 10, D-64289 Darmstadt, Germany

<http://www.ukp.tu-darmstadt.de>

**Abstract.** In this paper we address the combination of query translation approaches for cross-language information retrieval (CLIR). We translate queries with Google Translate and extend them with new translations obtained by mapping noun phrases in the query to concepts in the target language using Wikipedia. For two CLIR collections, we show that the proposed model provides meaningful translations that improve the strong baseline CLIR model based on a top performing SMT system.

## 1 Introduction

Multilingual information search becomes increasingly important due to the growing amount of online information available in non-English languages and the rise of multilingual document collections. Query translation for CLIR became the most widely used technique to access documents in a different language from the query. As CLIR is less accurate than monolingual IR, the combination of query translation techniques is a promising way to approximate monolingual accuracy. Despite the importance of the task, previous combination approaches showed limited success. Combination of statistical machine translation (SMT), machine readable dictionary (MRD) based models or similarity thesauri (ST) proved to be difficult [1] due to the difference in the accuracy of individual models (SMT tends to be superior); the aggregation of translation errors; or the topic drift caused by integrating multiple translations in a single query. Studies that report successful combination of different models require substantial extra computation and resources (syntactic analysis and NP translation patterns [2]).

For query translation, one can i) use an online translation service; ii) train an SMT system using parallel corpora; iii) employ MRDs to translate query words; or iv) make use of large scale multilingual knowledge sources like Wikipedia for cross-lingual mapping. CLIR based on information in Wikipedia [5] can reach 60-70% of monolingual accuracy, while using Google Translate is reported to reach 90% of the accuracy of monolingual search [4]. Other approaches usually perform in between, e.g. [3]. In this study, we develop translation methods that are simple and accurate to be good candidates for extending a high performance SMT system in a realistic search scenario.

---

\* On leave from Research Group on AI of the Hungarian Academy of Sciences.

## 2 Translation Models

**Google Translate.** As a baseline CLIR model, we use query translation by Google Translate. Due to robustness across domains and strong performance in translating Named Entities, using Google Translate for CLIR achieved the best results in the recent CLIR evaluation at CLEF 2008 [4].

**Wikipedia based concept mapping.** Wikipedia provides a natural source of multilingual information, with redirects and cross-language links between articles in different languages. E.g., the phrase *German school system* maps to the German concept '*Bildungssystem in Deutschland*'. Wikipedia, being an encyclopedia, typically uses formal terminology which is less likely to be ambiguous and detrimental to retrieval performance than lay terms. To exploit this, we mine all redirect and cross-language links to build a translation table which maps concepts to their target language equivalent. This offers a reliable, but incomplete source of translation information (e.g. adjectives are seldomly contained in Wikipedia). To map queries to Wikipedia concepts (titles), we first try to map the whole query, and then gradually proceed with mapping shorter word sequences. Thus, the query *German Spelling Reform* is mapped as a single phrase, while *Nuclear Transport in Germany* is mapped to '*Nuclear transport*' and '*Germany*'.

## 3 Experimental Setup

**Document collections.** We used two CLIR collections introduced in the CLEF *Domain Specific (DS)* and *Ad Hoc (AH)* tracks. They consist of 151,319 German social science and 294,339 newspaper articles and were used in CLEF between 2003-2008 and 2001-2003. We used two times 75 queries for DS (2003-5 and 2006-8) and 100 + 60 queries for AH (2001-2 and 2003). We used the 2-3 words long title field as the query. For more details, see the CLEF website.

**Retrieval model.** For retrieval and query expansion via pseudo relevance feedback (PRF), we used Terrier's Okapi BM25 model and Bo1 term weighting method, with their default parameters. We tokenized the queries and documents, removed stopwords and used stemming (with SnowBall). Since German is a compounding language and decompounding can add further, less specific terms to enrich a query (e.g. *Milchkonsum (Milk consumption)* can be split to *Milch* and *Konsum*), we used a compound splitter for German (BananaSplit package).

**Combination of alternative translations.** To improve Google based CLIR, we add the phrases obtained from the Wikipedia-based concept mapping to the query. However, alternative terms for noun phrases can cause topic drift. For the query *Maternity Leave in Europe*, using two translations for *Maternity Leave* can cause documents that contain both to be ranked higher than those containing only one and the term *Europe*. To avoid this, we downweight further translations.

## 4 Experimental Results

Our results are summarized in Table 1. The CLIR models, using the concept mapping with Wikipedia, Google Translate and their combination are presented

for all four collection parts used, together with a monolingual retrieval run using German queries as reference. We provide the mean average precision (MAP) scores and the relative accuracies to the monolingual run, with just tokenization and stemming used (BASE), with compound splitting (CSPLIT) and with query expansion based on PRF (CS+QE). For the combined *WP + Google* runs, the weighting parameter used in queries was set to the optimal value on the other query set. That is, we used the DS 2003-2005 queries to determine the weight value used for the DS 2006-2008 query set, etc. For the BASE, CSPLIT and CS+QE configurations, the weight values were 0.3, 0.4, 0.2 for DS 2003-5; 0.4, 0.3, 0.2 for DS 2006-8; 0.5, 0.3, 0.6 for AH 2001-2; 0.4, 0.4, 0.2 for AH 2003, respectively.

**Table 1.** MAP values on different collections. Significant improvements (paired t-test,  $p < 0.05$ ) over the Google based models are marked with † for the *WP + Google* model.

Collection	Method	BASE		CSPLIT		CS+QE	
		MAP	% Monolingual	MAP	% Monolingual	MAP	% Monolingual
DS 2003-5	Wikipedia	0.2397	69.20	0.2501	62.74	0.2739	63.65
	Google Trans.	0.3304	95.38	0.3543	88.89	0.3844	89.33
	WP + Google	0.3562†	102.83	0.3753†	94.15	0.4034†	93.75
	Monolingual	0.3464	–	0.3986	–	0.4303	–
DS 2006-8	Wikipedia	0.1742	59.72	0.1740	52.41	0.1888	54.98
	Google Trans.	0.2878	98.66	0.3081	92.80	0.3293	95.89
	WP + Google	0.3204†	109.84	0.3194	96.20	0.3424†	99.71
	Monolingual	0.2917	–	0.3320	–	0.3434	–
AH 2001-2	Wikipedia	0.2193	82.04	0.2237	80.44	0.2611	79.58
	Google Trans.	0.2879	107.71	0.2886	103.78	0.3342	101.86
	WP + Google	0.2984†	111.63	0.2960†	106.44	0.3417	104.15
	Monolingual	0.2673	–	0.2781	–	0.3281	–
AH 2003	Wikipedia	0.1878	62.98	0.1902	55.84	0.2216	53.58
	Google Trans.	0.3166	106.17	0.3328	97.71	0.3904	94.39
	WP + Google	0.3339†	111.97	0.3487†	102.38	0.3937	95.19
	Monolingual	0.2982	–	0.3406	–	0.4136	–

As can be seen in Table 1, the individual CLIR models perform similar to the results reported in previous works: the Wikipedia model achieves 50-80% of the monolingual result, while Google Translate performs around 90% of the monolingual run. The Wikipedia based concept mapping performs slightly worse than the more complex WP model by [5] but we use just title fields.

As regards the combination of the Wikipedia based and the Google translations, we see consistent improvements over the CLIR models using a single translation. In particular, the combination improves over the results obtained using Google Translate which was argued to be a very strong CLIR model [4], and performs very close to the monolingual result. Moreover, these improvements are statistically significant (except for the AH collection when using QE and DS 2006-8 for CSPLIT). The positive effect of alternative translations is observed regardless of using compound splitting or not, which indicates that Wikipedia provided genuinely different translation terms. These improvements are in part complementary to query expansion, which indicates that the additional phrases are not always contained in the top retrieved documents and recovered by QE.

## 5 Conclusion and Future Work

In this study, we introduced a simple CLIR model using Wikipedia, mapping concepts in one language to their equivalents in another language based on the redirect and cross-language links in multilingual Wikipedia versions. This simple WP-based model performs similar to previous results obtained by Wikipedia-based CLIR (60-70% accuracy of monolingual retrieval). We also showed that the Wikipedia translations are accurate and often quite different from the translations of an SMT service, and are capable of improving the accuracy of an SMT-based CLIR model. In particular, to our knowledge we are first to show consistent (and in most settings significant) improvements over the CLIR based on Google Translate, which is reported to perform very well for CLIR.

There are many ways to improve our results. In particular, in the current study we used a single global term weighting parameter for Wikipedia translations. However, the benefit of WP translations is highly correlated with the coverage of the concept mapping for the given query (the higher portion of the query is mapped, the more beneficial it is), and with the average length of the mappings (concepts corresponding to longer phrases are more beneficial). This calls for a query-dependent weighting scheme, which we just started to develop. Another promising future work is to improve the coverage of our concept mapping by exploiting further information in Wikipedia (e.g. anchor texts [\[6\]](#)), or by adding a complementary resource to map verbs and adjectives.

## Acknowledgements

This work was supported by the German Ministry of Education and Research under grant 'Semantics- and Emotion-Based Conversation Management in Customer Support (SIGMUND)', No. 01ISO8042D, and by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program (grant No. I/82806).

## References

1. Braschler, M.: Combination approaches for multilingual text retrieval. *Information Retrieval* 7(1-2), 183–204 (2004)
2. Gao, J., Nie, J.Y., Xun, E., Zhang, J., Zhou, M., Huang, C.: Improving query translation for cross-language information retrieval using statistical models. In: *Proceedings of SIGIR*, pp. 96–104. ACM, New York (2001)
3. Hedlund, T., Airio, E., Keskustalo, H., Lehtokangas, R., Pirkola, A., Järvelin, K.: Dictionary-based cross-language information retrieval: Learning experiences from CLEF 2000–2002. *Information Retrieval* 7(1-2), 99–119 (2001)
4. Kürsten, J., Wilhelm, T., Eibl, M.: The Xtrieval framework at CLEF 2008: domain-specific track. In: Peters, C., Deselaers, T., Ferro, N., Gonzalo, J., Jones, G.J.F., Kurimo, M., Mandl, T., Peñas, A., Petras, V. (eds.) *CLEF 2008*. LNCS, vol. 5706, pp. 215–218. Springer, Heidelberg (2009)
5. Nguyen, D., Overwijk, A., Hauff, C., Trieschnigg, R.B., Hiemstra, D., Jong de, F.M.G.: WikiTranslate: Query Translation for Cross-lingual Information Retrieval using only Wikipedia. In: *Proceedings of CLEF*, pp. 58–65 (2009)
6. Roth, B., Klakow, D.: Cross-language retrieval using link-based language models. In: *Proceedings of SIGIR*, pp. 773–774. ACM, New York (2010)

# Back to the Roots: Mean-Variance Analysis of Relevance Estimations

Guido Zuccon, Leif Azzopardi, and Keith van Rijsbergen

School of Computing Science, University of Glasgow  
{guido,leif,keith}@dcs.gla.ac.uk

**Abstract.** Recently, mean-variance analysis has been proposed as a novel paradigm to model document ranking in Information Retrieval. The main merit of this approach is that it diversifies the ranking of retrieved documents. In its original formulation, the strategy considers both the mean of relevance estimates of retrieved documents and their variance. However, when this strategy has been empirically instantiated, the concepts of mean and variance are discarded in favour of a point-wise estimation of relevance (to replace the mean) and of a parameter to be tuned or, alternatively, a quantity dependent upon the document length (to replace the variance). In this paper we revisit this ranking strategy by going back to its roots: mean and variance. For each retrieved document, we infer a relevance distribution from a series of point-wise relevance estimations provided by a number of different systems. This is used to compute the mean and the variance of document relevance estimates. On the TREC Clueweb collection, we show that this approach improves the retrieval performances. This development could lead to new strategies to address the fusion of relevance estimates provided by different systems.

## 1 Introduction

In recent works, mean-variance analysis has been proposed to address the problem of document ranking in Information Retrieval [1,2,3,4]. This proposal led to the introduction of a new document ranking strategy, known as Portfolio Theory for Information Retrieval, that aims to balance the mean of estimations of (probability of) document relevance and their variance. However, while the theoretical model considers that a number of relevance estimations are available for each document and it exploits variances to revise a document ranking; its empirical instantiations refrain to use the actual mean and variance of such estimations (e.g. [1,2]). This is because the function used to estimate document relevance does not provide a probability distribution, but rather assigns to each document a single point-wise estimation of that probability. In the latter situation, there is no sense in computing a mean or a variance. The reason being, the mean of a single point-wise estimation is just equal to the estimation itself, while the variance is null. Thus, in previous works, mean and variance are discarded. In particular, the mean is replaced by the probability estimation of relevance itself. While, the variance is treated either as one of the parameters of the model, and

thus needs to be estimated, or as a quantity dependent on document length. Furthermore, the ranking function obtained employing mean-variance analysis also depends upon the covariance between relevance estimates of different documents. Once again, as no probability distribution over a document is considered, the covariance has scarce meaning. Covariance is then usually substituted by the correlation between documents term vectors.

In this paper, we revise the use of mean-variance analysis for ranking documents. In contrast with previous attempts, we instantiate the ranking strategy derived from the mean-variance analysis by considering a (discrete) relevance distribution associated to each retrieved document. This gives us the chance to actually compute the mean of the relevance estimates, their variances and the covariances between relevance estimates of documents. We show that this approach improves the performances of the retrieval system with respect to the ranking that would be obtained by considering a naïve ensemble of the mean of relevance estimates. This finding opens up new scenarios for the use of mean-variance analysis within Information Retrieval.

## 2 Mean-Variance Analysis Applied to Document Ranking

In [12], Wang et al. have shown how a document ranking strategy can be derived from the framework of mean-variance analysis. In particular, they suggest that a document should be placed at a particular rank position if this choice maximises the overall relevance (which is represented by the mean) of the ranked list at a given risk level (represented by the variance). We revise this paradigm, by assuming that a relevance distribution is assigned to each retrieved document. Given the relevance distribution of a document, indicated with  $r_d(x) = r_d(1), \dots, r_d(n)$ , both the mean,  $E[r_d]$ , and the variance,  $\text{var}[r_d]$ , can be computed as

$$E[r_d] = \sum_{i=1}^n \frac{r_d(i)}{n} \qquad \text{var}[r_d] = E[r_d^2] - (E[r_d])^2$$

Furthermore, given two relevance distributions  $r_d(x)$  and  $r_{d'}(x)$ , associated with documents  $d$  and  $d'$  respectively, the covariance between the two distribution is defined as

$$\text{cov}[r_d, r_{d_i}] = E[r_d, r_{d_i}] - E[r_d]E[r_{d_i}]$$

Given these definitions, we can derive a ranking strategy over documents, similarly to the one proposed in [12]. In particular, if  $b$  is a parameter depending on the predilection (or aversion) of the user towards risk, and  $w_i$  a weight associated to the importance to the rank position  $i$ , documents can be ranked according to the mean-variance analysis paradigm. At each rank position, document  $d^*$  is selected if it satisfies the following condition:

$$d^* = \arg \max_d \left( E[r_d] - bw_n \text{var}[r_d] - 2b \sum_{i=1}^{n-1} w_i \text{cov}[r_d, r_{d_i}] \right) \qquad (1)$$

Although the obtained formula appears to be equivalent to the one introduced in [12], it is fundamentally different. In fact, they (1) assume a point-wise relevance estimation, (2) conceptually substitute the variance in the estimations obtained for a document with the variance of the scores of the documents already ranked, and (3) approximate covariance between the relevance distributions of documents in terms of correlation between documents features. In this paper, instead, the computations of mean and variance are referred to the (discrete) relevance distribution of a document, and similarly covariance is computed between distributions associated to different documents.

### 3 Experiments

We empirically tested the ranking approach based on mean-variance analysis outlined in the previous section. To do so, we employed the TREC Clueweb (part B) corpus and the topics used in the TREC 2009 Web track [5].

**Deriving relevance distributions.** In order to empirically investigate our revision of the mean-variance analysis paradigm for document ranking, a relevance distribution has to be derived for each retrieved document. We use the retrieval runs submitted to the TREC 2009 Web (ad-hoc) task (part B only). For each of the 50 queries used in TREC 2009 Web track, there were 34 rankings provided by as many retrieval systems (or variations of a common underlying system). Thus, for each document that has been retrieved by one of the participating systems, there are at maximum 34 relevance estimates. We used these estimates to form the relevance distribution for each document, after having normalised the sum of the scores of each document ranking to one. Note that a document might have been retrieved by a system, but not by any other. We thus discarded all the documents retrieved by only one system, as their variance would be null.

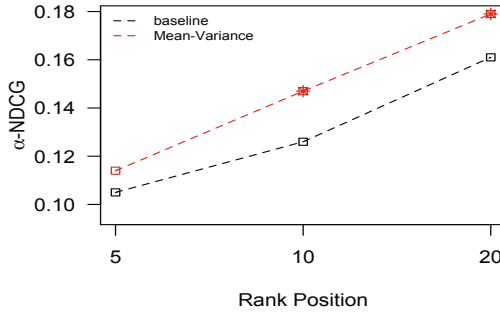
**Experimental methodology.** We compare our ranking approach (fed with the relevance distributions obtained from the TREC 2009 Web submissions) against the results obtained by a naïve strategy that re-ranks documents in decreasing order of the mean of their relevance estimations. We evaluate the obtained rankings in terms of MAP, MRR, precision at 10 (p@10), and  $\alpha$ -NDCG, to assess both the ability of the proposed approach to improve the relevance and the diversity of the rankings.

**Results and analysis.** In table 1 we report the performances in terms of MAP, MRR and P@10 obtained by the compared approaches. Similarly, figure 1 shows performances in terms of  $\alpha$ -NDCG at three different ranking depths. For the mean-variance analysis results, we let the parameter  $b$  vary in the range  $[-300, 300]$ , and we selected the run that provided best retrieval performances in terms of MAP and  $\alpha$ -NDCG@10. The results evidence that when both traditional measures (with  $b$  set to  $-250$ ) and diversity measures (with  $b = -85$ ) are considered, the mean-variance analysis strategy provides better ranking than just considering the mean of the relevance estimates. Furthermore, the improvements are statistically significant (except when considering  $\alpha$ -NDCG@5).



**Table 1.** Values of MAP, MRR and P@10 for the compared ranking approaches, when optimised for MAP. Statistical significance using t-test ( $p < 0.05$ ) is indicated with \*.

Measure	Baseline	Mean-Var
MAP	0.0472	0.0533*
MRR	0.2194	0.3377*
p@10	0.1380	0.2000*



**Fig. 1.** Values of  $\alpha$ -NDCG at various rank positions for the compared ranking approaches, when optimised for  $\alpha$ -NDCG@10. Statistical significance using t-test ( $p < 0.05$ ) is indicated with \*.

## 4 Conclusions

In this paper we have revisited the idea of using mean-variance analysis for document ranking, by considering the relevance distributions associated to documents retrieved by Information Retrieval systems. The outcome is a ranking function that provides a highly effective aggregation of document rankings. As a result, this approach could be applied to data fusion [6]. Further work will be directed towards: (1) investigating the robustness of the mean-variance approach under various conditions, and (2) comparing the proposed approach against state-of-the-art data fusion strategies.

*Acknowledgments.* The authors would like to thank the anonymous reviewers for their comments and P. Bruza, M. Sanderson, F. Scholer, and P. Thomas for useful discussions during early stages of this research. This work has been partially supported by the EPSRC Renaissance project (EP/F014384/1) and the Jim Gatheral Scholarship.

## References

1. Wang, J.: Mean-Variance Analysis: A New Document Ranking Theory in Information Retrieval. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 4–16. Springer, Heidelberg (2009)

2. Wang, J., Zhu, J.: Portfolio Theory of Information Retrieval. In: SIGIR 2009, pp. 115–122 (2009)
3. Aly, R.B.N., Aiden, D., Hiemstra, D., Smeaton, A.: Beyond Shot Retrieval: Searching for Broadcast News Items Using Language Models of Concepts. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 241–252. Springer, Heidelberg (2010)
4. Collins-Thompson, K.B.: Robust Model Estimation Methods for Information Retrieval. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA (2008)
5. Clarke, C.L.A., Craswell, N., Soboroff, I.: Overview of the TREC 2009 web track. In: Proc. of TREC 2009 (2009)
6. Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D., Frieder, O., Goharian, N.: Fusion of Effective Retrieval Strategies in the same Information Retrieval System. *J. Am. Soc. Inf. Sci. Tech.* 55(10), 859–868 (2004)

# A Novel Re-ranking Approach Inspired by Quantum Measurement

Xiaozhao Zhao<sup>1</sup>, Peng Zhang<sup>2</sup>, Dawei Song<sup>2</sup>, and Yuexian Hou<sup>1</sup>

<sup>1</sup> School of Computer Sci. & Tec., Tianjin University, China

<sup>2</sup> School of Computing, The Robert Gordon University, United Kingdom  
{0.25eye,krete1941}@gmail.com, {p.zhang1,d.song}@rgu.ac.uk

**Abstract.** Quantum theory (QT) has recently been employed to advance the theory of information retrieval (IR). A typical method, namely the Quantum Probability Ranking Principle (QPRP), was proposed to re-rank top retrieved documents by considering the inter-dependencies between documents through the “quantum interference”. In this paper, we attempt to explore another important QT concept, namely the “quantum measurement”. Inspired by the photon polarization experiment underpinning the “quantum measurement”, we propose a novel re-ranking approach. Evaluation on several TREC data sets shows that in ad-hoc retrieval, our method can significantly improve the first-round ranking from a baseline retrieval model, and also outperform the QPRP.

## 1 Introduction

Following van Rijsbergen’s pioneering work [3], which shows the potential of quantum theory (QT) in IR, a Quantum Probability Ranking Principle (QPRP) [5] was recently proposed. QPRP captures the inter-document dependencies in the form of “quantum interference”. In this paper, we aim to explore another important concept of the quantum theory, i.e., the “quantum measurement”. The photon polarization [2] is one of the key experiments that support the explanation of quantum measurement. Briefly, after a couple of polarization filters are inserted between the light source (which generates the photons) and a screen, the amount of light finally on the screen can be well explained by the quantum rather than classical measurement [2].

This inspires us to make an analogy of photon polarization in IR. We view the documents as photons generated from the source, and the retrieval process as measuring all the documents by the query polarization filter. In the first-round retrieval, only the query measurement ( $q$ -measure for short) is involved to measure the initial state of each document and yield the relevance probability. In order to re-rank the retrieved documents, we insert a  $t$  polarization filter with  $t$ -measure, which measures the document relevance with respect to topmost documents. The intuition is that usually the topmost (e.g. top 5) documents are more likely to be relevant. After the  $t$ -measure, the states of documents are changed, implying their relevance probabilities with respect to the query are revised. Based on the above ideas, we propose a novel re-ranking approach, called Quantum Measurement inspired Ranking model (QMR).

## 2 Quantum Measurement Inspired Ranking (QMR)

### 2.1 Introduction to Quantum Measurement

We first introduce the basic quantum measurement used in the photon polarization experiment. Please refer to [2] for the complete description for this experiment. A photon's polarization state can be modeled by a unit vector pointing to an appropriate direction. Specifically, the quantum state of any arbitrary polarization can be represented by a linear combination  $a|\uparrow\rangle + b|\rightarrow\rangle$  of two orthonormal basis vectors  $|\uparrow\rangle$  (vertical polarization) and  $|\rightarrow\rangle$  (horizontal polarization), where the amplitudes  $a$  and  $b$  are complex numbers such that  $|a|^2 + |b|^2 = 1$ . The quantum measurement on a state transforms the state into one of the measuring device's associated orthonormal basis. The probability that the state is measured by a basis vector is the squared magnitude of the amplitude in the direction of the corresponding basis vector. For example, a state  $\varphi = a|\uparrow\rangle + b|\rightarrow\rangle$  is measured by  $|\uparrow\rangle$  with probability  $|a|^2$ , and by  $|\rightarrow\rangle$  with probability  $|b|^2$ . After the measurement of  $|\uparrow\rangle$ , the state  $\varphi$  will collapse to  $a|\uparrow\rangle$ . Similarly, after the measurement of  $|\rightarrow\rangle$ ,  $\varphi$  will collapse to  $b|\rightarrow\rangle$ .

### 2.2 Our Proposed Model

We define the initial quantum state of any document  $d$  as:

$$\varphi_d = \alpha_d |1\rangle + \beta_d |0\rangle \quad (1)$$

where  $|\alpha_d|^2 + |\beta_d|^2 = 1$ , the state  $|1\rangle$  denotes the relevance basis, and the state  $|0\rangle$  denotes the non-relevance basis with respect to the given query  $q$ . For the  $q$ -measure,  $\varphi_d$  is measured by  $|1\rangle$  and then yields  $d$ 's relevance probability  $|\alpha_d|^2$ .

In the first-round retrieval, only the  $q$ -measure is involved to compute the document relevance. Therefore, we have  $|\alpha_d|^2 = p(d|q)$  and  $|\beta_d|^2 = 1 - p(d|q)$ , where  $p(d|q)$  is the relevance probability returned by a retrieval function.

To re-rank the documents, we introduce the  $t$ -measure, which measures any document  $d$  with respect to the topmost documents. Specifically, assume a top document  $d_t$  has its quantum state  $\varphi_{d_t} = \alpha_{d_t} |1\rangle + \beta_{d_t} |0\rangle$ , where  $|\alpha_{d_t}|^2 = p(d_t|q)$ . We are interested in the state of document  $d$  after measured by  $\varphi_{d_t}$  (i.e.  $t$ -measure). Accordingly, the following equation needs to be solved:

$$\varphi_d = \alpha_d |1\rangle + \beta_d |0\rangle = \lambda \varphi_{d_t} + \mu \varphi_{d_t}^{-1} \quad (2)$$

where  $\varphi_{d_t}^{-1}$  is orthonormal to  $\varphi_{d_t}$ . After formal calculations, we have

$$|\lambda| = |\alpha_d \alpha_{d_t} + \beta_d \beta_{d_t}| \quad (3)$$

After the  $t$ -measure,  $\varphi_d$  will collapse to the direction of  $\varphi_{d_t}$  and become

$$\varphi_d^t = \lambda \varphi_{d_t} = \lambda \alpha_{d_t} |1\rangle + \lambda \beta_{d_t} |0\rangle \quad (4)$$

where  $\varphi_d^t$  is the state vector of document  $d$  after the  $t$ -measure. Now, in order to obtain  $d$ 's relevance probability with respect to the query,  $d$ 's current state  $\varphi_d^t$  is then measured by  $q$ -measure, and the probability on the relevance basis  $|1\rangle$  is

$$p(d|d_t, q) = |\lambda \alpha_{d_t}|^2 \quad (5)$$

This shows that the revised relevance probability for the document  $d$  is  $p(d|d_t, q)$ . If  $d = d_t$ , then  $|\lambda| = 1$  and  $p(d|d_t, q) = |\alpha_{d_t}|^2 = p(d_t|q)$ , which means that  $d_t$ 's relevance probability is unchanged after the  $t$ -measure.

The above  $t$ -measure only considers one topmost document. If we consider  $k$  (e.g. 5) topmost documents, denoted as a set  $T$ , the revised relevance probability of a document  $d$  can be formulated as :

$$p(d|T, q) \propto \sum_{d_t \in T} p(d|d_t, q) \text{sim}(d, d_t) \tag{6}$$

where the  $\text{sim}(d, d_t)$  is the similarity between the document  $d$  and  $d_t$ , which indicates the importance of the  $t$ -measure with respect to the corresponding  $d_t$ . In our approach, the revised relevance probabilities by Eq. 6 are used to re-rank the documents. This is different from QPRP, in which the revised relevance probability is the sum of the original probability and the interference term. In addition, our QMR uses the inter-document similarity to indicate the weight of the corresponding  $t$ -measure, while QPRP integrates the inter-document similarity into the interference term.

### 3 Empirical Evaluation

Experiments are constructed on four TREC collections: WSJ87-92 (with topics 151-200), AP88-89 (with topics 151-200), ROBUST2004 (with topics 601-700) and WT10G (with topics 501-550). The title field of topics are used as queries. Lemur toolkit 4.7 is used for indexing and retrieval. All collections are stemmed using Porter stemmer with standard stop words removed during indexing.

The first-round retrieval is carried out by the query-likelihood (QL) model [4]. The smoothing method for document language model is the Dirichlet prior with the fixed  $\mu = 700$ . QL is set as the baseline method. The top  $n$  retrieved documents by the QL are involved in the re-ranking process. The normalized QL scores of these retrieved documents are used to indicate the relevance probabilities, i.e.,  $p(d|q)$ . We report the rank performance of top  $n = 50$  and  $n = 70$  documents, while we have similar observations when  $n = 30$  and  $n = 90$ .

The aim of this evaluation is to test the performance of two quantum-inspired re-ranking methods, i.e., QPRP and QMR. In QPRP, for the estimation of the interference term, we adopt  $\sqrt{p(d|q)p(d'|q)}\rho(d, d')$ , rather than  $-\sqrt{p(d|q)p(d'|q)}\rho(d, d')$  as used in [5], since the positive interference performs better in our experiments. In QMR, we adopt the Cosine function to measure the similarity between the  $tf \times idf$  vectors of two documents with the query words removed, the parameter  $k$  (the number of topmost documents used) is selected from  $\{5, 10\}$ , and the best performance is reported. We adopt Mean Average Precision (MAP) as the primary evaluation metric and the Wilcoxon signed-rank test as the statistical significance test method.

The Evaluation results are summarized in Table 1. We can observe that both QMR and QPRP achieve significant improvement over the QL in most cases. In addition, the proposed QMR outperforms the QPRP. This is possibly because that in QPRP, all the previously ranked documents are used to interfere with

**Table 1.** Evaluation Results on Top  $n$  Documents

MAP% (+chg%)	#Doc $n = 50$			#Doc $n = 70$		
	QL	QPRP	QMR	QL	QPRP	QMR
WSJ9872	21.57	22.57(+4.64)	23.77(+10.2*)	23.71	25.00(+5.44)	26.07(+9.95*)
AP8889	18.49	19.65(+6.27*)	20.74(+12.2*)	20.61	21.95(+6.50*)	23.20(+12.6*)
ROBUST04	22.78	24.26(+6.50*)	24.68(+8.34*)	24.28	26.13(+7.62*)	26.70(+9.97*)
WT10G	12.63	14.03(+8.83*)	14.32(+12.3*)	13.71	15.35(+11.9*)	15.56(+13.5*)

Significant improvements (at level 0.05) over QL are marked with \*.

the current document. On the other hand, in QMR, only the topmost (e.g. top 5) documents, which are more likely to be relevant, are involved to measure the current document.

## 4 Conclusion and Future Work

In this paper, we explore the application of the quantum measurement in the document re-ranking process and propose a novel re-ranking approach, called Quantum Measurement inspired Ranking (QMR). Evaluation results show that QMR can significantly improve the rank performance of top  $n$  documents retrieved by a typical language modeling approach. We also compared QMR with the recently proposed quantum interference based model QPRP, and results show that QMR outperforms the QPRP. In the future, we will compare QMR with other quantum inspired models, e.g. the Hilbert subspace based model in [1].

## Acknowledgments

This research is funded in part by the UK's EPSRC (Grant No: EP/F014708/2), the China's NSFC (Grant No: 61070044), the EU's Marie Curie Actions-IRSES (Grant No: 247590), and the NSFC of Tianjin, China (Grant No: 09JCY-BJC00200).

## References

1. Piwowarski, B., Frommholz, I., Lalmas, M., van Rijsbergen, C.J.: What can quantum theory bring to information retrieval. In: CIKM, pp. 59–68 (2010)
2. Rieffel, E.G., Polak, W.: An introduction to quantum computing for non-physicists. *ACM Comput. Surveys.* 32, 300–335 (2000)
3. van Rijsbergen, C.J.: *The Geometry of Information Retrieval*. Cambridge University Press, New York (2004)
4. Zhai, C., Lafferty, J.D.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: SIGIR, pp. 334–342 (2001)
5. Zuccon, G., Azzopardi, L.: Using the quantum probability ranking principle to rank interdependent documents. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) *ECIR 2010*. LNCS, vol. 5993, pp. 357–369. Springer, Heidelberg (2010)

# Simple vs. Sophisticated Approaches for Patent Prior-Art Search<sup>\*</sup>

Walid Magdy<sup>1</sup>, Patrice Lopez<sup>2</sup>, and Gareth J.F. Jones<sup>1</sup>

<sup>1</sup> CNGL, School of Computing, Dublin City University, Dublin 9, Ireland  
{wmagdy, gjones}@computing.dcu.ie

<sup>2</sup> INRIA & Humboldt Universitat zu Berlin, Germany  
patrice.lopez@inria.fr

**Abstract.** Patent prior-art search is concerned with finding all filed patents relevant to a given patent application. We report a comparison between two search approaches representing the state-of-the-art in patent prior-art search. The first approach uses simple and straightforward information retrieval (IR) techniques, while the second uses much more sophisticated techniques which try to model the steps taken by a patent examiner in patent search. Experiments show that the retrieval effectiveness using both techniques is statistically indistinguishable when patent applications contain some initial citations. However, the advanced search technique is statistically better when no initial citations are provided. Our findings suggest that less time and effort can be exerted by applying simple IR approaches when initial citations are provided.

## 1 Introduction

Prior-art search task in patent retrieval is concerned with finding all prior-art patents that are relevant to a patent application. Relevant prior-art patents have common technical aspects with a patent application, and include patents that can invalidate the novelty of the invention and patents that describe the state-of-the-art in the field of the invention on which the patent application is building [4, 5]. Identified relevant patents are cited in a search report which is part of the publication of the patent application. A typical patent application when filed to a patent office will include some initial patent citations describing the state-of-the-art. These citations are considered useful for patent examiners to understand the key aspects of an application and to start a search for relevant existing patents. However, large proportions of these initial citations are ultimately not found to be relevant, and are not included by patent examiners in the search report. Moreover, patent examiners usually identify a large amount of additional relevant patents.

Patent prior-art search task was addressed in the CLEF-IP task in both 2009 [5] and 2010 [4]. In 2010, relevant documents identified by the European Patent Office (EPO) in the search reports acted as the relevance set, and the initial patent application filed to the EPO acted as the topic [4]. The objective was to identify the

---

<sup>\*</sup> This research is partially supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (CNGL) project at Dublin City University.

relevant documents for each patent topic automatically. Submitted runs from two participants achieved considerably higher retrieval effectiveness than the other submitted runs [1, 3]. These two participants used IE techniques to extract the patent citations provided in the patent application. Later they utilized these citations in two different ways to improve the retrieval effectiveness. The participant group who achieved the best run used an advanced search approach for the retrieval process including key-term extraction, multiple retrieval models, multiple indexes, and post-ranking techniques [1]. The other participant group used a simpler IR technique [3].

In this paper, the best two runs in the CLEF-IP 2010 are revisited and compared after using the same extracted citations for both runs. The comparison was applied on the English topics and divided into two sets. The first set contains topics for which patent citations can be extracted from its text, and the second set contains topics that do not include any patent citations in their description. The comparison results show that when patent citations can be extracted from the text of a patent topic, the simple search approach achieves results comparable to the more sophisticated method. However, when no citation could be extracted from the patent topic, the advanced search approach achieves significantly better results. This finding suggests that using simpler approaches could be used effectively for patent search when applicants provide initial citations, which is the situation for more than half of the filed patents. Otherwise, following this hypothesis more complex approaches could be used to improve the retrieval effectiveness when no initial citations are provided.

## 2 Retrieval Methodologies in CLEF-IP 2010

Different retrieval methodologies were used in the 25 runs submitted for the CLEF-IP 2010 prior-art patent search task. Excluding the best two runs, the other 23 runs achieved retrieval effectiveness ranging from 0.007 to 0.14 MAP, and 0.01 to 0.21 PRES@100. PRES (Patent Retrieval Evaluation Score) is a new evaluation metric used in CLEF-IP 2010 [4] which emphasises the quality of the system in retrieving a larger portion of the relevant documents at relatively high ranks according to a user given cut-off ( $N_{max}$ ) [2]. The best two runs used a citation extraction methodology to achieve significantly higher scores. The second ranked run achieved 0.2 MAP and 0.32 PRES@100 while the first ranked run achieved 0.26 MAP and 0.39 PRES@100, which are considerably higher score than those for the other runs.

A comparison of the best two runs was carried out using 1348 English topics provided by the CLEF-IP 2010 to search a collection of 1.35M patents from the European Patent Office (EPO) [4]. The comparison is based on the same set of extracted citations, which are those extracted by the first participant [1]. This set is used because it included more citations, since it used additional external resources to improve the results by using patent family look-up [1]. The extracted patent citations included 7706 citations extracted from 728 topics. For the experiments, the 1328 topics were divided into two sets: 728 topics that have citations extracted from their text and 620 topics that have no citations.

### 2.1 Simple Search Approach

The approach presented in [3] uses a straightforward IR technique to retrieve a ranked document list, then appends it to the extracted citations list to create the final results



list. In this approach patent documents are treated as plain text neglecting their structure. The query is constructed from terms in the description section of the patent topic after filtering out terms that appeared once, in addition to bigrams (two consecutive terms) that appeared in the title and abstract sections of the patent topic more than one time. The Indri search toolkit was used for the retrieval process [6]. The retrieved results are then filtered based on the patent classification, where each patent document or topic has a classification according to the scope of the invention. This filtering process guarantees that the patent topic and the retrieved results share the same first three levels of classification [3]. The produced results list is then simply appended to the extracted citations after removing the duplicates.

## 2.2 Advanced Search Approach

The approach presented in [1] uses a more sophisticated retrieval method:

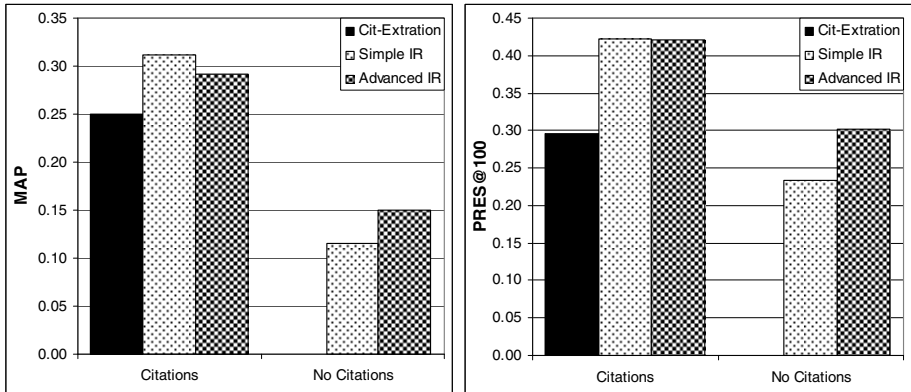
1. Creating a working set for each patent topic for pruning the search space. This working set is built recursively from patents which share common classification, inventor, or citations with the patent topic and extracted patent citations.
2. Applying multiple retrieval models (BM25 and Indri) using different indexes (English lemma, phrases, and concepts) for producing several sets of ranked results, and then merging them based on multiple SVM regression models and a linear combination of the normalized ranking scores.
3. Post-ranking the results based on an SVM model exploiting patent metadata.

This system used several complementary indexes, including phrase and conceptual indexes. The phrase index is based on the extraction of key terms from the patent topics using vast ranges of metrics and features. The conceptual index is based on a large scale database resulting from merging various terminological resources (METS, UMLS, the Gene Ontology, Wikipedia, etc.).

## 3 Results

Figure 1 shows the retrieval effectiveness of the simple and advanced IR techniques for the patent prior-art search task. Results are reported for the two topic sets when citations could and could not be extracted. The retrieval effectiveness when only extracted citations are used without any kind of IR is reported as a baseline. From Figure 1 it can be seen that the extracted citations achieve higher retrieval effectiveness than either IR approach when no citations could be extracted. Also it is clear that both systems achieved nearly double the performance level for topics that contain patent citations within their text. Comparing both systems on their performance, it can be seen that when citations exist, the simple IR approach is as effective as the more complex approach when compared using PRES and even better when compared by MAP. However, when no citations could be extracted as an initial step, the complex approach is significantly better. This observation leads to the hypothesis that when a patent application includes directly cited prior-art patents, simple search approaches are sufficient to achieve good retrieval results.

To further support this finding, the results list of the complex approach was appended to the extracted citations list after removing duplicates in the same way as the final step of the simple approach. This approach gave MAP of 0.3 and PRES@100 of 0.43 which



**Fig. 1.** Retrieval results for simple and complex IR approaches with CLEF-IP prior-art search task, when citations could be and could not be extracted

is statistically indistinguishable from the results for the simple approach shown in Figure 1.

## 4 Conclusion

In this paper, we have presented a comparison between two approaches for the patent prior-art search task. The first approach is characterized by its simplicity and low resources requirement, while the second one is more sophisticated, using an advanced level of content analysis. The results show that the simple search approach is as effective as the sophisticated one when initial citations are provided. This is the situation for 54% of the test collection used in our experiments, but also a legal requirement of the EPO (Rule 27(b) of the EPC, European Patent Convention). The observation that simple IR approaches can in many cases achieve similar results to current sophisticated ones, suggests that further investigation is required to better understand the retrieval process in patent prior-art search in order to develop more effective methods for this task.

## References

1. Lopez, P., Romary, L.: Experiments with citation mining and key-term extraction for Prior Art Search. In: Proceedings of CLEF 2010 (2010)
2. Magdy, W., Jones, G.J.F.: PRES: a score metric for evaluating recall-oriented information retrieval applications. In: SIGIR 2010 (2010)
3. Magdy, W., Jones, G.J.F.: Applying the KISS Principle for the CLEF-IP 2010 Prior Art Candidate Patent Search Task. In: Proceedings of CLEF 2010 (2010)
4. Piroi, F.: CLEF-IP 2010: Retrieval Experiments in the Intellectual Property Domain. In: Proceedings of CLEF 2010 (2010)
5. Roda, G., Tait, J., Piroi, F., Zenz, V.: CLEF-IP 2009: Retrieval experiments in the intellectual property domain. In: Peters, C., Di Nunzio, G.M., Kurimo, M., Mostefa, D., Penas, A., Roda, G. (eds.) CLEF 2009. LNCS, vol. 6241, pp. 385–409. Springer, Heidelberg (2010)
6. Strohan, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: Proceedings of ICIA (2004)

# Towards Quantum-Based DB+IR Processing Based on the Principle of Polyrepresentation

David Zellhöfer<sup>1</sup>, Ingo Frommholz<sup>2</sup>, Ingo Schmitt<sup>1</sup>, Mounia Lalmas<sup>3</sup>,  
and Keith van Rijsbergen<sup>2</sup>

<sup>1</sup> Brandenburg University of Technology Cottbus, Department of Computer Science  
Postfach 10 13 44, 03013 Cottbus, Germany

[david.zellhoefer@tu-cottbus.de](mailto:david.zellhoefer@tu-cottbus.de)

<http://dbis.informatik.tu-cottbus.de/>

<sup>2</sup> University of Glasgow, School of Computing Science  
18 Lilybank Gardens, Glasgow G12 8QQ, Scotland

[ingo.frommholz@glasgow.ac.uk](mailto:ingo.frommholz@glasgow.ac.uk)

<http://ir.dcs.gla.ac.uk/>

<sup>3</sup> Yahoo! Research, Spain

[mounia@acm.org](mailto:mounia@acm.org)

**Abstract.** The cognitively motivated principle of polyrepresentation still lacks a theoretical foundation in IR. In this work, we discuss two competing polyrepresentation frameworks that are based on quantum theory. Both approaches support different aspects of polyrepresentation, where one is focused on the geometric properties of quantum theory while the other has a strong logical basis. We compare both approaches and outline how they can be combined to express further aspects of polyrepresentation.

## 1 Introduction

The core idea behind the principle of polyrepresentation of documents [2] is that a document is defined by different representations that can be combined to determine the cognitive overlap where it is assumed highly relevant documents are likely to be contained. Examples of different representations of the same document are user-given ratings, reviews and comments, the author-given textual content or non-textual features of a multimedia document.

Inspired by van Rijsbergen's idea of applying the mathematics behind quantum theory for IR, seamlessly combining geometry, probability theory and logics [5], recent frameworks have approached polyrepresentation from different viewpoints. While in [1] a geometric framework is proposed which has a probabilistic interpretation, [9] comes from the database domain and has a quantum logic-based background [6], which can also be interpreted probabilistically. These approaches are complementary in the sense that they focus on different aspects of polyrepresentation on the one hand, and in its viewpoint of the underlying theory (geometrical vs. logic-based) on the other hand, with geometry as their common mathematical ground. Our intention in this study is therefore to learn from both approaches and figure out the potential we can gain from combining them.

## 2 Quantum-Based Frameworks for Polyrepresentation

Van Rijsbergen argues in his seminal work [5] in support of a quantum mechanic/logic-based interpretation of IR. He outlines how probability theory, logic, and geometry relate in the context of quantum theory. In the following subsections, we will discuss two approaches which are based on these findings. Both approaches reflect different aspects of the principle of polyrepresentation.

### 2.1 The IQIR Framework

The motivation behind the IQIR framework (Interactive Quantum-based IR) is to provide means for (interactive) information retrieval on the ground of quantum mechanics [1]. The framework is based on the assumption that there is an information need (IN) space, which is a real-valued Hilbert space. The user's IN is represented by a set of unit state vectors, reflecting the uncertainty the system has about the user's IN. The event that a document is relevant is modelled as a subspace. The probability of relevance is now determined by the squared length of the projections of the vectors onto the document subspace. To support polyrepresentation, a separate Hilbert space for each document representation is created. To determine the cognitive overlap, the single representation spaces and their state vectors are combined using the tensor product, which establishes a polyrepresentation space. Geometric means are provided to weight single representations according to their importance to the user and to control the cognitive overlap. Within the polyrepresentation space, a non-separable (or entangled) state expresses dependencies between document representations from the user's point of view. Another inherent feature of the framework is that the vectors representing the user's IN can dynamically be transformed to reflect several forms of user interaction and information need drifts. This is motivated by the fact that information needs are indeed dynamic by nature [2].

### 2.2 CQQL

In contrast to the aforementioned approach, the commuting quantum query language (CQQL) [6] models the IN as a subspace of a hypothetical real-valued Hilbert space on which documents, represented as vectors, will be projected in order to determine their probability of relevance. Again, polyrepresentation and the creation of the cognitive overlap is supported by combining the isolated Hilbert spaces, each describing the attributes of a single document representation, by means of the tensor product. Because of CQQL's background in DB theory, it differentiates between attribute values that will be modelled by orthogonal state vectors mirroring Boolean values and probability values that rely on non-orthogonal vectors. Although quantum logic itself does not form a Boolean algebra (because the law of distributivity is violated), the commuting projector describing the IN subspace is consistent with these rules. To guarantee this – generally speaking – CQQL restricts the query to not use more than one probability condition on one attribute, e.g.  $title \approx "polyrepresentation" \vee title \approx "quantum\ logic"$ . If this restriction is accepted, an IN can be modelled using the full structural power of a Boolean algebra, i.e. conjunctions, disjunctions, and

negations. Note that this does not mean the re-introduction of the shortcomings of Boolean retrieval models such as unordered result sets or the absence of term weighting. Instead, CQQL fully supports weighted logical connectors in order to adjust a query according to the user's needs [8], i.e, to construct the cognitive overlap. Given a structured query that models the cognitive overlap expressing the user's IN, the relevance of all documents can be assessed.

### 3 Probabilities, Geometry and Logics for Polyrepresentation – Opportunities and Challenges

Because of their conceptual similarities, it is tempting to combine both approaches into one quantum-based DB and IR query model that supports the principle of polyrepresentation. Regarding a polyrepresentative query model, IQIR lacks the opportunity to express highly structured queries. Such queries relying on Boolean connectors are shown to support the polyrepresentation principle best [3]. CQQL offers means to incorporate such queries into the query model. An interesting result from the utilisation of quantum theory for IR in IQIR is the potential to introduce concepts like non-separable (entangled) states reflecting inter-relationships between parts of a query, the representation of the system's uncertainty about the user's IN and the possibility to reflect user interaction by means of quantum measurement. The first idea is not supported in CQQL so far. CQQL addresses the latter by a machine-based learning relevance feedback approach in order to adjust the present condition weights that eventually determine the cognitive overlap [7,8].

Both approaches have in common that they rely on weights to steer the influence of different representations onto the cognitive overlap. IQIR offers so-called "don't care" aspects whereas CQQL equips all logical connectors with weights in order to personalise a query. Hence, both methods use – in a way – pseudo features to express different grades of importance for terms or parts of a query. A combination of both approaches would give us the possibility to create a framework that is able to reflect the system's uncertainty about the user's IN, react on IN drifts and represent interrelations between document representations, in combination with powerful querying mechanisms provided by a query language that supports concepts from databases and IR. This way, complex and possibly interactive retrieval strategies, as structured queries to given knowledge base of polyrepresented documents combining factual and content-oriented aspects, are supported.

One of the biggest open challenges to achieve this goal is the dual way documents and information needs are represented in both frameworks. In IQIR, the unit vectors represent the user's IN and subspaces represent documents, whereas in CQQL, these vectors describe one representation of a document and the user state (the query) is reflected by a projector/subspace. In CQQL, the document is considered the dynamic part that is measured against the projector, which resembles the viewpoint taken in [4]. Although somewhat dual, there is no standard way to translate one view into the other. A solution may be to regard the set of unit vectors in IQIR as an average or a probability distribution of projectors in CQQL. This may impose some restrictions on how we can model these

vectors; we may for a start just deal with the case that the user's IN is represented by one vector only. As both approaches rely on weights to incorporate the dynamics of the user's search goal and the weights in CQQL already affect the projector it seems appropriate to pursue this idea further.

## 4 Conclusion

In this study we analysed two polyrepresentation frameworks that are based on the mathematical formalism of quantum mechanics. We have shown that both approaches are complementary when it comes to polyrepresentation, with geometry as common mathematical grounds. A combination of both approaches would lead to a powerful theoretical and also practical framework for polyrepresentation, giving rise to supporting complex retrieval strategies. One of the biggest challenges is the dual nature of both approaches, which we are going to address in our future work.

## Acknowledgements

This research was supported by an Engineering and Physical Sciences Research Council grant (Grant Number EP/F015984/2). This paper was written when Mounia Lalmas was a Microsoft Research/RAEng Research Professor.

## References

1. Frommholz, I., Larsen, B., Piwowarski, B., Lalmas, M., Ingwersen, P., van Rijsbergen, K.: Supporting Polyrepresentation in a Quantum-inspired Geometrical Retrieval Framework. In: Proceedings of the 2010 Information Interaction in Context Symposium, pp. 115–124. ACM, New Brunswick (2010)
2. Ingwersen, P., Järvelin, K.: The Turn: Integration of Information Seeking and Retrieval in Context. Springer-11645 /Dig. Serial. Springer, Dordrecht (2005), <http://dx.doi.org/10.1007/1-4020-3851-8>
3. Larsen, B., Ingwersen, P., Kekäläinen, J.: The polyrepresentation continuum in IR. In: IiX: Proceedings of the 1st International Conference on Information Interaction in Context, pp. 88–96. ACM, New York (2006)
4. Melucci, M.: A basis for information retrieval in context. ACM Transactions on Information Systems (TOIS) 26(3) (2008), <http://portal.acm.org/citation.cfm?id=1361687>
5. van Rijsbergen, C.J.: The Geometry of Information Retrieval. Cambridge University Press, New York (2004)
6. Schmitt, I.: QQL: A DB&IR Query Language. The VLDB Journal 17(1), 39–56 (2008), <http://cat.inist.fr/?aModele=afficheN&cpsidt=20038185>
7. Zellhöfer, D.: Inductive User Preference Manipulation for Multimedia Retrieval. In: Böszörményi, L., Burdescu, D., Davies, P., Newell, D. (eds.) Proc. of the Second International Conference on Advances in Multimedia (2010)
8. Zellhöfer, D., Schmitt, I.: A Preference-based Approach for Interactive Weight Learning: Learning Weights within a Logic-Based Query Language. In: Distributed and Parallel Databases (2009), doi:10.1007/s10619-009-7049-4
9. Zellhöfer, D., Schmitt, I.: Approaching Multimedia retrieval from a Polyrepresentative Perspective. In: Proc. of the 8th International Workshop on Adaptive Multimedia Retrieval (2010) (to appear)

# ATTention: Understanding Authors and Topics in Context of Temporal Evolution

Nasir Naveed, Sergej Sizov, and Steffen Staab

Institute for Web Science and Technologies  
University of Koblenz-Landau, Germany  
{naveed, sizov, staab}@uni-koblenz.de

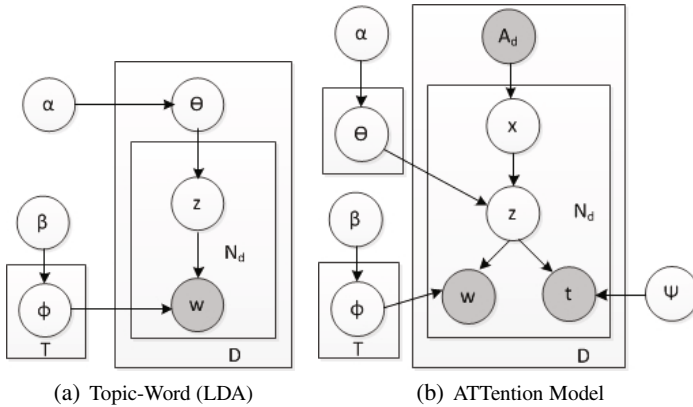
**Abstract.** Understanding thematic trends and user roles is an important challenge in the field of information retrieval. In this contribution, we present a novel model for analyzing evolution of user's interests with respect to produced content over time. Our approach ATTention (a name derived from analysis of Authors and Topics in the Temporal context) addresses this problem by means of Bayesian modeling of relations between authors, latent topics and temporal information. We also present results of preliminary evaluations with scientific publication datasets and discuss opportunities of model use in novel mining and recommendation scenarios.

## 1 Introduction and Background

The world wide web provides a platform for content sharing activities where people can share views, participate in discussions, publish technical domain specific blogs and research papers, thereby, contribute tremendous online contents related to different topics. It has been observed that topics discussed in collaborative social networks exhibit spikes (sudden topics, linked to current events, or enjoying a limited-time interest) and chatters (more recurring, long term topics) indicating strong correlation.

Consider a scenario where a user tries to track back a particular topic for its emergence, growth patterns, popularity and underlying key players contributing to it over a period of time. In such a scenario manual analysis of this tremendous amount of text for finding latent topics, capturing topic evolution, identifying the author's interests is expensive in terms of time and labor. Following this observation the challenge is to provide an approach which can help user in information filtering and finding users with similar interests. One such approach can also be helpful in finding the most influential authors at different stages of topic evolution and thus can be helpful in characterizing the authors as pioneers, mainstream or laggards in different subject areas.

To tackle the above mentioned challenges we exploited probabilistic methods which in recent years have proved to be very successful for modeling topics in document collections. One such method is Latent Dirichlet Allocation (LDA) [1]. LDA is a Bayesian multinomial mixture model which has become a popular method in text analysis due to its ability to produce interpretable and semantically coherent topics. It uses the Dirichlet distribution to model the distribution of the topics for each document. Each word is considered sampled from a multinomial distribution over words specific to this topic.



**Fig. 1.** Latent Dirichlet Allocation and ATTention Models for document content generation

LDA is a well-defined generative model and generalizes easily to new documents. Since LDA is highly modular and hierarchical, therefore, it can easily be extended. Many extensions to basic LDA model have been proposed to incorporate document metadata. In this type of model, each topic has a distribution over words as in the standard model, as well as a distribution over metadata values. Examples of such models include, the Topics over Time model [4], the Group-Topic model [5], the Author-Topic model [3], the Linked Topic and the Interest Model [2].

None of the given approaches models documents and author together with the temporal information. In this paper, we propose ATTention (a name derived from analysis of Authors and Topics in the Temporal context); a model of topic dynamics in social media which connect the temporal topic dependency with the social actors, thereby, providing an insight into the evolution of topics over time along with capturing the author interests for a given time period.

## 2 The ATTention Model

We extend LDA by incorporating document metadata i.e. author and timestamp of the document. Figure 1 is a graphical representation of LDA and ATTention. In the model each author is modeled as having distribution overs topics and each topic is modeled as having distribution over words. The ATTention model has three sets of unknown parameters; the author distribution over topics  $\theta$ , the topic distribution over words  $\phi$  and the topic distribution over time  $\psi$ . Both  $\theta$  and  $\phi$  have multinomial distributions with symmetric Dirichlet priors having the hyperparameters  $\alpha$  and  $\beta$  respectively. To avoid time discretization we use a continuous per-topic parametric Beta distribution  $\psi$  over absolute time values in the generative process, this gives a natural distribution of topics over time. We normalize the time-stamps to values between 0 and 1 for parameter estimation.

The generative process of the ATTention model which corresponds to the process used in Gibbs sampling for parameter estimation is described as follows.



1. Draw  $\theta \sim Dir(\alpha)$
2. Draw  $\phi \sim Dir(\beta)$
3. For each document  $d$ , pick an author from the list of authors  $a_d$  and draw a multinomial  $\theta_d$  from Dirichlet prior  $\alpha$ ; then for each of the  $N_d$  words,  $w_i$ ,
  - Draw a topic  $z_{d_i}$  from multinomial  $\theta_d$ ;
  - Draw a word  $w_{d_i}$  from multinomial  $\phi_{z_{d_i}}$ ;
  - Draw a timestamp  $t_{d_i}$  from Beta  $\psi_{z_{d_i}}$

In the ATTention model three parameters  $\theta$ ,  $\phi$ ,  $\psi$  are estimated. Exact inference of the parameters of LDA type models is intractable, therefore, we use Gibbs sampling to perform approximate inference. In the model there are three latent variables  $z$ ,  $a$  and  $t$ . Each set  $(z_i, a_i, t_i)$  of these latent variables is drawn as block conditioned on all other variables. We begin with the joint probability of dataset, and using the chain rule we obtain conditional probability for

$$p(z_i = j, x_i = k, t_i = l | w_i = m, z_{-i}, x_{-i}, t_{-i}, w_{-i}, a_d) \quad (1)$$

where  $z_i, x_i, t_i$  represent topic, author and time assigned to  $w_i$  whereas  $z_{-i}, x_{-i}, t_{-i}$  are all other assignments of that topic, author and time excluding the current assignment.  $w_{-i}$  represents all other words in the document set and  $a_d$  is the observed author of the document.

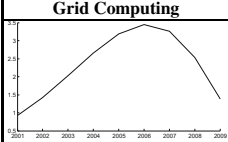
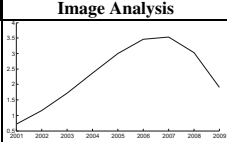
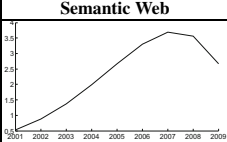
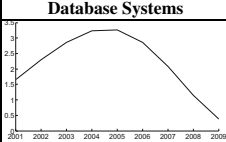
Learning joint probabilities of these three latent variables enables us to query the model conditioned on any combination of these variables using Baye's rule. For example given the author and time find the authors interest in that time period  $P(\phi_d | a, t)$  or given the topic and time find the top authors contributing to the topic in that time  $P(\theta_d | z, t)$ .

The presented approach can be used for variety of applications. For example authors that have high probability for a topic when it starts emerging can be seen as "topic pioneers" who conduct innovative research in that topic. Moreover, active authors that frequently change their topics of interest can be considered as "trend setters" in the respective research community. On the other hand, authors that have high probability at the peak topic activity can be seen as "mainstream" researchers that follow general trends and interests of the community. Finally, authors that have time-independent profiles with stable topics of interest can be recognized as foundational researchers that act independently of fluctuating trends and popular issues. From the application perspective, this knowledge can be exploited in a variety of ways, e.g. for advanced impact ranking, similarity-based contact recommendation for future collaborations, or better summarization of recent research trends and prediction of their further evolution.

### 3 Experiments

We apply our approach to a subset of the CiteSeer dataset consisting of abstracts and titles of research papers published by authors having more than 150 publications from 2001 to 2009. The minimum limit of 150 publications is applied to have sufficient text for capturing author interest over time. Dataset is preprocessed to remove stop words and noise by removing highly frequent terms and terms occurring in less than

**Table 1.** Four topics captured by ATtention model with influential authors and beta PDF showing topic activity

Grid Computing		Image Analysis		Semantic Web		Database Systems	
							
Word	Prob.	Word	Prob.	Word	Prob.	Word	Prob.
grid	0.0297	image	0.0185	resource	0.0380	database	0.0223
framework	0.0190	spectral	0.0150	web	0.0375	query	0.0190
dynamic	0.0175	test	0.0130	metadata	0.0298	sequence	0.0114
resource	0.0175	cluster	0.0120	rdf	0.0211	control	0.0100
integration	0.0131	statistic	0.0120	semantic	0.0195	search	0.0095
Author	Prob.	Author	Prob.	Author	Prob.	Author	Prob.
L. Tong	0.0746	X. Gu	0.1201	I. Horrocks	0.0892	S. Staab	0.1000
E. Gold	0.0207	C. Chan	0.0047	S. Staab	0.0686	I. Horrocks	0.0510
W. Zhao	0.0186	H. Lin	0.0025	A. Lin	0.0042	A. Joshi	0.0478
H. Lin	0.0134	J. Gao	0.0013	W. Nejdl	0.0011	W. Nejdl	0.0328

10 documents. We set the number of topics to  $K=100$  and fix the hyperparameters  $\alpha = 50/K$  and  $\beta = 0.01$ . The results shown are obtained by sampling from the 2000th iteration of Gibbs Sampler. Due to space constraints we are showing four topics with their most influential authors and beta PDF modeling the topic distribution over time. Table 1 shows the top 5 terms and the top 4 authors for each topic. The interesting observation from the results is that the activities in the Semantic Web and Database System topics are correlated. As one topic starts gaining, the activity in other topic starts decreasing. It also shows that as the topic of semantic web started to emerge, influential authors in the database systems topic shifted to semantic web topic.

## 4 Conclusion

In this paper we have proposed a probabilistic approach that models text, authors and timestamps in a given set of documents enabling us to capture temporal topic activity and finding out influential authors for the captured topics. Joint modeling and learning posterior probabilities of text, author and time allows us to query model for any combination of these variables conditioned on each other for finding information about how author’s interests change over time and how activity in topics changes with emergence of new topics. Results from the application of this model to the CiteSeer dataset show the applicability of the model to arbitrary document collections with author and temporal information for detecting topics trends, topic evolution and author’s interests.

## References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* 3, 993–1022 (2003), <http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993>
2. Cheng, V., Li, C.H.: Linked topic and interest model for web forums. In: Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, vol. 01, pp. 279–284. IEEE Computer Society, Washington, DC (2008), <http://portal.acm.org/citation.cfm?id=1486927.1487045>

3. Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The author-topic model for authors and documents. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, UAI 2004, pp. 487–494. AUAI Press, Arlington (2004), <http://portal.acm.org/citation.cfm?id=1036843.1036902>
4. Wang, X., McCallum, A.: Topics over time: a non-markov continuous-time model of topical trends. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2006, pp. 424–433. ACM, New York (2006), <http://doi.acm.org/10.1145/1150402.1150450>
5. Wang, X., Mohanty, N., McCallum, A.: Group and topic discovery from relations and text. In: Proceedings of the 3rd International Workshop on Link Discovery, LinkKDD 2005, pp. 28–35. ACM, New York (2005), <http://doi.acm.org/10.1145/1134271.1134276>

# Role of Emotional Features in Collaborative Recommendation

Yashar Moshfeghi and Joemon M. Jose

School of Computing Science, University of Glasgow,  
Glasgow, UK  
{yashar, jj}@dcs.gla.ac.uk

**Abstract.** The aim of this poster is to investigate the role of emotion in the collaborative filtering task. For this purpose, a kernel-based collaborative recommendation technique is used. The experiment is conducted on two MovieLens data sets. The emotional features are extracted from the movie reviews and plot summaries. The results show that emotional features are capable of enhancing recommendation effectiveness.

## 1 Introduction

Emotions are playing an increasingly important role in social media based Internet culture. For instance, User Generated Contents (UGC) like blogs and social networks are of a steadily increasing importance. They provide the means for a large number of people to express their personal opinions, ideas, and feelings on the web. Data such as product reviews contain emotionally rich content, however, these examples are not the only sources of personal feelings and emotions. Emotional features can also be observed in news articles. Given this proliferation of emotionally active data and online activity, it is important to study the role of emotional features in retrieval and recommendation. Due to recent advances in psychology and linguistics which make emotion extraction from textual documents feasible, we believe it is an appropriate time to consider emotional features in IR. In this poster, we study the effectiveness of emotional features for collaborative recommendation.

Emotion can be extracted in two different ways. First, the emotion that a user experienced during an information seeking process can be captured [1]. Second, the emotion embedded in the content of the document can be analysed (e.g. sentiment analysis and opinion mining) [2]. The essential issue in sentiment analysis is to identify the positive (favourable) or negative (unfavourable) opinion expressed in text. Other works such as Shaikh et al. [2] take one step further and try to extract emotion from text.

Our research question is to investigate whether *emotional features are useful to improve the effectiveness of IR systems*. We use the MovieLens data sets because movies have reviews, comments and plot summaries which are emotionally rich contents. Movies also have other metadata assigned to them (e.g. actor, genre, etc.) which have been used as semantic information to improve rating prediction

[3]. Given the sparsity<sup>1</sup> of the data sets used in collaborative recommendation domain, we hypothesise that the inclusion of emotional features will improve both the rating prediction and movie recommendation accuracy. This research may also be applicable to other domains in which the user comments or item reviews or descriptions are provided.

## 2 Approach

In our study, the memory-based unified model by Wang et al. [4] is used as the baseline. In this model, the rating of an unseen item for a given user is a weighted average of the available ratings in the user-item rating space. This contribution is calculated based on a gaussian kernel density estimation approach whereby users (or items) which are close to the given user and item in the rating space have more influence on the outcome. To calculate the distance, Wang et al. chose a metric based on cosine similarity. For two users  $u$  and  $u'$ , the distance is defined as  $2 - 2\cos(u, u')$ , and similarly for items. They take advantage of user similarity and item similarity embedded in the user-item rating space to improve the probability estimation and counter the problem of data sparsity.

Our goal is to define a new space based on the emotion information and calculate the user similarity and item similarity embedded in these spaces. In a similar approach, Moshfeghi et al. [3] incorporate semantic information, whereas we investigate the effect of emotion information to alleviate data sparsity. The next section explains how emotion spaces are created and the cosine similarity between two users (or items) are calculated.

### 2.1 Construction of an Emotion Space

Based on the work of Shaikh et al. [2], we have implemented an emotion extractor which categorises emotion into 22 emotional categories as defined in the OCC emotion model<sup>2</sup>. In order to create an emotion space, for each movie, the emotions for the sentences in the reviews are extracted and accumulated. Preliminary experiment have shown that considering the top 5 most frequently appearing emotions in the reviews led to the best performing results, and therefore we used these settings as our further experiments. This approach will unify the representation of the emotion features and semantic features of a movie. For example, only the dominant genre in a movie is considered as its genre. Therefore, a movie  $i$  in the emotion space is represented as a vector  $p'_i = (1, \dots, 0, 0, 1, \dots)$  where each dimension corresponds to an emotion extracted from its reviews. A component is set to 1 if the corresponding emotion was dominant, 0 otherwise. Similarly, the semantic spaces are created.

<sup>1</sup> Users do not provide ratings for all the items they have visited.

<sup>2</sup> The OCC emotion model specifies 22 emotion types and two cognitive states. The OCC emotion categorises as joy, distress, happy-for, sorry-for, resentment, gloating, hope, fear, satisfaction, fears-confirmed, relief, disappointment, shock, surprise, pride, shame, admiration, reproach, gratification, remorse, gratitude and anger. The two cognitive states are love and hate [5].

From the representation of a movie, we construct the vector representing a user by giving more importance to the emotion representation of movies that the user liked, i.e. those for which the rating is high. More formally, we define the vector of a user in the emotion space as  $p'_u = \sum r_{u,i} p'_i$  where  $r_{u,i}$  is the rating of the item  $i$  for user  $u$ . Cosine similarity amounts to the percentage of common features, so appears to be a natural choice. Finally, we create a user-item emotion space, where users are represented as vectors in the space and each dimension corresponds to an item  $i$ , and the vector component is the similarity between item  $i$  and the user  $u$ . The vector can be rewritten for such a user as  $p_u = (\dots, \cos(p'_u, p'_i), \dots)$ . We can define a similar vector space for items. We use the rows as the vector representation of users, and columns as the vector representation of items, in order to compute the cosine similarity used in the density estimation formulas. We can then construct a distance based on it, defined as  $2 - 2 \cos(p_u, p_{u'})$  for users and  $2 - 2 \cos(p_i, p_{i'})$  for items.

### 3 Experiment and Result

We performed an evaluation using two MovieLens data sets containing 100,000 ratings for 1682 movies by 943 users (100K data set) and 500,000 ratings for 3900 movies by 3020 users (500K data set) respectively. The latter was extracted from the 1M MovieLens data set by randomly selecting half of the users (for computational reasons). We considered the genre, the actor, and the director as our semantic spaces and the two emotion spaces created from plot summaries and movie reviews information as our emotion spaces. The semantic spaces are used for comparison purpose. We extracted the information needed to define the different emotion and semantic spaces from the IMDb Website<sup>3</sup>.

Evaluation was performed through 10-fold cross validation. In the test set, user ratings were randomly split into two sets of equal size, one for observed items and the other for held-out items. Held-out items for a user were discarded when predicting, and were only used for evaluation purposes, i.e. to compare the predicted rating with the observed one. The predicted ratings were compared to their real value using mean squared error (MSE) which is a standard metric for CF along with mean average precision (MAP). We tested the performance of five experiment scenarios, each of which was conducted to examine the effect of one space on rating prediction performance. We modified the Wang et al. [4] work by including two new kernels for items and users, based on one of the following spaces: Rating (R), Genre (G), Actor (A), Director (D), Plot Summary Emotion (P), and Movie Review Emotion (M) spaces. The results are presented in the Tables 1 and 2. We used pairwise combinations of rating plus other spaces for predicting ratings. The purpose of this decision is to isolate the potential contribution of each space to the rating. In each pair a total kernel was defined as the product of the two individual kernels. Each scenario is labeled by the letters corresponding to the spaces, for example, RM refers to the combination of rating, and movie review emotions spaces.

<sup>3</sup> The Internet Movie Database (IMDb, [www.imdb.com](http://www.imdb.com)).

**Table 1.** MSE for 100K and 500K test collections. The percentage corresponds to the improvement.

	Rating	Rating + Semantic			Rating + Emotion	
	Baseline	RA	RD	RG	RP	RM
100K	0.911	0.859 +5.7%	0.851 +6.5%	0.859 +5.7%	0.879 +3.5%	0.875 +3.9%
500K	0.881	0.82 +6.9%	0.804 +8.7%	0.814 +7.6%	0.84 +4.6%	0.828 +6%

**Table 2.** MAP for 100K and 500K test collections. The percentage corresponds to the improvement.

	Rating	Rating + Semantic			Rating + Emotion	
	Baseline	RA	RD	RG	RP	RM
100K	0.654	0.729 +11.4%	0.731 +11.7%	0.704 +7.6%	0.69 +5.5%	0.697 +6.5%
500K	0.705	0.778 +10.3%	0.776 +10%	0.76 +7.8%	0.75 +6.3%	0.76 +7.8%

## 4 Discussion and Conclusions

The results show that emotional features consistently play a role in improving the recommendation quality by comparison to the scenario where only rating space is used (i.e. the baseline). This indicates that emotion spaces encapsulate a potential source of information. A comparison between the improvement achieved in MSE and MAP values presented in Table 1 and 2 shows that emotion spaces are more effective to improve the ranking of the movies than predicting the actual ratings.

The results also show that the effectiveness of emotion spaces increases with the size of the corresponding data set. This behaviour is not observed for the actor and director spaces. It is interesting to note that movie review emotion space and genre space have the same improvement in terms of MAP for the 500K data set. The emotion features are the outcome of an emotion extraction system and not manually created metadata as it is the case for the semantic spaces such as actor, director, and genre. Thus they do not require human intervention which is costly and time consuming. Therefore, we believe that emotional features can be a good alternative for semantic features.

Finally, emotion extracted from the movie plot summary and movie review emotion spaces affect system performance differently. This is perhaps because movie reviews include a richer emotional content than plot summary does. It is also important to consider that there is room for improving the accuracy of emotion extraction techniques.

In this poster we investigated the effectiveness of the emotional features in a state-of-the-art memory-based collaborative recommendation approach. The results of this work provides a foundation for future research on utilising emotion information in IR tasks. An unanswered question remains: what would be the effect of such information if it is combined with semantic features using

a model-based collaborative filtering approach. The future work will be directed towards answering this question.

**Acknowledgement.** The authors would like to thank the reviewers, B. Piwowarski, A. Huertas-Rosero, and R. Villa for their comments and D. Hannah for his help implementing the OCC model.

## References

1. Arapakis, I., Jose, J.M., Gray, P.D.: Affective feedback: an investigation into the role of emotions in the information seeking process. In: SIGIR 2008, pp. 395–402 (2008)
2. Shaikh, M.A.M., Prendinger, H., Ishizuka, M.: A Linguistic Interpretation of the OCC Emotion Model for Affect Sensing from Text. *Affective Information Processing*, 45–73 (2009)
3. Moshfeghi, Y., Agarwal, D., Piwowarski, B., Jose, J.M.: Movie Recommender: Semantically Enriched Unified Relevance Model for Rating Prediction in Collaborative Filtering. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 54–65. Springer, Heidelberg (2009)
4. Wang, J., de Vries, A.P., Reinders, M.J.T.: Unified relevance models for rating prediction in collaborative filtering. *ACM TOIS*, 1–42 (2008)
5. Ortony, A., Clore, G., Collins, A.: *The cognitive structure of emotions*. Cambridge University Press, Cambridge (1990)



# The Importance of the Depth for Text-Image Selection Strategy in Learning-To-Rank

David Buffoni, Sabrina Tollari, and Patrick Gallinari

Université Pierre et Marie Curie - Paris 6, LIP6, Paris, France

**Abstract.** We examine the effect of the number documents being pooled, for constructing training sets, has on the performance of the learning-to-rank (LTR) approaches that use it to build our ranking functions. Our investigation takes place in a multimedia setting and uses the ImageCLEF photo 2006 dataset based on text and visual features. Experiments show that our LTR algorithm, OWPC, outperforms other baselines.

## 1 Introduction

In the case of text-image retrieval, the task is to produce a sorted list of images given a user query (that has a text and an image part). The sorted list is created by ordering the documents according to a score given to each document by a function. The quality of the list depends on the position of the images that are relevant to the query: since only the few first images are presented to the user, it is necessary to have a high precision on the top of the list. We consider the problem of learning the function used to score the (query, document) pairs, on the basis of a training set of queries for which relevant documents are known.

[1] conducted works on the impact of selecting documents on the efficiency and effectiveness of Learning-to-Rank (LTR) algorithms. They employed a number of document selection methodologies such as depth-k pooling and active-learning (MTC), investigating how they affect efficiency, effectiveness, and robustness of built datasets. In this work, we wanted to evaluate the impact of the depth in a pooling methodology on a created LTR dataset for a text-image retrieval application.

In Section 2, we employed three models, one serving as a baseline, that is a linear combination of textual and visual similarities and two state-of-the-art LTR algorithms, that is  $SVM^{RANK}$  [3] and OWPC [6]. In Section 3, we compare their results obtained on a built dataset for text-image retrieval.

## 2 Models

*Baseline.* Our baseline model is a manual combination between a BM25 function [4] and a visual similarity as in [5]. For a query  $q$ , composed of a text part  $q_t$  and an image set  $q_i$ , we can write the baseline of a document  $d$  as follows:  $BL(q, d) = \lambda BM25(q_t, d_t) + (1 - \lambda) \max(Histo_{HSV}(q_i, d_i))$  where  $\lambda$  is a trade-off between the textual and the visual information and  $\max_{q_i}(Histo_{HSV}(q_i, d_i))$  is

the maximum fusion operator applied on histogram distance of HSV descriptors between the pictures of an image query set  $q_i$  and the image  $d_i$ . HSV descriptors are taken on 9 different regions of each image. In section 3, we set different values of  $\lambda \in \{0, 0.1, \dots, 1\}$ . With  $\lambda = 0$ , we recover the visual model  $\max(Histo_{HSV})$ , with  $\lambda = 1$ , the textual model BM25 is applied and with  $\lambda = *$  the model is a combination between the two sources where  $\lambda$  is chosen on a validation set.

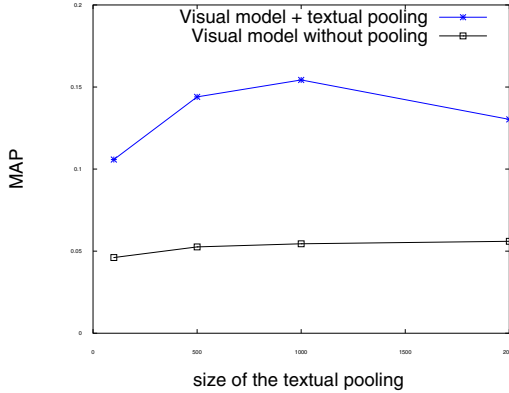
*SVM<sup>RANK</sup>*. *SVM<sup>RANK</sup>* formulates a ranking problem as a SVM problem and optimizes it by taking into account a convex loss as the mean of the pairwise classification losses. Learning with such a loss function is equivalent to optimizing the mean rank of the relevant documents.

*OWPC*. Instead of taking the mean of the pairwise classification losses, *OWPC* is an extension pairwise classification approach, using convex Ordered Weighted Averaging (OWA) operators [7]. OWA operators are arbitrarily fixed by a decreasing weighting schemes based on generator functions of the form  $\alpha_j = \frac{g(j)}{\sum_{d=1}^n g(d)}$ , where  $n$  is the number of irrelevant documents. Also, we can affect the degree to which the loss function focuses on the top of the list. The constant weights  $g_{=1}(d) = 1$  allow us to recover the pairwise classification setting where the error function optimizes the mean rank of relevant documents (identically to *SVM<sup>RANK</sup>*). The weighting scheme we chose in the experiments is based on the rank of  $d$  as  $g_{\sqrt{\cdot}}(d) = \sqrt{(1/d)}$  applied on the entire list of irrelevant documents.

### 3 Experimental Results

*Document collection*. ImageCLEFphoto'06 [2] benchmark contains 20,000 photos. Each image has a corresponding semi-structured caption. We use the 60 query topics and the corresponding binary relevance assessments. Each query is composed of a text part and a visual part (3 images). We first remove standard English stop words. Then the word suffix is stripped using a Porter stemming algorithm. For each document answering to a query, we extracted a total of 60 similarities (both textual and visual). For example, BM25, TFIDF or Language Models for textual similarities and different fusion operators applied on histogram distance of different image descriptors (HSV, SIFT...).

Then, when interrogating the corpus, we employ a depth- $k$  pooling technique based on textual similarities between the query and the documents to select a subset of the collection of size  $k$ . As experimentally shown in [1], the selection strategy of the documents plays an important role in the learnt ranking function performance. In our work, we adopted this methodology to improve diversity and avoid being biased by only one similarity during the learning step. We chose to retrieve the top  $k$  documents of four textual models: BM25, TFIDF [8] and two language models with Jelinek-Mercer and Absolute Discount smoothing functions as in [9]. We think these models are quite different and would result in uncorrelated documents which could improve the diversity of LTR algorithms. The reader can notice these 4 textual similarities are also used (with others textual similarities) as features in the learning step.



**Fig. 1.** Variations of MAP performances depending on a textual depth- $k$  pooling. Two visual models ( $BL_{\lambda=0}$ ), one with a textual pooling processed before and one without.

*Experimental protocol.* The ImageCLEFphoto dataset was split in 5 folds where each fold contains training/validation/test query sets. Each document features were normalized by query and by similarity. For each of the LTR algorithms, training is done for various values of the coefficient  $C \in \{10^{-2}, 10^{-1}, \dots, 10^2\}$  of the SVM problem (as well for the  $\lambda$  parameter in the  $BL$  model) where the best on the validation set is chosen for based on their MAP score [8]. We retain the MAP as evaluation measure.

*Pooling experiments.* We compare the impact of a textual pooling approach on our systems, aiming to examine the importance of the depth of the pooling documents on general performances of the systems. We can see in Figure 1 an improvement of performances in terms of MAP for the visual model ( $BL_{\lambda=0}$ ) when we have processed a textual pooling before. So filtering visual documents by textual pooling demonstrates the impact of the textual information on a visual model's performances. Textual similarities of documents are more discriminative than visual similarities. Consequently, constituting a filtered pool by textual information improves necessarily visual model performances. Then we tested with various values of  $k \in \{100, 500, 1000, 2000\}$  and we would like to see the performances evolution of our three models ( $BL$ ,  $SVM^{RANK}$  and OWPC).

Performances in MAP of various systems are reported in Table 1 where all system performances are improved with a  $k < 1000$  and seem to reach a level with  $k \geq 1000$ .

Consequently, the size of this subset is very crucial. In our case, having  $k < 1000$  is too small because a lot of relevant documents are considered non-relevant because there were not selected in the pool. We can infer that most of the relevant documents are captured in the pool and what varies across the systems is the order of elements within the pool. As expected OWPC gives the best results

<sup>1</sup> The documents, within the chosen subset, judged relevant are assumed to be the only relevant ones for the query, and all unjudged documents are considered non-relevant.

**Table 1.** Test performances of BM25, a visual model  $\max(Histo_{HSV})$ , the baseline  $BL_{\lambda=*}$  and two LTR algorithms ( $SVM^{RANK}$  and  $OWPC_{g\sqrt{J}}$ ) on the ImageCLEF-photo'06 dataset with a variation of the depth of the textual pooling in MAP

	100	500	1000	2000
$BL_{\lambda=1}$ (BM25)	<b>0.240</b>	0.294	0.299	0.302
$BL_{\lambda=0}$ ( $\max(Histo_{HSV})$ )	0.106	0.144	0.154	0.130
$BL_{\lambda=*}$	0.235	0.278	0.285	0.291
$SVM^{RANK}$	0.220	0.283	0.297	0.294
$OWPC_{g\sqrt{J}}$	0.226	<b>0.297</b>	<b>0.306</b>	<b>0.303</b>

except for very small pools ( $d = 100$ ) and  $SVM^{RANK}$  has surprisingly lower performances than BM25.

## 4 Conclusion

We applied two LTR algorithms on a text-image retrieval dataset built according to a depth-k pooling methodology. Experimentally speaking, we showed the importance of pooling documents according their textual part and the evolution in performances of LTR algorithms due to the depth of the pooling. These results are encouraging and a future step would be to evaluate the impact of adding visual models during the pooling step.

## References

- [1] Aslam, J.A., Kanoulas, E., Pavlu, V., Savev, S., Yilmaz, E.: Document selection methodologies for efficient and effective learning-to-rank. In: ACM SIGIR (2009)
- [2] Clough, P., Grubinger, M., Deselaers, T., Hanbury, A., Müller, H.: Overview of the imageCLEF 2006 photographic retrieval and object annotation tasks. In: Peters, C., Clough, P., Gey, F.C., Karlgren, J., Magnini, B., Oard, D.W., de Rijke, M., Stempfhuber, M. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 579–594. Springer, Heidelberg (2007)
- [3] Joachims, T.: Optimizing search engines using clickthrough data. In: KDD (2002)
- [4] Robertson, S.E., Walker, S., Hancock-Beaulieu, M., Gull, A., Lau, M.: Okapi at trec. In: TREC, pp. 21–30 (1992)
- [5] Tollari, S., Glotin, H.: Web image retrieval on imageval: Evidences on visualness and textualness concept dependency in fusion model. In: ACM CIVR (2007)
- [6] Usunier, N., Buffoni, D., Gallinari, P.: Ranking with ordered weighted pairwise classification. In: ICML, pp. 1057–1064 (2009)
- [7] Yager, R.R.: On ordered weighted averaging aggregation operators in multi-criteria decision making. IEEE Transactions on Systems, Man and Cybernetics 18 (1988)
- [8] Yates, R.B., Ribeiro-Neto, B.: Modern Information Retrieval. A. Wesley, Reading (1999)
- [9] Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst. 22, 179–214 (2004)

# Personal Blog Retrieval Using Opinion Features

Shima Gerani, Mostafa Keikha, Mark Carman, and Fabio Crestani

Faculty of Informatics, University of Lugano, Lugano, Switzerland

`{shima.gerani,mostafa.keikha,mark.carman,fabio.crestani}@usi.ch`

**Abstract.** Faceted blog distillation aims at finding blogs with recurring interest to a topic while satisfying a specific facet of interest. In this paper we focus on the personal facet and propose a method that uses opinion features as indicators of personal content. Experimental results on TREC BLOG08 data-set confirm our intuition that personal blogs are more opinionated.

## 1 Introduction

Recently, blogs are growing rapidly and becoming one of the most important sources of information in the web. Considering the huge amount of information in this category and the specific needs of blog search users, designing a system for blog search seems necessary. In order to address the blog search problem, TREC (Text REtrieval Conference) organizers have introduced *blog distillation* task as part of the *blog track* in 2007 [1,2]. Blog distillation task only focused on relevance retrieval until 2009 when the *faceted blog distillation* task was introduced in which one aims to find blogs with a recurring interest in topic X while having a specific facet. Facets that are considered are Opinionated vs. Factual, Personal vs. Company and In-depth vs. Shallow [3]. Facet blog distillation can be seen as a two step process. In the first step, a system retrieves blogs only considering their relevance to the topic and ignores any facet related feature. In the second step, facet related features are taken into account in order to re-rank the retrieved blogs.

In this paper we focus on the personal vs. company facet and consider the first value of the facet (i.e. personal) in our system. The purpose of personal blog distillation is to help users in finding blogs that are written in personal time rather than those written by companies which mostly contain commercial contents. Personal blog retrieval has been shown to be very challenging in the last year running of the task, such that most of the participants could not manage to improve their relevance retrieval baselines [3]. We propose using opinion features as indicators of personal blogs since we believe that “personal content” is on the whole more likely to contain opinions than “official content”.

In the rest of this paper we first explain our method in scoring a blog based on the personal score of its posts. We then explain how the personal score of a post is calculated using its opinion features. Finally, we report our results on re-ranking the standard TREC baselines.

## 2 Personal Blog Retrieval

We calculate personal facet scores for each retrieved blog, denoted  $personal(b)$  based on its relevant posts to the query, (i.e. posts of the blog that appear in the top 15000 relevant posts to the topic). The personal facet score of a blog is then calculated based on the average of its posts' facet score:

$$score_{personal}(b) = E_b[score_{personal}(d)] = \sum_{d \in b} p(d|b) score_{personal}(d) \quad (1)$$

We consider uniform probability over posts of a blog, (i.e.  $p(d|b) = 1/|posts|$ ). The facet score induce a ranking, denoted  $r_{personal}(b, q)$ , which we combined with the original relevance ranking  $r_{rel}(b, q)$  using the Borda Fuse aggregation method as follows:<sup>1</sup>

$$score_{BF}(d, q) = \alpha r_{rel}(d, q) + (1 - \alpha) r_{personal}(d, q) \quad (2)$$

## 3 Calculating Post-level Personal Score

We estimate the personal score of a post based on the presence of opinion features. The opinion features are extracted using the TREC Blog06 collection and the corresponding relevance/opinion judgments. Features (terms) are weighted according to a document-frequency-based version of the Mutual Information (MI) metric [4]. We then calculate personal facet score for each retrieved post by averaging over the weight for each word in the post (see equation 4 below.)

In order to calculate *opinionated* weights for terms we split the Mutual Information metric into two values as follows. Let  $\mathcal{T}$  denote the event that a document contains the particular term  $t$ , and  $\bar{\mathcal{T}}$  the event that the document does not contain the term. Then let  $\mathcal{O}$  denote the event that a document is classed as being (relevant and) opinionated about the query and  $\bar{\mathcal{O}}$  that it is (relevant but) not opinionated about the query. We calculate the opinion score for a term by calculating the MI summation only over the two positively correlated quadrants (i.e.  $\mathcal{T} \cap \mathcal{O}$  and  $\bar{\mathcal{T}} \cap \bar{\mathcal{O}}$ ) as follows:

$$opin(t) = p(\mathcal{T}, \mathcal{O}) \log \frac{p(\mathcal{T}, \mathcal{O})}{p(\mathcal{T}), p(\mathcal{O})} + p(\bar{\mathcal{T}}, \bar{\mathcal{O}}) \log \frac{p(\bar{\mathcal{T}}, \bar{\mathcal{O}})}{p(\bar{\mathcal{T}}), p(\bar{\mathcal{O}})} \quad (3)$$

We calculate the required joint and marginal probabilities using document frequency estimates using the sets of *opinionated*  $O$  and *relevant*  $R$  documents in the TREC Blog06 collection as:

$$\begin{aligned} p(\mathcal{T}, \mathcal{O}) &= \mathbf{df}(t, O) / |R| \\ p(\mathcal{T}) &= \mathbf{df}(t, R) / |R| \\ p(\mathcal{O}) &= |O| / |R| \end{aligned}$$

<sup>1</sup> Note that whenever there are ties in the ranking, (i.e. blogs  $b_1$  and  $b_2$  have the same score), then the rank for those blogs is the average of the (total order) ranking.

**Table 1.** MAP of the Personal facet re-ranking over the three TREC baselines for 2009 queries

	MAP	Rprec	bpref	P@10
stdBaseline1	0.2473	0.2743	0.2212	0.2200
personal	<b>0.2866 (15.86%)</b> *	<b>0.3017</b> †	<b>0.2599</b> *	<b>0.2500</b> †
stdBaseline2	0.2034	0.2487	0.1979	0.1900
personal	<b>0.2229 (9.59%)</b> †	<b>0.2616</b> †	<b>0.2177</b> †	<b>0.2000</b> †
stdBaseline3	0.0745	0.0941	0.0578	0.1100
personal	<b>0.0786 (5.50%)</b> †	<b>0.1078</b> *	<b>0.0628</b> *	0.1000

Where  $\text{df}(t, O)$  is the number of opinionated documents containing the term  $t$ . The other joint and marginal probabilities required for equation 3 are estimated analogously.

Having calculated opinion weight for each term, we then average these lexicon weights over each document to calculate personal facet score for the document as follows:

$$\text{personal}(d) = E_d[\text{personal}(t)] = \sum_{t \in d} p(t|d) \text{personal}(t) \quad (4)$$

## 4 Experimentation and Conclusion

In our experiments we use the BLOG08 collection and the set of 26 topics. We use the corresponding positive facet judgements (i.e. personal) for evaluation. The Terrier Information Retrieval system<sup>2</sup> is used to index the collection with the default stemming and stopwords removal.

In order to have a fair comparison between different facet ranking methods, in TREC 2010, the organizers distributed three standard baselines among participants to carry on their facet detection methods. We use these standard baselines and re-rank their results for personal queries using our facet scoring method. The appropriate value for the parameter  $\alpha$  in the model is learnt in order to maximize Mean Average Precision (MAP). We use the TREC 2009 data (i.e. relevance judgments) for training when reporting the results over 2010 queries and use the TREC 2010 data for the case of reporting the results over 2009 queries.

Tables 1 and 2 show the performance of the proposed personal blog retrieval system over the 3 standard TREC baselines. We report the Mean Average Precision (MAP) as well as R-Precision (R-Prec), binary Preference (bPref), and Precision at 10 documents (P@10) in terms of personal blog finding. The first row for each baseline in the tables indicates the performance of the TREC baselines without any personal blog finding feature. The second row (personal) for each baseline represents the performance of our proposed method in scoring

<sup>2</sup> <http://ir.dcs.gla.ac.uk/terrier/>

**Table 2.** MAP of the Personal facet re-ranking over the three TREC baselines for 2010 queries

	MAP	Rprec	bpref	P@10
stdBaseline1	0.1370	0.1382	0.1022	0.1733
personal	<b>0.1814 (32%)</b>	<b>0.2071</b>	<b>0.1596</b>	0.1600
stdBaseline2	0.0945	0.1156	0.0712	0.0933
personal	<b>0.0990</b>	0.1042	0.0619	<b>0.1</b>
stdBaseline3	0.1120	0.1298	0.0885	0.1066
personal	0.0844	0.1113	0.0549	<b>0.1067</b>

and re-ranking the results of the baseline. Statistical significant tests are done using wilcoxon signed-rank test and indicted by † and \* at level 0.05 and 0.01 respectively<sup>3</sup>.

The results show that the proposed personal blog retrieval system can improve all three standard baselines in terms of almost all reported performance measures for the TREC09 query set. However, the improvements over the TREC10 queries are not significant. This is mainly due to the difficulty of TREC10 queries which on average have less number of relevant blogs in the assessments. The average number of relevant blogs per query in TREC09 is 15.7 while this is 9.7 for TREC10 queries. This introduces more noise in the result set of TREC10 baselines which makes the re-ranking more difficult.

## 5 Conclusion and Future Work

In this paper we considered the problem of personal blog retrieval. We have shown that opinion features are good indicators of personal content.

For future work we plan to investigate the effect of using other features such as the number of emoticons, writing style, spelling error and etc. on the personal blog retrieval.

## References

1. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the trec-2007 blog track. In: Proceedings of the 16th Text REtrieval Conference (TREC 2007) (2007)
2. Ounis, I., De Rijke, M., Macdonald, C., Mishne, G., Soboroff, I.: Overview of the TREC-2006 blog track. In: Proceedings of the 15th Text REtrieval Conference (TREC 2006), pp. 15–27 (2006)
3. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the TREC 2009 Blog Track. In: Proceedings of TREC 2009 (2009)
4. Manning, C.D., Schtze, H.: Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge (June 1999)

<sup>3</sup> Since the number of queries for each year is small (less than 10), we calculated exact probabilities in the statistical tests.



# Processing Queries in Session in a Quantum-Inspired IR Framework

Ingo Frommholz<sup>1</sup>, Benjamin Piwowarski<sup>1</sup>, Mounia Lalmas<sup>2</sup>,  
and Keith van Rijsbergen<sup>1</sup>

<sup>1</sup> School of Computing Science, University of Glasgow  
ingo@dcs.gla.ac.uk, benjamin@bpiwowar.net, keith@dcs.gla.ac.uk  
<sup>2</sup> Yahoo! Research Barcelona  
mounia@acm.org

**Abstract.** In a search session, users tend to reformulate their queries, for instance because they want to generalise or specify them, or because they are undergoing a drift in their information need. This motivates to regard queries not in isolation, but within the session they are embedded in. In this poster, we propose an approach inspired by quantum mechanics to represent queries and their reformulations as density operators. Differently constructed densities can potentially be applied for different types of query reformulation. To do so, we propose and discuss indicators that can hint us to the type of query reformulation we are dealing with.

## 1 Introduction

Common IR systems regard queries in isolation, albeit allowing for query expansion based on relevance feedback. However, in a search session users often reformulate their initial query, for instance because it was ill-specified or they noticed that it did not precisely represent their information need as a more specialised or generalised query would. For example, a query for “processors” may be deemed too broad when looking for processors manufactured by a specific company and thus might be reformulated into “amd processors” or “intel processors”. A query reformulation might also represent a minor drift in information needs, for instance a user could seek for pubs in San Francisco after looking for hotels in San Francisco, or it may be the result of a new, independent, information need. This observation motivates that queries should be seen in a session context rather than in isolation, and motivated the new session track at TREC<sup>1</sup>.

Using user session information has already been studied in IR. For example, Zhai [5] proposes a risk minimisation framework where each interaction between a user and the system can be captured by a profile described as a language model. However, this type of profile-based approach does not exhibit how to actually take into account query reformulation. Another type of related work is done on identifying query reformulations in query logs, where the typical usage is to propose the user potentially relevant query reformulations [1].

---

<sup>1</sup> <http://ir.cis.udel.edu/sessions/index.html>

Recently a quantum-inspired IR framework was introduced that assumes that there exists an “information need space”, geometrically modelled as a Hilbert space (a vector space with an inner product). The system’s view on the user’s information need is represented as a set of *state vectors*  $\varphi_i$  (unit vectors in the Hilbert space) with a given probability  $p_i$ . Documents are modelled as subspaces in the information need space. A matching function is realised by taking the squared length of the projection of the state vectors and their respective probabilities [3]. Formally, the user’s information need can be represented as a *density operator*  $\rho = \sum_i p_i \varphi_i \varphi_i^\top$ , which defines a probability distribution over the subspaces in a Hilbert space [4] in the following way. The probability  $\Pr(S)$  that a document represented by a subspace  $S$  is relevant is defined to be  $\text{tr}(\rho \hat{S})$  where  $\hat{S}$  is the projector onto the subspace  $S$ . It has been shown that by doing so the framework can compete with BM25 in an ad hoc scenario [3]. Following this line, we regard  $\rho$  as a query representation in the further considerations.

A potential but so far rather unexplored aspect of the above framework is its ability to dynamically react on different kinds of user interaction, based on the current state the system is in (i.e., the query representation  $\rho$ ). Translated to queries in a session, the system is able to change its state (based on a newly arriving query or query reformulation) while considering the state it was previously in, for instance based on a previous query.

## 2 Processing Queries in Session

We illustrate the approach for the simple case of two queries, a query  $q$  and a consecutive query  $q'$ , although it can potentially be extended to a session of more than 2 queries, or to sessions with different types of interaction (e.g., clicks, trackback).

It is desirable to detect what is the relationship between queries  $q$  and  $q'$ , i.e. whether  $q'$  is related (generalisation, specialisation or information need drift) or unrelated to  $q$ . The idea is that a consecutive query  $q'$  should be processed differently, depending on its type. This requires an automatic categorisation of the query reformulation, for which we can apply the quantum formalism as well, and we present the general idea on how this can be done in Section 2.1. We then describe in Section 2.2 how query densities can be created depending on the relationship between the two queries.

The methods presented below rely, for each query  $q$ , on the computation of densities  $\rho_q$ , which can be done as described in [3]. We also rely on the definition of a subspace  $O_q$  that has the property that *any* possible information need vector of a user that has typed the query  $q$  is contained by it. This subspace is the subspace spanned by all the vectors that define the density.

### 2.1 Query Categorisation

To categorise a query reformulation, the idea is to use the densities and their associated subspaces. We consider the probabilities of observing an information

need corresponding to the query  $q$  (resp.  $q'$ ) given that the query was  $q'$  (resp.  $q$ ):

$$\Pr_{q'}(O_q) = \text{tr}(\rho_{q'} O_q) \text{ and } \Pr_q(O_{q'}) = \text{tr}(\rho_q O_{q'})$$

The relationship between the two probabilities indicates what is the relationship between the two queries. Let us illustrate this idea by assuming we have documents which are about “intel processors” and documents about “amd processors”. Given the way document subspaces and query densities are constructed in the quantum-based framework [3], this would mean our information need space is contained in a two-dimensional Hilbert space with the two dimensions representing “intel processors” and “amd processors”, respectively. Let us assume that  $q =$  "intel processors" and  $q' =$  "processors", so  $q'$  is a generalisation of  $q$ .  $O_q$  would then be a 1-dimensional subspace made of the “intel processors” dimension, whereas  $\rho_{q'}$  would span over both dimensions and would not be contained in  $O_q$ , leading to a lower  $\Pr_{q'}(O_q)$ .  $O_{q'}$ , on the other hand, would likely comprise both dimensions, whereas  $\rho_q$  would only contain the “intel processors” dimension. In this case  $\rho_q$  would be fully contained in  $O_{q'}$ , and hence  $\Pr_q(O_{q'})$  would be 1.

Generalising, our hypothesis is that firstly, if  $\Pr_{q'}(O_q)$  is lower than  $\Pr_q(O_{q'})$ , this is an indicator for generalisation (and vice versa for specialisation). Secondly, low values for both  $\Pr_{q'}(O_q)$  and  $\Pr_q(O_{q'})$  may mean that  $q$  and  $q'$  are independent, as the respective densities are now at a higher distance to each other. Finally, high but comparable values for  $\Pr_{q'}(O_q)$  and  $\Pr_q(O_{q'})$  may indicate an information need drift since it means that both densities are closely together, but do not have a specialisation/generalisation relationship.

## 2.2 Query Processing

Having categorised the query reformulation  $q'$  as proposed above, we can now process it according to its type.

For independence, we propose to simply use  $\rho_{q'}$  as the representation the query, since there is no obvious way to use the information contained in the previous query  $q$ .

In the case of a specialisation or a topic drift, we would choose a strategy and compute a density  $\rho_{q'|q}$  that represents a density  $\rho_q$  restricted to the region of the information need space defined by  $O_{q'}$ . The idea is that if the user has submitted the query  $q'$ , it is because he or she believes that the information need has to be restricted or changed, but supposes that the previous interactions, in our case the submission query  $q$ , defines the set of initial potential information needs. In the “San Francisco” example, the interest would be that “San Francisco” has already been disambiguated, and we just need to drift from “hotels” to “pubs”. A similar process would occur in the case of “intel processors”, for example if the interest of the user about processors (e.g., power consumption) had already been specified.

In order to do so, we use the projection operator as defined in [2, p. 100]. This is analogous to a measurement in quantum mechanics, where the subspace  $O_{q'}$

represents an observed event and its measurement changes the state of the system. Interpreted in an IR scenario, the current state of the system, represented by the density  $\rho_q$ , is changed as we observed that the user submitted the query  $q'$ .

Finally, when  $q'$  generalises  $q$ , we might use  $\rho_{q'}$  as in the case of independence. It might however also be interesting to extract some of information about the previous query, but this is more prospective. The idea would be, using ontologies, lexical resources or user logs, to generalise the query  $q$  *until* the transformed query  $q_g$  generalises  $q'$ . Then, we would apply the previous method to compute  $\rho_{q'|q_g}$ . For example, going from “intel processor” to “processor” should still retain the fact that the user is interested by micro-processors (and not food processors). In order to do so, “intel processor” would have to be generalised into “micro-processors”.

### 3 Conclusion

We have discussed how a query can be represented in a quantum-inspired framework, taking into account previous queries that appeared in the same session. We proposed how we can detect within this framework the type of query reformulation we are dealing with, and, based on this type, how to transform the query representation. The various hypotheses formulated in this poster need to be verified. For our evaluation, we plan to use the test collection from TREC 2010 session track, where approaches using a query and consecutive query reformulation are evaluated.

### Acknowledgement

This research was supported by an Engineering and Physical Sciences Research Council grant (Grant Number EP/F015984/2). This paper was written when Mounia Lalmas was a Microsoft Research/RAEng Research Professor.

### References

1. Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., Vigna, S.: The query-flow graph: model and applications. In: CIKM (2008)
2. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Interaction. Cambridge University Press, Cambridge (2000)
3. Piwowarski, B., Frommholz, I., Lalmas, M., Van Rijsbergen, K.: What can Quantum Theory Bring to Information Retrieval? In: CIKM (October 2010)
4. van Rijsbergen, C.J.: The Geometry of Information Retrieval. Cambridge University Press, New York (2004)
5. Zhai, C., Lafferty, J.: A risk minimization framework for information retrieval. Information Processing & Management 42(1) (2006)

# Towards Predicting Relevance Using a Quantum-Like Framework\*

Emanuele Di Buccio<sup>1</sup>, Massimo Melucci<sup>1</sup>, and Dawei Song<sup>2</sup>

<sup>1</sup> University of Padua, Italy

<sup>2</sup> Robert Gordon University, Aberdeen, UK

**Abstract.** In this paper, the user’s relevance state is modeled using quantum-like probability and the interference term is proposed so as to model the evolution of the state and the user’s uncertainty about the assessment. The theoretical framework has been formulated and the results of an experimental user study based on a TREC test collection have been reported.

## 1 Introduction

An Information Retrieval (IR) system has to decide whether a document contains information relevant to a user who interacts with the document. By means of relevance prediction, the system may propose documents, queries, advertisements or other information. In this paper, it is distinguished between *relevance assessment* (or assessment) and *relevance state* (or state) – the states are “stored” in the user’s mind and cannot be observed by the system whereas the assessments can be observed only at the end of the interaction time. Thus, a state can be viewed as a superposition of assessments which “collapses” to an assessment when the user may make it explicit.

If the retrieval system monitors the interaction, it can collect interaction features (e.g., the display time) and the final assessment. The collected assessments may be used to train the system, but they are only the final outcome of a quite complex interaction in which many unobserved variables have been hidden to the system. If the user is not even willing to provide an assessment, it is customary that the system implements implicit feedback based on observed interaction features [1,2].

Differently from the assessment, the state may change while the user interacts with the document, but it is unobservable. Moreover, the state is not necessarily the final assessment because the latter is only one of potential values of the state. What the system should do is to predict the state at an arbitrary interaction instant. To this end, a probabilistic model may be used, however, either the training dataset is unavailable or only stores (final) assessments whereas a training set should store states so as to predict the state at every instant.

---

\* The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement N. 247590 and the UK’s Engineering and Physical Sciences Research Council under the Renaissance project (Grant No: EP/F014708/2).

In this paper, an approach based on state modeling which does not rely only on interaction features is proposed. The state is modeled as a *quantum-like* probability (QLP) function which does not obey to the classical probability laws – the obedience to these laws would imply that the state *is* either relevance or non-relevance at every interaction instant. QLP instead includes an *interference term* emerging from the superposition of relevance and non-relevance. The interference term updates the prediction probability depending on the degree to which the state evolves and on the user’s uncertainty about relevance.

Our hypothesis is that the superposition of relevance and non-relevance, which reflects the user’s uncertainty about his assessment, affects the user’s behaviour and therefore the interaction features, thus making prediction performed by the system less precise and more prone to error. If our hypothesis is true, it can have important theoretical and practical implications. For example, if interference is detected, the system may support the user to clarify his state by suggesting example documents or by presenting the results in more effective way. Moreover, if there is interference, the system may infer that the initial query is difficult and therefore invite the user to add terms.

Recently, QLP has been investigated in the context of document ranking [3], cognition [4] and dependency between topics [5] or between documents [6]. In this paper, the dependency between the time intervals of the interaction and the assessment is addressed both at the theoretical level and through an experimental user study based on a TREC test collection.

## 2 Probability with Relevance State

Suppose the display time is divided into a given number of equally sized time intervals. The time interval is represented by an observable  $T$  such  $T = t$  means that the  $t$ -th interval has been observed. Using the distributive law,

$$(T = t) = (T = t \wedge R = r) \vee (T = t \wedge R \neq r) \quad (1)$$

where  $R = r$  refers to an assessment<sup>1</sup>. Using classical probability,

$$\Pr(T = t) = \Pr(T = t | R = 0) \Pr(R = 0) + \Pr(T = t | R = 1) \Pr(R = 1) . \quad (2)$$

Prediction requires the estimation of  $\Pr(R = 1 | T = t)$  by means of Bayes’ theorem. However, classical probability *assumes* that the state is either relevance or non-relevance at every interaction instant, that is, that the user is concerned about relevance or non-relevance, exclusively. In fact, the exclusiveness between relevance and non-relevance is not true and a superposition state better models this uncertainty because it does not admit the distributive law [7]. Note that the same argument is valid also for non-binary relevance.

A superposition state  $\phi$  can be modeled by using QLP:

$$|\phi\rangle = a_0|R = 0\rangle + a_1|R = 1\rangle \quad |a_0|^2 + |a_1|^2 = 1 \quad |a_r|^2 = \Pr(R = r) . \quad (3)$$

<sup>1</sup> For the sake of simplicity, binary assessments are supposed,  $r = 0, 1$ .

where constants and vectors are in the complex field.  $\phi$  is neither the positive assessment, the negative assessment nor the average of the two (i.e., the expected value, the mean or the mixture) because, if it were, the existence of *any* of the predefined assessments should be admitted for *every* user. QLP is expressed as

$$q(t) = |\langle \varphi | T = t \rangle|^2 = p(t) + 2|a_0||a_1||\langle R = 0 | T = t \rangle| |\langle T = t | R = 1 \rangle| \cos \theta \quad (4)$$

where  $p(t) = \Pr(T = t)$ ,  $\cos \theta$  is the real part of the complex number  $\bar{a}_0 a_1 \langle R = 0 | T = t \rangle \langle T = t | R = 1 \rangle$  and the second term of the right hand of Eq. 4 is the *interference term*.

### 3 An Experimental Study

Our experimental study aimed at measuring the interference term. Through a user study, the subjects were asked to interact with the WT10g test collection and to assess the relevance after browsing the documents; for example, the event that the user `user1` submitted the topic 506, started to interact with the document WTX074-B09-156 on the 29th of July, 2008, at 5:44:23pm and made the relevance assessment after 167 seconds, was recorded as `user1 | 506 | WTX074-B09-156 | 17:44:23 | 2008-07-29 | 0 | 167000`. The experiment protocol instructed the subjects not to provide any assessment if they believed that the document was irrelevant. The dataset and the other details were described in [2].

Each display time was divided into non-overlapping ten-second intervals, thus obtaining a number of records for each event – the majority of the records do not have any assessment because only the last interval has been associated to an assessment. An access occurs when a user is interacting with a document within a given time interval. In this way, the number of accesses can be calculated for each event; for example, the user `user1` interacted the document WTX074-B09-156 in 17 distinct and consecutive intervals – the assessment was recorded only at the 17th interval.

Table 1 summarises our experimental results, where the interference term, i.e.,  $q(t) - p(t)$ , is shown in the last column. It was supposed that  $R = 0 \equiv r = 0$  and  $R = 1 \equiv r > 0$ . The table is truncated after ten intervals because the other intervals have low frequencies and most of the interactions ended before one hundred seconds.

The interference term at the early intervals when the proportion of accesses was the highest and, incidentally, when the user's interaction often ends with an assessment [8] is significant. This outcome signals that, even when it was supposed that the majority of users reach an agreement on relevance, QLP may reveal interference and then uncertainty in the user's state.

If our hypothesis is true, it can have some implications. First, the interference term can help model the evolution of the state while the user is interacting with the document. The presence of a Cosine in the interference term means that  $q(t)$  may be less or greater than  $p(t)$  at different  $t$ 's, thus helping model the evolution of the assessment in the user's mind.

**Table 1.** Interference across 10-second intervals

$t$	Interval	$q(t)$	$p(t)$	$p(t 0)$	$p(t 1)$	$p(t 2)$	$p(t 3)$	Interference	
0	0	10	0.192	0.179	0.228	0.108	0.074	0.153	7.3%
1	10	20	0.131	0.260	0.264	0.194	0.255	0.306	-49.6%
2	20	30	0.097	0.145	0.145	0.129	0.148	0.159	-33.1%
3	30	40	0.074	0.098	0.088	0.173	0.074	0.096	-24.5%
4	40	50	0.061	0.057	0.047	0.050	0.107	0.057	7.0%
5	50	60	0.052	0.039	0.042	0.043	0.027	0.038	33.3%
6	60	70	0.044	0.033	0.032	0.043	0.040	0.025	33.3%
7	70	80	0.036	0.032	0.027	0.029	0.040	0.051	12.5%
8	80	90	0.031	0.022	0.022	0.022	0.027	0.019	40.9%
9	90	100	0.027	0.016	0.013	0.007	0.040	0.013	68.8%

Second, the knowledge of the behaviour of the interference term is crucial to prediction. Indeed, if the system were able to measure the interference, it could predict when the state converges to the assessment.

Third, if the system could predict the interference term on the basis of some observed variables, it would be possible to predict the relevance assessment – the predicted assessment would be the closest vector  $|R = r\rangle$  to the vector  $|\phi\rangle$ .

Finally, interference can be seen as a mass of probability that may be distributed across the  $p(t|r)$ 's, thus changing the  $p(r|t)$ 's and then the prediction outcome and the ranking of suggested items. The modeling and the prediction using the QLP will be the focus of our future work.

## References

1. Liu, C., White, R.W., Dumais, S.: Understanding web browsing behaviors through weibull analysis of dwell time. In: Proc. of SIGIR, pp. 379–386 (2010)
2. Di Buccio, E., Melucci, M., Song, D.: Exploring combinations of sources for interaction features for document re-ranking. In: Proc. of HCIR (2010), <http://www.hcir2010.org/>
3. Zuccon, G., Azzopardi, L.: Using the quantum probability ranking principle to rank interdependent documents. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 357–369. Springer, Heidelberg (2010)
4. Bruza, P., Busemeyer, J.R., Gabora, L.: Special issue on quantum cognition. J. of Math. Psych. 53, 303–305 (2009)
5. Wang, J., Song, D., Zhang, P., Yuexian, H., Bruza, P.: Explanation of relevance judgement discrepancy with quantum interference. In: Proc. of QI, pp. 117–124 (2010)
6. Zhang, P., Song, D., Hou, Y., Wang, J., Bruza, P., Melucci, M., McCall, J.: Automata modeling for cognitive interference in users' relevance judgment. In: Proc. of QI, pp. 125–133 (2010)
7. van Rijsbergen, C.: The geometry of information retrieval. Cambridge University Press, UK (2004)
8. White, R.W., Kelly, D.: A study on the effects of personalization and task information on implicit feedback performance. In: Proc. of CIKM, pp. 297–306 (2006)



# Fusion vs. Two-Stage for Multimodal Retrieval

Avi Arampatzis, Konstantinos Zagoris, and Savvas A. Chatzichristofis

Department of Electrical and Computer Engineering,  
Democritus University of Thrace, Xanthi 67100, Greece  
{avi,kzagoris,schatzic}@ee.duth.gr

**Abstract.** We compare two methods for retrieval from multimodal collections. The first is a score-based fusion of results, retrieved visually and textually. The second is a two-stage method that visually re-ranks the top- $K$  results textually retrieved. We discuss their underlying hypotheses and practical limitations, and contact a comparative evaluation on a standardized snapshot of Wikipedia. Both methods are found to be significantly more effective than single-modality baselines, with no clear winner but with different robustness features. Nevertheless, two-stage retrieval provides efficiency benefits over fusion.

## 1 Introduction

Nowadays, information collections are not only large, but they may also be *multimodal*. Take as an example Wikipedia, where a single topic may be covered in several languages and include non-textual media such as image, sound, and video. Moreover, non-textual media may in turn be annotated.

We focus on two modalities, text and image. On the one hand, textual descriptions are key to retrieving relevant results for a topic, but at the same time provide little information about image content [5]. On the other hand, the visual content of images contains large amounts of information, which can hardly be described by words, making content-based image retrieval (CBIR) ineffective and computationally heavy in comparison to text retrieval. Thus, hybrid techniques which combine both worlds are becoming popular.

Traditionally, the method that has been followed in order to deal with multimodal databases is to search the modalities separately and fuse their results [4], e.g. with a linear combination of retrieval scores of all modalities per item. While fusion has been proven robust, we argue that it has a couple of issues: a) appropriate weighing of modalities and score normalization/combination are not trivial problems and may require training data, and b) if results are assessed by visual similarity only, fusion is not a theoretically sound method: the influence of textual scores may have a negative impact on the visual relevance of end-results.

An approach that may tackle the issues of fusion would be to search in a two-stage fashion: first rank with a secondary modality, draw a rank-threshold  $K$ , and then re-rank only the top- $K$  items with the primary modality. The assumption on which such a two-stage setup is based on is the existence of a primary modality (i.e. the one targeted and assessed by users) and its success would largely depend on the relative effectiveness of the two modalities involved. For example, if in the top- $K$ , text retrieval performs better

than CBIR, then CBIR is redundant. Thus, the underlying hypothesis is that CBIR can do better than text retrieval in the top- $K$  results retrieved by text.

Thresholding for two-stage retrieval can be performed statically (i.e. a fixed pre-selected threshold for all topics, e.g. [7]) or in a dynamic manner (i.e. a variable threshold optimizing a pre-defined measure per topic, e.g. [2]). In recent literature, the effectiveness of static thresholding has been mixed. For instance, static thresholding was found to perform worse in mean average precision (MAP) than the text-only with pseudo relevance feedback baseline in [7] (but better than fusing image and text modalities by a weighted-sum). However, others found that two-stage retrieval with dynamic thresholding is more effective and robust than static thresholding, performing significantly better than a text-only baseline [2].

A possible drawback of two-stage setups is that visually relevant images with empty or very noisy text modalities would be completely missed, since they will not be retrieved by the first stage. Moreover, if there are any improvements compared to single-stage text-only or image-only setups, these will first show up on early precision since only the top results are re-ranked; MAP or other measures may improve as a side effect. Fusion does not have these problems.

Next, we provide an experimental comparison of fusion to two-stage retrieval. Although we argued theoretically against fusion, in view also of the underlying assumption, hypothesis and drawbacks of two-stage retrieval, a comparison of the effectiveness of the two methods is in order.

## 2 An Experiment on Wikipedia

In this section, we report on experiments performed on the ImageCLEF 2010 Wikipedia test collection, which consists of 237434 images associated with noisy and incomplete user-supplied textual annotations. There are 70 test topics, each one consisting of a textual and a visual part, with one or more example images. The topics were assessed by visual similarity to the image examples.

We index the images with two descriptors that capture global image features: the Joint Composite Descriptor (JCD) and the Spatial Color Distribution (SpCD) [3]. For text indexing and retrieval, we employ the Lemur Toolkit V4.11 and Indri V2.11 with the tf.idf retrieval model; tf.idf has been found to work well with the the dynamic thresholding method we will describe in Section 2.2 [1]. We use the default settings that come with these versions of the system except that we enable Krovetz stemming. We index only the English annotations, and use only the English query of the topics. We evaluate on the top-1000 results with MAP, precision at 10 and 20, and bpref.

### 2.1 Fusion of Modalities

Let  $i$  the index running over example images ( $i = 1, 2, \dots$ ) and  $j$  running over the visual descriptors ( $j \in \{1, 2\}$ ). Thus,  $DESC_{ji}$  is the score of a collection item against the  $i$ th example image for the  $j$ th descriptor. We normalize  $DESC_{ji}$  values with MinMax, taking the maximum score seen across example images per descriptor. Assuming that the descriptors capture orthogonal information, we add their scores per example image. Then, to take into account all example images, the natural combination is to assign to

each collection image the maximum similarity seen from its comparisons to all example images; this can be interpreted as looking for images similar to *any* of the example images. Incorporating text, again as an orthogonal modality, we add its contribution. Summarizing, the score  $s$  for a collection image against the topic is defined as:

$$s = (1 - w) \max_i \left( \sum_j \text{MinMax}(\text{DESC}_{ji}) \right) + w \text{MinMax}(\text{tf.idf}). \quad (1)$$

The parameter  $w$  controls the relative contribution of the two media; for  $w = 1$  retrieval is based only on text while for  $w = 0$  is based only on image. We report for five  $w$  values between 0 and 1.

### 2.2 Dynamic Two-Stage Retrieval

For dynamic thresholding, we use the Score-Distributional Threshold Optimization (SDTO) as described in [1]. The SDTO method fits a binary mixture of probability distributions on the score distribution (SD). For tf.idf scores, we used the *technically truncated* model of a normal-exponential mixture. The method normalizes retrieval scores to probabilities of relevance (prels), enabling the optimization of  $K$  for any user-defined effectiveness measure. Per query, we search for the optimal  $K$  in  $[0, 2500]$ . Thus, for estimation with the SDTO we truncate at the score corresponding to rank 2500 but use no truncation at high scores as tf.idf has no theoretical maximum.

We experiment with the SDTO by thresholding on prel. This was found in [2] to be more effective and robust than thresholding on estimated precision. Thresholding on fixed prels happens to optimize *linear utility measures* [6]. We report for five prel thresholds. The top- $K$  results are re-ranked using Equation 1 for  $w = 0$ .

### 2.3 Experimental Results

Table 1 presents the effectiveness of fusion and two-stage against text- and image-only runs. Irrespective of measure, the best parameter values are roughly at: 0.6666–0.8000

**Table 1.** Retrieval effectiveness for fusion and dynamic two-stage retrieval. The best results per measure and retrieval type are in boldface. Significance-tested with a bootstrap test, one-tailed, at significance levels 0.05 ( $\hat{\Delta}^\nabla$ ), 0.01 ( $\hat{\Delta}^\wedge$ ), and 0.001 ( $\hat{\Delta}^*$ ), against the text-only baseline.

	MAP	P@10	P@20	bpref	
text-only	.1293	.3614	.3307	.1809	
fusion $w$	.9000	.1380 $\hat{\Delta}^\nabla$	.3786 $\hat{\Delta}^\nabla$	.3414 $\hat{\Delta}^\nabla$	.1901 $\hat{\Delta}^\nabla$
	.8000	<b>.1410<math>\hat{\Delta}^\nabla</math></b>	.4029 $\hat{\Delta}^\nabla$	.3514 $\hat{\Delta}^\nabla$	.1955 $\hat{\Delta}^\nabla$
	.6666	.1403 $\hat{\Delta}^\nabla$	.4129 $\hat{\Delta}^\nabla$	<b>.3664<math>\hat{\Delta}^\nabla</math></b>	<b>.1969<math>\hat{\Delta}^\nabla</math></b>
	.5000	.1185 $\hat{\Delta}^\nabla$	<b>.4157<math>\hat{\Delta}^\nabla</math></b>	.3657 $\hat{\Delta}^\nabla$	.1758 $\hat{\Delta}^\nabla$
	.3333	.0767 $\hat{\Delta}^\nabla$	.3871 $\hat{\Delta}^\nabla$	.3329 $\hat{\Delta}^\nabla$	.1278 $\hat{\Delta}^\nabla$
	two-stage $\theta$	.9900	.1376 $\hat{\Delta}^\nabla$	.4286 $\hat{\Delta}^\nabla$	.3714 $\hat{\Delta}^\nabla$
.9500		.1390 $\hat{\Delta}^\nabla$	.4314 $\hat{\Delta}^\nabla$	.3771 $\hat{\Delta}^\nabla$	.1917 $\hat{\Delta}^\nabla$
.8000		<b>.1428<math>\hat{\Delta}^\nabla</math></b>	<b>.4443<math>\hat{\Delta}^\nabla</math></b>	<b>.3857<math>\hat{\Delta}^\nabla</math></b>	<b>.1959<math>\hat{\Delta}^\nabla</math></b>
.5000		.1405 $\hat{\Delta}^\nabla$	.4357 $\hat{\Delta}^\nabla$	.3821 $\hat{\Delta}^\nabla$	.1943 $\hat{\Delta}^\nabla$
.3333		.1403 $\hat{\Delta}^\nabla$	.4357 $\hat{\Delta}^\nabla$	.3807 $\hat{\Delta}^\nabla$	.1942 $\hat{\Delta}^\nabla$
image-only		.0107 $\hat{\Delta}^\nabla$	.0871 $\hat{\Delta}^\nabla$	.0871 $\hat{\Delta}^\nabla$	.0402 $\hat{\Delta}^\nabla$

for fusion's  $w$ , and 0.8000 for two-stage's  $\theta$ . Both methods perform significantly better than text-only and far better than image-only. On the one hand, two-stage achieves better results than fusion, but it has more variability across topics: fusion passes the test at lower significance levels (i.e. higher confidence). On the other hand, effectiveness is less sensitive to the values of  $\theta$  than the values of  $w$ : two-stage provides significant improvements in all measures for a wide range of thresholds (i.e. 0.3333–0.9900), while fusion can significantly deteriorate effectiveness for unsuitable choices of  $w$ .

### 3 Conclusions

We compared fusion to two-stage retrieval from multimodal databases and found that both methods are significantly better than text- and image-only baselines. Indicatively, the largest improvements in MAP against the text-only baseline are +9.0% and +10.4% for fusion and two-stage respectively, while the corresponding improvements in P@10 are +15.0% and +22.9%.

While two-stage performs better than fusion in 3 out of 4 measures, improvements are statistically non-significant at the 0.05 level. Further, both methods are robust in different ways: fusion provides less variability across topics but it is sensitive to the weighing parameter of the contributing media, while two-stage provides a much lower sensitivity to its thresholding parameter but has a higher variability. Nevertheless, two-stage has an obvious efficiency benefit over fusion: it cuts down greatly on costly image operations. Although we have not measured running times, only the 0.02–0.05% of the items (on average) had to be scored at the image stage. While there is some overhead for estimating thresholds, this offsets only a small part of the efficiency gains.

### References

1. Arampatzis, A., Kamps, J., Robertson, S.: Where to stop reading a ranked list: threshold optimization using truncated score distributions. In: SIGIR, pp. 524–531. ACM, New York (2009)
2. Arampatzis, A., Zagoris, K., Chatzichristofis, S.A.: Dynamic two-stage image retrieval from large multimodal databases. In: Clough, P., et al. (eds.) ECIR 2011. LNCS, vol. 6611, pp. 326–337. Springer, Heidelberg (2011)
3. Chatzichristofis, S.A., Arampatzis, A.: Late fusion of compact composite descriptors for retrieval from heterogeneous image databases. In: SIGIR, pp. 825–826. ACM, New York (2010)
4. Depeursinge, A., Muller, H.: Fusion techniques for combining textual and visual information retrieval. In: ImageCLEF: Experimental Evaluation in Visual Information Retrieval. Springer, Heidelberg (2010)
5. van Leuken, R.H., Pueyo, L.G., Olivares, X., van Zwol, R.: Visual diversification of image search results. In: WWW, pp. 341–350. ACM, New York (2009)
6. Lewis, D.D.: Evaluating and optimizing autonomous text classification systems. In: SIGIR, pp. 246–254. ACM Press, New York (1995)
7. Maillot, N., Chevallet, J.-P., Lim, J.-H.: Inter-media pseudo-relevance feedback application to imageCLEF 2006 photo retrieval. In: Peters, C., Clough, P., Gey, F.C., Karlgren, J., Magnini, B., Oard, D.W., de Rijke, M., Stempfhuber, M. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 735–738. Springer, Heidelberg (2007)

# Combination of Feature Selection Methods for Text Categorisation

Robert Neumayer<sup>1</sup>, Rudolf Mayer<sup>2</sup>, and Kjetil Nørvåg<sup>1</sup>

<sup>1</sup> Norwegian University of Science and Technology,  
Department of Computer and Information Science, Trondheim, Norway

{neumayer,noervaag}@idi.ntnu.no

<sup>2</sup> Vienna University of Technology,  
Institute of Software Technology and Interactive Systems, Vienna, Austria  
mayer@ifs.tuwien.ac.at

**Abstract.** Feature selection plays a vital role in text categorisation. A range of different methods have been developed, each having unique properties and selecting different features. We show some results of an extensive study of feature selection approaches using a wide range of combination methods. We performed experiments on 18 test collections and report a subset of the results.

## 1 Introduction

Feature selection is an essential technique to facilitate reduction in dimensionality which is a vital component of any text categorisation system. Indeed, most machine learning algorithms could not be applied at all without it. A range of methods have been suggested and evaluated to this end. A good overview and a comprehensive survey of the whole area is given in [4].

A recent and extensive empirical study of feature selection is performed in [1]. Here, the author compares a list of 11 (8 without modified methods) feature selection methods. The performance evaluation is done on 19 test collections of different size and difficulty. The author uses one-against-all classification and as such averages all results over 229 binary classification problems.

Feature selection combination was, for example, suggested in [3]. The authors selected feature selection methods based on ‘uncorrelatedness’ and presented results for two document collections. More experiments for text categorisation are reported in [2]. Experiments are done with four different feature selection methods and a test collection sampled from RCV1-v2. It is shown that certain combination methods improve peak R-precision and  $F_1$ . Both studies only partly work with benchmark collections and the results are difficult to compare.

## 2 Feature Selection Methods

We show the different feature selection methods we use in this paper in Table 1. If a method does not rely on previously assigned labels it is an unsupervised

**Table 1.** Feature selection methods used throughout the paper

Method		Explanation
Document Freq.	(DF)	The number of documents a term occurs in.
Inverse Document Freq.	(IDF)	The inverse of the document Freq.
Collection Freq.	(CF)	The total number of occurrences of a term.
Inverse Collection Freq.	(ICF)	The inverse of the collection frequency.
Term Freq. Document Freq. (TFDF)		A method based on thresholds for DF.
Information Gain	(IG)	A information theoretic method taking into account both negative and positive examples.
Mutual Information	(MI)	Another method from information theory.
Odds Ratio	(OR)	A probabilistic feature selection method.
Class Discrimination Value	(CDV)	OR variant targeted at multi-class problems.
Word Freq.	(WF)	The weighted number of occurrences per class.
$\chi^2$ statistic	( $\chi^2$ )	Statistical method based on the independence of features.
NGL-Coefficient	(NGL)	A $\chi^2$ variant only looking at positive examples.
Categorical Proportional Difference	(CPD)	Considers only positive examples.
GSS-Coefficient		Another simplified $\chi^2$ method.
Bi-Normal Separation	(BNS)	Incorporates the inverse standard distribution and both positive and negative classes.

method (methods belonging there are shown in the first part of the table), if it does it belongs to the category of supervised methods (shown in the second part of the table).

### 3 Combination Methods

In the following we show a range of ranking merging methods applicable to the problem of merging feature rankings generated by different methods. The available methods are listed in Table 2. The first part of the table lists method based on rank. The second and third part list methods based on value and on the round robin strategy, respectively.

### 4 Experiments

We used SVMs and five runs of four-fold cross validation. The results given are the macro averaged classification accuracies for both single methods and selected combinations. Based on the performance of the individual methods we chose the following combinations of feature selection methods (combined with the methods from Table 2) for our experiments: BNS,  $\chi^2$ , DF, GSS, IG, MI, TFDF, WF and OR. This selection presents a good cross-section of the methods listed above since they both belong to different categories of methods and have show to have

**Table 2.** Ranking Merging methods used

Method		Explanation
Highest Rank	(HR)	A feature's highest rank in all single rankings.
Lowest Rank	(LR)	The lowest of all rankings is used as final score.
Average Rank	(AR)	The average over all single ranks is used.
Borda Ranking Merging	(BRM)	Gives scores according to the length of the single rankings.
Condorcet Ranking Merging	(CRM)	Is a majoritarian method favouring the candidate beating every other candidate in pair-wise comparisons.
Reciprocal Ranking Merging	(RRM)	In this setting, the final score for a feature is the sum of 1 divided by the rank in the single rankings.
Divide by Max. then OR	(DMOR)	The average over all single feature values in this setting we normalise by the maximum.
Divide by Length then OR	(DLOR)	Normalisation is performed via dividing by the length of the vector.
Pure Round Robin	(RR)	One feature is added from each ranking in turn until the desired number of features is reached.
Top $N$ Ranking Merging	(Top $N$ )	The top $n$ features from each ranking in turn are added until enough features are collected.
Weighted $N$ Ranking Merging	(WN)	The first $n$ % are taken from the first ranking, the remaining $1 - n$ % are composed of the other rankings in equal parts.

good performance in other studies in the past and show minimal to negative correlation with each other (based on both rank coefficient and classification performance).

We use a set of categorisation problems also used for binary classification experiments in [1], which were initially used by Han and Karypis. The collections were already preprocessed by basic stemming and stop-word removal. However, we use the sets for multi-class classification.

We show only a selection of all experiments. The accuracies for the top 200 selected features for both the single methods and combinations in Table 3. We chose to show results for 200 features because it is low enough so the classification is well possible. The best result per data set is shown in bold letters, the best result per method/combination in italic font. Overall we see that the combination methods outperform the single methods only in some cases, and never by much. On the other hand, the combinations are never much worse than the best single method. There is neither any single type of aggregation which provides the best results. However, for 100, 200, 500, and 1000 features, the method with the best averaged results is a combination method, even though the performance increase is very small.

**Table 3.** Average Classification Accuracies for the Top 200 Features

	BNS	$\chi^2$	DF	GSS	IG	MI	OR	TFDF	WF	IG-BNS	IG-OR	OR- $\chi^2$
la1	71.32	85.41	82.91	85.94	<i>86.27</i>	85.17	70.86	83.75	82.18	86.15	<b>86.39</b>	85.77
la12	71.60	87.29	85.38	87.55	<i>88.23</i>	86.73	69.87	85.48	83.85	88.23	<b>88.40</b>	86.94
oh0	86.20	<i>87.74</i>	67.10	85.24	86.44	86.38	82.41	75.91	68.57	86.44	86.66	<b>88.00</b>
oh10	<b>78.13</b>	77.70	67.18	76.52	77.96	77.11	74.53	73.26	69.85	<i>77.98</i>	<i>77.98</i>	77.77
oh15	79.15	<b>80.53</b>	62.98	78.25	79.39	78.29	70.58	72.75	63.64	79.41	79.59	<i>80.24</i>
oh5	<b>85.56</b>	<i>84.47</i>	74.68	85.03	84.10	84.34	81.68	79.35	79.26	84.12	84.18	<i>84.47</i>
ohsc	<i>75.89</i>	<b>77.87</b>	70.29	76.92	<i>77.23</i>	77.64	61.62	72.31	75.23	<i>77.22</i>	<i>77.22</i>	77.05
la2	70.67	86.64	83.64	87.57	<i>88.27</i>	86.13	73.40	84.34	82.43	88.26	<b>88.38</b>	87.19
wap	66.72	73.82	72.92	76.72	<b>80.83</b>	75.04	76.67	72.83	62.42	80.62	<i>80.81</i>	77.50
fbis	73.50	76.09	71.73	75.84	<i>82.83</i>	75.36	78.19	75.06	72.84	82.83	<b>82.90</b>	81.79
re1	83.68	85.23	74.29	84.49	<i>86.80</i>	83.32	78.94	78.20	73.82	86.76	<b>87.04</b>	86.13
tr11	85.07	<b>86.95</b>	83.77	86.13	86.43	85.75	84.54	85.31	83.00	86.33	<i>86.52</i>	86.42
tr12	85.11	82.74	73.87	80.64	<i>85.56</i>	83.89	79.17	77.70	70.10	85.49	<b>85.69</b>	84.83
tr21	80.24	87.92	83.57	89.40	<i>94.23</i>	83.81	86.96	84.52	82.44	93.81	94.23	<b>95.06</b>
tr23	64.51	80.39	83.53	82.25	<i>86.27</i>	81.86	86.27	82.94	74.71	86.18	86.78	<b>89.51</b>
tr31	95.84	95.83	92.68	95.58	<b>96.78</b>	95.36	93.69	95.64	91.54	<i>96.76</i>	96.57	95.60
tr41	92.71	<i>95.56</i>	88.68	94.65	95.03	94.94	91.82	91.78	87.79	95.03	95.15	<b>95.88</b>
tr45	86.00	91.42	83.77	90.84	<i>92.49</i>	91.36	87.39	86.03	81.34	92.38	92.93	<b>93.45</b>

## 5 Outlook and Future Work

We performed extensive feature selection and classification experiments on 18 different multi-class text categorisation problems. Further we used a wide range of ranking merging methods for combining features from multiple methods. However, no combination showed to be generally superior to the best single methods. Future work will deal with presenting more results in an accessible way and assessing the feasibility of ensemble methods to increase classification performance.

## References

1. Forman, G.: An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research* 3, 1289–1305 (2003)
2. Olsson, J.S., Oard, D.W.: Combining feature selectors for text classification. In: *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM 2006)*, Arlington, Virginia, USA, November 6-11, pp. 798–799. ACM, New York (2006)
3. Rogati, M., Yang, Y.: High-performing feature selection for text classification. In: *Proceedings of the 11th ACM International Conference on Information and Knowledge Management (CIKM 2002)*, McLean, Virginia, USA, November 4-9, pp. 659–661. ACM, New York (2002)
4. Sebastiani, F.: Machine learning in automated text categorization. *ACM Computing Surveys* 34(1), 1–47 (2002)



# Time-Surfer: Time-Based Graphical Access to Document Content<sup>\*</sup>

Hector Llorens<sup>1</sup>, Estela Saquete<sup>1</sup>, Borja Navarro<sup>1</sup>, and Robert Gaizauskas<sup>2</sup>

<sup>1</sup> University of Alicante, Spain

{hlllorens,stela,borja}@dlsi.ua.es

<sup>2</sup> University of Sheffield, United Kingdom

R.Gaizauskas@dcs.shef.ac.uk

**Abstract.** This demonstration presents a novel interactive graphical interface to document content focusing on the time dimension. The objective of Time-Surfer is to let users search and explore information related to a specific period, event, or event participant within a document. The system is based on the automatic detection not only of time expressions, but also of events and temporal relations. Through a *zoomable* timeline interface, it brings users an *dynamic* picture of the temporal distribution of events within a document. Time-Surfer has been successfully applied to history and biographical articles from Wikipedia.

## 1 Introduction

The amount of digitized information has grown to reach human tractable limits. Information retrieval (IR) studies methods to retrieve the most relevant documents in a collection given a user search query. However, these documents are often large and contain much more than the information the user is interested in. Consequently, methods to help the user locate the information he wants within the top-ranked retrieved documents are of considerable interest. Current search engines show the most relevant snippets of each search result, but the value of the temporal dimension in IR has been highlighted [2]. Temporal information is present not only in document timestamps or metadata, like creation or modification date, but also within the document content in time and event expressions.

Taking these observations as a starting point, the objective of our work is to demonstrate an information access approach within a document, using advanced temporal representational and navigational techniques. Specifically, our approach, Time-Surfer, relies on identifying times, events and temporal relations in documents and then utilizes this information within a graphical interface to provide users with dynamic time-based access to texts.

## 2 Related Work

Recent research work demonstrates the importance of the temporal dimension in IR [24]. The majority of this work uses the timeline as a basic time

<sup>\*</sup> Paper supported by the Spanish Government, project TIN-2006-15265-C06-01.

representation [3,1]. A clear sign of its usefulness is Google’s timeline feature [4], and also Yahoo’s Time Explorer [2] based on SIMILE ([www.simile-widgets.org](http://www.simile-widgets.org)).

The cited approaches represent search results on a timeline using the time expressions contained in the documents or within metadata (e.g. date of publication). Only [1] represents the main events as reported in the content of the retrieved documents, but they are extracted manually by humans.

Our approach differs from these proposals in the following respects. First, Time-Surfer focuses on accurately representing all the temporally grounded events *contained within large documents* and provides search, comparison and navigation facilities including *dynamic zooming in time*. Secondly, Time-Surfer uses computational tools to *extract references to events, times and temporal relations*, making *the whole process automatic*.

### 3 Time-Surfer

The Time-Surfer architecture consists of three main steps (see Fig. 1). The first step involves the extraction of the temporal expressions, events and temporal relations from an input document. Recent advances in the field of temporal information extraction (IE) have resulted in the proliferation of systems capable of extracting such elements following the TimeML [6] annotation scheme. In this demonstration, we use the TIPSem system [5] to annotate the input documents.

The second step groups the events (TimeML) linked to the same time reference into *event groups*. The event class, location in the source text, participants and textual context are saved in a structured format for subsequent use.

Finally, this structured information is loaded into our interactive graphical interface. To develop this web-based interface, we adapted and extended the Flot jQuery library [3] with new representational, navigational and search features.

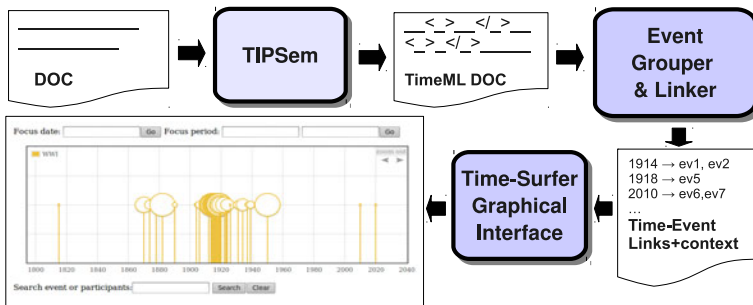


Fig. 1. Time-Surfer Architecture

<sup>1</sup> <http://newstimeline.googlelabs.com/>

<sup>2</sup> <http://fbmya01.barcelonamedia.org:8080/future/>

<sup>3</sup> <http://flot.googlecode.com>

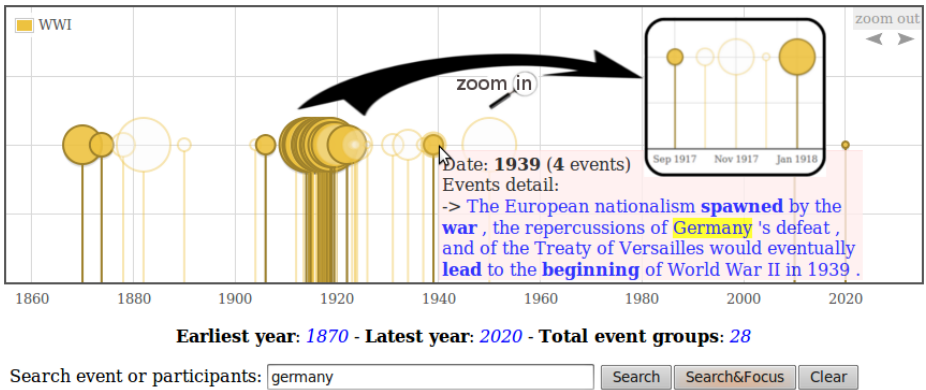


Fig. 2. Searching for “Germany” over WWI, hovering 1939, and zooming

The interface offers the following interactive capabilities:

- **Overview - Timeline:** Each event group is represented by a bubble whose position depends on the events’ time, and whose radius on the number of events it contains. This gives a general view of the temporal distribution of events in the document, illustrating time bounds and hot periods. Fig. 1 shows this view for the First World War (WWI) Wikipedia article.
- **Surfing and searching:** The interface lets the user navigate the timeline.
  - **Hovering:** When hovering an event group, detailed information is shown (time reference, number of events, list of the events in context).
  - **Clicking:** By clicking an event group, the previous information is presented in a pop-up layer. From there, the user can click on a specific sentence and go to the text of the original document.
  - **Zooming:** Using the mouse-wheel the user can zoom-in and out in time. This dynamic zooming allows the exploring of overlapping event groups.
  - **Panning:** The user can scroll backwards and forwards along the timeline intuitively by drag and drop.

Time-Surfer also includes forms to search a date (e.g., 1916) or period of focus (e.g., 1914-1918), and to search for events (e.g., “battle”), event participants (e.g., “Germany”) or participants relations (e.g., “Hitler - Mussolini”). If a query is introduced, bubbles containing relevant events are colored, while others are made semi-transparent. In the text detail, matched instances are highlighted as well as the sentences containing them (see Fig. 2).

- **Comparing documents in time:** The interface allows multiple documents to be compared in time, maintaining the previous dynamic features. Here event groups from each document are displayed using a distinct colour, but are positioned on the same timeline (see Fig. 3).

Time-Surfer has been applied to a set of history and biographical articles from Wikipedia in order to demonstrate the described features – see the on-line demo at <http://gplsi.dlsi.ua.es/demos/TIMEE/Time-Surfer/>

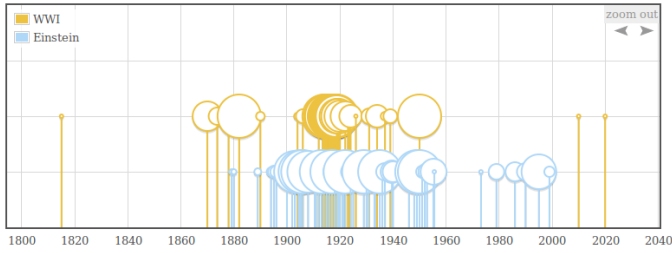


Fig. 3. WWI and Einstein searchable and navigable comparison

## 4 Analysis and Conclusions

This demonstration presents a novel dynamic user interface, Time-Surfer, that enables time-based access to document content. The system lets users search and navigate, through zooming and panning, the information via a combined graphical and textual representation of the temporal content of a text.

The strengths of the approach include: (1) the interactive interface helps users to *rapidly understand the temporal setting of a document* – the output clearly marks starting and ending time points, as well as the areas of concentration of event groups; (2) such concentration areas can be explored using the *zooming facility* which is missing in related works; (3) the *temporal IE, including events, is automatic*; (4) the search facilities make it easy to find information related to a specific date, period, event or event participant; (5) the *document comparison capability* supports finding relations in time between events from different documents. Limitations of the approach are: (1) substantial processing is needed to extract the temporal content from each document, requiring off-line preprocessing; (2) temporal IE performance is not perfect (85% approx).

In the future we expect both the speed and accuracy of temporal IE to improve, given that it has become the focus of an international research effort. Regarding Time-Surfer, a user-centered evaluation, ideally task-based, needs to be carried out, both to better understand the strengths and weaknesses of the system and to gather further ideas for refining the interface.

## References

1. Alonso, O., Berberich, K., Bedathur, S., Weikum, G.: NEAT: News exploration along time. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., van Rijsbergen, K. (eds.) ECIR 2010. LNCS, vol. 5993, pp. 667–667. Springer, Heidelberg (2010)
2. Alonso, O., Gertz, M., Baeza-Yates, R.: On the Value of Temporal Information in Information Retrieval. SIGIR Forum 41(2), 35–41 (2007)
3. Alonso, O., Gertz, M., Baeza-Yates, R.: Search Results using Timeline Visualizations. In: SIGIR, pp. 908–908. ACM, New York (2007)

4. Berberich, K., Bedathur, S., Alonso, O., Weikum, G.: A language modeling approach for temporal information needs. In: Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Røelleke, T., Røger, S., van Rijsbergen, K. (eds.) *ECIR 2010*. LNCS, vol. 5993, pp. 13–25. Springer, Heidelberg (2010)
5. Llorens, H., Saquete, E., Navarro-Colorado, B.: TIPSem (English and Spanish): Evaluating CRFs and Semantic Roles in TempEval-2. In: *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 284–291. ACL (2010)
6. Pustejovsky, J., Castaño, J.M., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., Katz, G.: TimeML: Robust Specification of Event and Temporal Expressions in Text. In: *IWCS-5* (2003)

# ARES: A Retrieval Engine Based on Sentiments

## Sentiment-Based Search Result Annotation and Diversification

Gianluca Demartini\*

L3S Research Center, Appelstr. 9a, 30167 Hannover, Germany  
demartini@L3S.de

**Abstract.** This paper introduces a system enriching the standard web search engine interface with sentiment information. Additionally, it exploits such annotations to diversify the result list based on the different sentiments expressed by retrieved web pages. Thanks to the annotations, the end user is aware of which opinions the search engine is showing her and, thanks to the diversification, she can see an overview of the different opinions expressed about the requested topic. We describe the methods used for computing sentiment scores of web search results and for re-ranking them in order to cover different sentiment classes. The proposed system, built on top of commercial search engine APIs, is available on-line<sup>1</sup>.

## 1 Introduction

As the Web is becoming more accessible and, thus, social, more and more people are able to express their opinions creating textual content (e.g., writing blog postings). Moreover, as the amount of content available on the Web grows, search engines become an essential tool for users to enter the Web and to find information. For the same reasons, the number of pages relevant to a query is growing over time, forcing users to trust the search engine ranking as they can not read the entire result set. Moreover, as hundreds of features are considered, it is not possible to control algorithmic search results. Thus, it may happen that search results computed using features like popularity (e.g., PageRank), textual similarity (e.g., BM25), or topical diversity are biased towards a certain opinion. For example, most of the top results for the query ‘termination of pregnancy’ run against the Google search engine are medical pages providing technical definitions while there is no page providing information about anti-abortion movements. This motivates the study of sentiments expressed in search results: how can we explicitly show the user such sentiments? How can we enrich current search engine result pages providing an overview of different sentiments expressed in top N search results?

---

\* This work is partially supported by the EU Large-scale Integrating Projects Living-Knowledge (contract no. 231126) and Arcomem (contract no. 270239).

<sup>1</sup> <http://ares.L3S.uni-hannover.de>

Show:  only positive  only negative  all diversified  all results

**0.24** [Death Penalty Information Center](#)  
 Provides state-by-state information on executions, history of the **death penalty**, discusses mental retardation, race, innocence, deterrence, and botched execution.  
<http://www.deathpenaltyinfo.org>

**-0.61** [Capital Punishment - Wikipedia](#)  
 User-created article about the **death penalty**, exploring the history of the judicially-ordered execution of a prisoner as punishment for a serious crime.  
[http://en.wikipedia.org/wiki/Capital\\_punishment](http://en.wikipedia.org/wiki/Capital_punishment)

**-0.03** [Death Penalty Focus](#)  
 Dedicated to the abolition of capital punishment through grassroots organizing, research, and the dissemination of information about the **death penalty** and its alternatives.  
<http://www.deathpenalty.org>

**0.41** [Part I: History of the Death Penalty | Death Penalty ...](#)  
 1907-1917 - Nine states abolish the **death penalty** for all crimes or strictly limit it. ... A Gallup poll shows support of the **death penalty** at only 42 ...  
<http://www.deathpenaltyinfo.org/article.php?scid=15&did=410>

**0.03** [Pro-Death Penalty.com](#)  
 Includes articles, commentary and interviews; information about appeals and legislation; and a list of scheduled executions.  
<http://www.prodeathpenalty.com>

1 2 3 4 5 6 7

**Fig. 1.** A screenshot of the proposed system for the query ‘death penalty’. Official governmental pages tend to show a positive bias toward the topic. Other pages (e.g., from Wikipedia) show a negative bias as they also talk about religious views and public opinion. This shows how such topic can be controversial as different sources express different opinions.

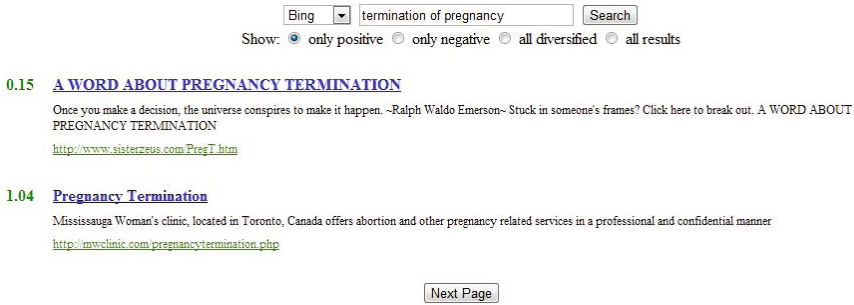
We propose a system that, building on top of current commercial search engines, provides sentiment annotation of search results, the possibility to re-rank the results in order to see an overview of different sentiments expressed on the Web about the requested topic, and the option to filter out positive/negative results. Figure 1 shows how sentiment annotations are displayed to the end user. Figure 2 is a screenshot of the system when the user asks only for positive results.

The following section describes the different components of our system that allows 1) to assign a sentiment score to a web page with respect to the user query, 2) to retrieve only positive/negative results, and 3) to diversify search results based on sentiment annotations.

## 2 System Components

### 2.1 Sentiment Estimation

As described in [1], there may be different techniques for estimating the sentiment expressed in web search results. The task of polarity detection in the context of blog postings has also been widely studied (see [4] for a survey). In our system we exploit a sentiment classifier based on linear SVMs. We trained our classifier on a collection of 18,142 manually judged opinionated blog postings about different topics from the polarity task at the TREC 2008 Blog Track [5].



**Fig. 2.** A screenshot of the proposed system for the query ‘termination of pregnancy’ and for positive results. The results contain information about techniques and places where to terminate pregnancy rather than discussing ethical issues about the topic.

A possible way to measure the confidence of SVMs (with positive or negative sign) is the distance of a vector from the hyperplane which separates different classes. We use this value as an estimation of the sentiment expressed by a sentence about the user query. We define  $Sent(d)$  as the set of sentences of a document  $d$  which can be obtained using extraction tools<sup>2</sup>. In the current version of the system we compute  $s_q(d)$  the estimation of the sentiment expressed by a document  $d$  about a query  $q$  based exclusively on the sentences containing the query terms in the following way:

$$s_q(d) = \frac{\sum_{st \in Sent_q(d)} polarity(st)}{|Sent_q(d)|} \quad (1)$$

where  $Sent_q(d) \subset Sent(d)$  is the set of sentences containing all query terms from document  $d$ , and  $polarity(st)$  is the estimation of the expressed sentiment computed as described above. Then, we can compute the score of a document by computing the average over all the sentences about the user query.

Another option our system includes is to estimate sentiment scores based on a lexicon of opinionated terms [1] (e.g., SentiWordNet [2] contains terms that are assigned to three classes: positive, negative, and objective). As the average of computed scores may confuse the user, an implemented alternative method is to estimate the sentiment of a document as  $s(d) = +1$ ,  $s(d) = 0$ , or  $s(d) = -1$  aggregating the scores computed over the sentences. In this way we can assign to each web search result one of the three considered sentiment classes.

## 2.2 Sentiment Diversification

Diversification algorithms in the context of Web Search have been mainly focusing on the topical dimension. Our system diversifies search results based on the expressed sentiment. We exploited the xQuAD diversification framework [6], which was one of the top performers in the TREC 2009 and 2010 Web tracks, in

<sup>2</sup> In our implementation we use GATE: <http://gate.ac.uk/>



order to diversify search results. In detail, the maximization objective function in xQuAD is defined as follows:

$$(1 - \lambda)P(d|q) + \lambda P(d, \bar{S}|q) \quad (2)$$

where  $P(d|q)$  is, in our system, estimated by the original search engine ranking while  $P(d, \bar{S}|q) = \sum_{q_i \in q} P(q_i|q)P(d, \bar{S}|q_i)$  represents the diversity component. The  $P(q_i|q)$  component models the importance of different sub-topics  $q_i$  while  $P(d, \bar{S}|q_i) = P(d|q_i)P(\bar{S}|q_i)$  represents the *coverage* of  $d$  on different sub-topics  $q_i$  and the *novelty* with respect to the current diversified ranking  $S$ . For sentiment diversification, instead of considering sub-topics  $q_i$  of a query  $q$ , we consider 3 sentiment classes (i.e.,  $s_i \in \{pos, obj, neg\}$ ) to be the query aspects we want to cover within the final ranking. We compute the estimated  $P(d|s_i)$  based on the scores provided by the sentiment classification method. Thus, we can apply the xQuAD framework for sentiment diversification of search results.

### 3 System Implementation

The developed system is based on three different search services provided by Bing, Google, and Yahoo!. After submitting the user query to the selected search service, the system obtains the list of retrieved URLs. Then, it fetches the HTML code associated to each URL and applies template removal techniques [3] in order to obtain the plain content of the page for sentiment classification. The system then exploits sentiment estimation techniques for obtaining annotation scores to be shown to the user. In case the user wants to see a diversified ranking based on the sentiment dimension, the system exploits such annotations to re-rank top 10 results based on estimated probabilities of a document given a sentiment class in the xQuAD framework. The user can also filter search results in order to see only pages with a positive or negative point-of-view. Note that all these steps are run at query time, that is, no preprocessing of documents is done before the user has issued the query.

### References

1. Demartini, G., Siersdorfer, S.: Dear Search Engine: What's your opinion about...? - Sentiment Analysis for Semantic Enrichment of Web Search Results. In: Semantic Search 2010 Workshop located at the 19th Int. World Wide Web Conference WWW 2010 (2010)
2. Esuli, A., Sebastiani, F.: SENTIWORDNET: A publicly available lexical resource for opinion mining. In: In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC 2006), pp. 417–422 (2006)
3. Kohlschütter, C., Fankhauser, P., Nejdl, W.: Boilerplate detection using shallow text features. In: WSDM, pp. 441–450 (2010)
4. Macdonald, C., Santos, R.L.T., Ounis, I., Soboroff, I.: Blog track research at trec. SIGIR Forum 44(1), 58–75 (2010)
5. Ounis, I., Macdonald, C., Soboroff, I.: Overview of the TREC 2008 Blog Track. In: TREC (2008)
6. Santos, R.L.T., Macdonald, C., Ounis, I.: Exploiting query reformulations for web search result diversification. In: WWW, pp. 881–890 (2010)

# Web Search Query Assistance Functionality for Young Audiences

Carsten Eickhoff<sup>1</sup>, Tamara Polajnar<sup>2</sup>, Karl Gyllstrom<sup>3</sup>, Sergio Duarte Torres<sup>4</sup>,  
and Richard Glassey<sup>2</sup>

<sup>1</sup> Delft University of Technology

`c.eickhoff@tudelft.nl`

<sup>2</sup> University of Glasgow

`{tamara,rjg}@dcs.gla.ac.uk`

<sup>3</sup> Katholieke Universiteit Leuven

`karl.gyllstrom@cs.kuleuven.be`

<sup>4</sup> University of Twente

`duartes@cs.utwente.nl`

**Abstract.** The Internet plays an important role in people’s daily lives. This is not only true for adults, but also holds for children; however, current web search engines are designed with adult users and their cognitive abilities in mind. Consequently, children face considerable barriers when using these information systems. In this work, we demonstrate the use of query assistance and search moderation techniques as well as appropriate interface design to overcome or mitigate these challenges.

## 1 Introduction

Today, children are frequently consulting the Internet’s search facilities to satisfy their personal information needs. However, the current popular web search engines hardly reflect the specific demands of very young users. Throughout the last 10 years, various studies have identified the typical challenges children face when interacting with search engines. In this work we will summarize their findings and show how targeted means of Information Retrieval and interface design can be used to increase success and enjoyment of young searchers. We focus mainly on children between 8 and 12 years as they already show a sufficient degree of literacy skills to operate and understand textual search interfaces. The following challenges have been observed for members of said age group. **(A)** The first and often most frustrating problem for children is the query formulation step [93]. Children typically have a far smaller active vocabulary than adults. Therefore, they often do not know the exact term to describe their information needs, or, if they do, they will not always be able to spell it correctly.

**(B)** Once the query has been issued the child has to identify those results that are relevant to her or his search interest. Children often struggle with the task of understanding extensive textual result snippets. The cognitive load of interpreting and comparing several multi-sentence snippets including gaps, urls, statistics, etc. appears to be substantial, and forms one of the main challenges at

this point. **(C)** Besides the individual length and degree of detail at which every single result is presented, the overall number of retrieved results per page that children can handle is lower than for adults [8]. This is additionally underlined by the fact that children rarely scroll down the result page. Often they are not even aware of the functionality or at least do not use it intuitively. **(D)** A specific habit that has often been observed for children is a preference for browsing over searching [4]. Where adult users often explore a given topic by iteratively respecifying the search query, children tend to browse through the results of the initial query to find the desired pieces of information. **(E)** There is a large proportion of the content offered on the Internet that is not suitable or not understandable for children. Even factoring out erotic content, which state of the art search engines can reliably detect, many pieces of information are potentially harmful for children. Depending on the specific query the proportion of unsuitable material varies strongly. **(F)** Finally, the Internet and especially the search engine niche have experienced growing influences from the advertisement industry. While adult users rarely fall for the advertisers' tricks such as pop-ups or banners, children are less resistant to such marketing methods and require appropriate protection in this domain [10].

## 2 Functionality Overview

Keeping the previously introduced challenges for children's web search in mind, we propose a system, based on the PuppyIR framework [2], that allows for easy modular combination of web search services with focus on child audiences. The system is based on a conventional web search engine, augmented by additional pre- and post-processing, to alter both, the issued queries, as well as the returned result lists. In the following we will describe the concrete measures taken to address children's specific needs in web search scenarios. We will refer to the challenge indices from Section 1.

### 2.1 Faceted Query Expansion

Following the idea of faceted content exploration, we provide a means of expanding queries in such a way that they focus on one specific angle of the search topic. The CollAge system [7], introduced the use of media types commonly consumed by children as facets for exploration. Examples of media types are colouring pages, puzzles, cartoons, or games. In order to make these categories more easily understandable for children, a visual representation is given. This is typically an example of the things to be found in the respective category. Query expansions address children's problems with query formulation (A), by providing easy means of redirecting the query's focus without having to manually reformulate the query, which has been found to be frustrating. Additionally, it allows for a better coverage of browsing strategies (D).

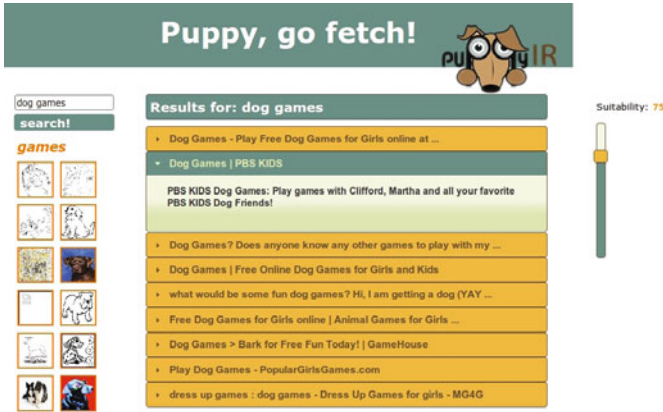


Fig. 1. Query assistance interface

## 2.2 Community-Created Query Expansion

A second source of query expansion terms is based on human expertise in the form of social bookmarking tags. A given query is issued to a number of independent search engines and for each of them the  $n$  top-ranked results are looked up on the social bookmarking platform *Del.icio.us*. The most frequently assigned tags over all retrieved pages are used as query expansion options. In a second step *Wikipedia* and the *ODP* web taxonomy [1] are used to infer high level semantic categories from the tags. These semantic concepts can also be offered as expansion candidates. Similar to faceted query expansion, the community-based approach eases query formulation (A) and, by diversifying the scope of results, enhances the success rate of browsing (D).

## 2.3 Content Moderation

In order to ensure the suitability of retrieved results for the user's age we offer a result filter, that follows the approach of Eickhoff et al. [5], to automatically estimate each web page's suitability based on a wide range of on-page features. Prominent examples are reading level scores of textual content, language modelling approaches, a high-level syntactical analysis of the way the user is addressed on the page, an estimate of the page's commercial intent as well as an analysis of the page's link neighbourhood. The resulting suitability score ranges from 0 to 1, where 1 indicates a page is definitely suitable for children. A customisable threshold value of this score enables fine-grained system tuning towards the user's personal preferences. In addition to eliminating topically unsuitable results (E), the content moderation step allows the filtering of strongly commercially motivated pages (F).

## 2.4 Search Interface

A number of potential problems for young users arise from web search interfaces that are mainly designed for adult users. Our interface (see Figure 1) follows the

paradigms detailed by Glassey et al. [6]. In order not to overwhelm the child with the amount of text (B), we only show web page titles that are expandable to show the full original snippet. The number of search results per page is limited as to eliminate the need to scroll (C). A side panel shows possible faceted routes of content exploration. Children can be very sensitive to colourful designs [10]. To improve the user's experience and enjoyment, different graphical styles can be chosen for the interface.

### 3 Demonstrator Requirements

While the demonstrator can be easily adapted to work based on a closed collection of documents, the query suggestion mechanics, however, rely on on-line services such as Del.icio.us. Therefore, an Internet connection would be required for our demonstration.

### Acknowledgement

We would like to express our thanks to Franciska de Jong, Leif Azzopardi, Arjen de Vries and Djoerd Hiemstra for their counsel and advice to this demonstrator. This research is part of the PuppyIR project. It is funded by the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement no. 231507.

### References

1. The Open Directory Project (2010), <http://www.dmoz.org/>
2. Azzopardi, L., Glassey, R., Lalmas, M., Polajnar, T., Ruthven, I.: *PuppyIR: Designing an Open Source Framework for Interactive Information Services for Children* (2009)
3. Borgman, C.L., Hirsh, S.G., Walter, V.A., Gallagher, A.L.: *Children's searching behavior on browsing and keyword online catalogs: the Science Library Catalog project*. *JASIS* 46(9) (1995)
4. Druin, A., Foss, E., Hutchinson, H., Golub, E., Hatley, L.: *Children's roles using keyword search interfaces at home*. In: *CHI 2010*. ACM, New York (2010)
5. Eickhoff, C., Serdyukov, P., de Vries, A.P.: *Web Site Classification on Child Suitability*. In: *CIKM 2010*, Toronto, Canada (2010)
6. Glassey, R., Elliott, D., Polajnar, T., Azzopardi, L.: *Interaction-based information filtering for children*. In: *IiX 2010*. ACM, New York (2010)
7. Gyllstrom, K., Moens, M.F.: *A picture is worth a thousand search results: finding child-oriented multimedia results with collAge*. In: *SIGIR 2010*, Geneva, Switzerland (2010)
8. Large, A., Beheshti, J., Rahman, T.: *Design criteria for children's Web portals: The users speak out*. *JASIST* 53(2), 79–94 (2002)
9. Moore, P.A., St George, A.: *Children as Information Seekers: The Cognitive Demands of Books and Library Systems*. *School Library Media Quarterly* 19(3), 161–168 (1991)
10. Nielsen, J.: *Kids corner: Website usability for children*. *Jakob Nielsens Alertbox* (2002)

# Conversation Retrieval from Twitter

Matteo Magnani<sup>1</sup>, Danilo Montesi<sup>1</sup>, Gabriele Nunziante<sup>1</sup>, and Luca Rossi<sup>2,\*</sup>

<sup>1</sup> Dept. of Computer Science, University of Bologna

{magnanim,montesi,nunziant}@cs.unibo.it

<sup>2</sup> Dept. of Communication Studies, University of Urbino Carlo Bo

luca.rossi@uniurb.it

**Abstract.** The process of retrieving conversations from social network sites differs from traditional Web information retrieval because it involves human communication aspects, like the degree of interest in the conversation explicitly or implicitly expressed by the interacting people and their influence/popularity. Our demo allows users to include these aspects into the search process. The system allows the retrieval of millions of conversations generated on the popular Twitter social network site, and in particular conversations about trending topics.

**Keywords:** Conversation retrieval, Twitter.

## 1 Introduction

Today a growing amount of online information is produced by users (User Generated Content) and published on social network sites. Social motivations laying below this process are various and go from the users' need for information to their desire for social interaction such as *online conversations* or *online dating*.

Even if it is possible to gather many different services under the Web 2.0 label it is important to highlight that each service shows its own peculiarities and deserves to be understood and studied according to those. Twitter, the service we are dealing with in our demo, belongs to the sub-category of microblogging sites: services that allow users to share short text messages (tweets) with a defined group of users called *followers*. Users can reply to each other simply by adding a @ sign in front of the name of the user they are replying to. This fairly simple set of socio-technical rules has made possible for Twitter to host a wide range of social interactions [6] from the *broadcasting* of personal thoughts to a large public to more structured *conversations* among groups of friends. The analysis of those communications can be useful from many points of view (from marketing research to political consensus analysis) but in order to be fruitful it requires to take into careful consideration not only the textual relevance of the searched keywords but also the social relationships existing among the users involved in the conversation.

In our demo we provide a full conversation retrieval system extracting conversations about trending topics from Twitter and providing a query system where

---

\* This work has been partly funded by Telecom Italia.

users can specify the impact of social and communication aspects on the ranking of conversations.

## 2 Conversation Retrieval Model

The concept of *conversation retrieval for social network sites*, has been introduced in [10] and builds over previous research on structured [8,9,7,3,2,4], hypertext and Web information retrieval [1,5].

In our demo system a conversation is modeled as a tree where nodes represent short *text messages* posted by a *user* at specific *timestamps* in *reply* to their parent nodes, as exemplified in Figure 1. The ranking of a conversation depends on all these aspects, as follows.

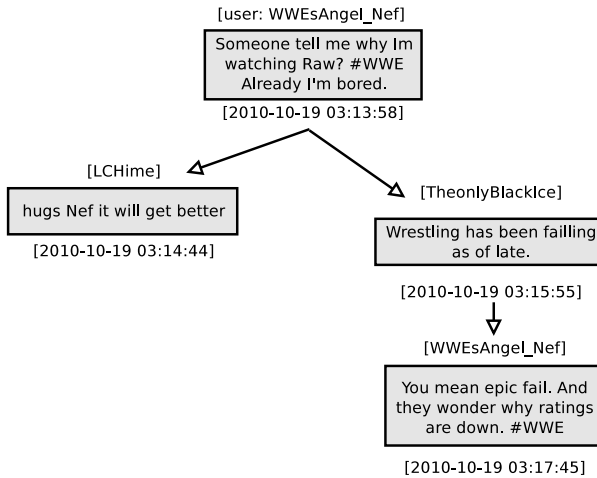
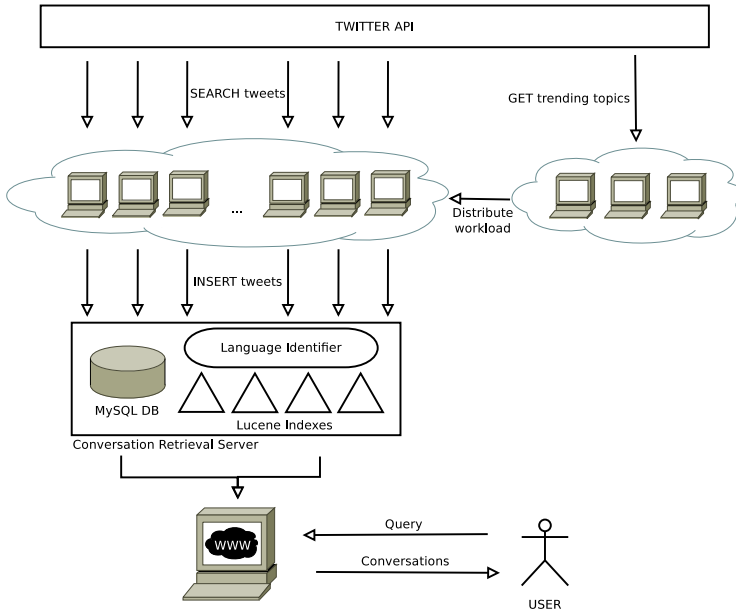


Fig. 1. Part of a Twitter conversation composed of 4 tweets

The **text relevance** of a conversation is obviously one of the ranking criteria. However, the same tweets posted by different users may have different degrees of importance — for instance a tweet about a product or brand assumes special importance in a brand monitoring system if it has been posted by a well known blogger. We will thus use a concept of **popularity** of users and conversations. In addition, the same tweet posted at different times may be more or less important — for example, a five-year-old tweet can often be regarded as less important than very recent news. We call this the **timeliness** of a conversation. Moreover, the rate at which tweets are exchanged can be indicative of the level of interest/emotion attached to the conversation — in the following this aspect is indicated as the **density** of a conversation. Finally, the number of tweets exchanged during a conversation (**size**) and the number of participating users (**audience**) can also be regarded as ranking criteria.



**Fig. 2.** Architecture of the Conversation Retrieval Demo

In our demo the text relevance is computed using an existing IR engine, and size and audience are computed by analyzing the tweets composing the conversation. As measures of user popularity we consider the number of followers and the ratio between posted tweets and replies received, while we use the number of retweets as a measure of popularity of a conversation, i.e., how many times its tweets have been shared by other users. The density of a conversation is computed as the average inverse time interval between tweets. Finally, the timeliness is computed by comparing the timestamps of the tweets with the date specified in the query.

### 3 Conversation Retrieval System

In Figure 2 we have illustrated the architecture of our system. A small number of programs periodically get a set of keywords from the Twitter API<sup>1</sup> indicating the trending topics, i.e., the most discussed topics at that instant. These topics are then distributed to several programs retrieving related tweets using the Twitter Search API<sup>2</sup>. Whenever a tweet indicates that it is part of a reply chain, all the chain of tweets is collected.

The extracted tweets and conversations are then uploaded to a conversation retrieval server. This server hosts a relational database management system (MySQL) to store tweets and other information like the number of followers of

<sup>1</sup> <http://dev.twitter.com/doc>

<sup>2</sup> <http://dev.twitter.com/doc/get/search>



every user participating to the conversations. The IR engine Lucene<sup>3</sup> is used to index the text of the conversations and to associate it to their identifiers in the database, from which they can be later efficiently retrieved.

At this point users can ask queries through a web interface where they can specify how much each of the aforementioned social aspects should be weighted in computing the ranking of the result. While we are currently evaluating some predefined query profiles to ease the specification of the search parameters, one important feature of the system is that users can change these values to improve the result of previous research tasks by analyzing the retrieved conversations and updating the weights of the ranking criteria.

## 4 Demo Session and Concluding Remarks

During the demo session users will be able to query directly the system and tune the ranking parameters to evaluate the impact of different social aspects on the results of their search tasks. One of the innovative features of the system and the underlying theoretical model that will be appreciated during the session is the focus on user social relations in addition to the traditional emphasis on words and document relationships.

## References

1. Agosti, M., Smeaton, A.F.: Information retrieval and hypertext. Kluwer Academic, Boston (1996)
2. Amer-Yahia, S., Botev, C., Shanmugasundaram, J.: Texquery: a full-text search extension to XQuery. In: WWW (2004)
3. Amer-Yahia, S., Fernandez, M.F., Srivastava, D., Xu, Y.: Phrase matching in XML. In: Proceedings of the International Conference on Very Large Data Bases (2003)
4. Amer-Yahia, S., Lakshmanan, L.V.S., Pandit, S.: Flexpath: Flexible structure and full-text querying for xml. In: SIGMOD Conference (2004)
5. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Computer Networks and ISDN Systems, pp. 107–117 (1998)
6. Danah, B., Scott, G., Gilad, L.: Tweet, Tweet, Retweet: Conversational Aspects of Retweeting on Twitter. In: HICSS-43, January 6. IEEE, Kauai (2010)
7. Fuhr, N., Großjohann, K.: XIRQL: A query language for information retrieval in XML documents. In: SIGIR Conference (2001)
8. Fuhr, N., Rölleke, T.: A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems* 15(1), 32–66 (1997)
9. Lalmas, M.: Dempster-Shafer's theory of evidence applied to structured documents: modelling uncertainty. In: Proceedings of the 20th Annual International ACM SIGIR Conference, pp. 110–118. ACM Press, New York (1997)
10. Magnani, M., Montesi, D.: Toward conversation retrieval. In: Agosti, M., Esposito, F., Thanos, C. (eds.) IRCDL 2010. Communications in Computer and Information Science, vol. 91, pp. 173–182. Springer, Heidelberg (2010) isbn: 978-3-642-15849-0

<sup>3</sup> <http://lucene.apache.org>

# Finding Useful Users on Twitter: Twittomender the Followee Recommender\*

John Hannon, Kevin McCarthy, and Barry Smyth

CLARITY: Centre for Sensor Web Technologies,  
School of Computer Science & Informatics,  
University College Dublin, Ireland  
`{firstname.surname}@ucd.ie`

**Abstract.** This paper examines an application for finding pertinent friends (followees) on Twitter<sup>1</sup>. Whilst Twitter provides a great basis for receiving information, we believe a potential downfall lies in the lack of an effective way in which users of Twitter can find other Twitter users to follow. We apply several recommendation techniques to build a followee recommender for Twitter. We evaluate a variety of different recommendation strategies, using real-user data, to demonstrate the potential for this recommender system to correctly identify and promote interesting users who are worth following.

## 1 Introduction

Twitter has proven to be one of the most surprising success stories of recent web history. To get the most from Twitter, users need to follow others so that they can benefit from the tweets of these followees. But who should a user follow, beside their immediate friends? Helping users to find new people to follow is an important challenge and the focus of this paper. Twitters' own recommendation system utilises two main approaches when attempting to create new connections between users. Firstly, they use the categorisation approach, this aims to box-off celebrities or popular users into categories such as Entertainment, Music, etc. for users to select from. This approach, whilst appropriate for initial connections for new users, may become less useful as users engage with the system and find that forming connections is usually easier by reading content from Re-Tweeted (forwarded) tweets from users they are not connected to or from other online resources relevant to them that have associated Twitter accounts. Twitter's second approach uses collaborative filtering techniques. Much research has been carried out using collaborative filtering to aid in filling in the missing links, be it a rating on a movie review site [4] or in this case filling in the links in ones social graph (see also [1,2]). The main idea behind this approach is that your friend's friend has the potential to be your friend. These approaches alone do not fully allow

---

\* This work is supported by Science Foundation Ireland under grant 07/CE/I1147 and by Amdocs Inc.

<sup>1</sup> <http://www.twitter.com>

for the best or most relevant recommendations for a user to be produced and neither does sifting through the Twitter public timeline of thousands of tweets per second in the hope of finding someone of interest. The collaborative approach is good at filling in the missing links, but what happens if we want to find users with similar interests outside of our extended social circle? This is where *Twittomender* [3] comes in, helping its users in finding useful followees, who produce relevant content, and preventing information overload. Here, we examine the proven collaborative style approaches and also look at the content contributed to Twitter – we utilise users tweets as a source of information to compare against other users and produce a content-based recommendation system [5].

## 2 Twittomender: Recommending Users to follow

Twittomender has been developed as web service<sup>2</sup>. Twittomender mines a large number of user profile details and content (tweets, followee/follower user ids) by following the links between users on the social graph with the aid of Twitter's API<sup>3</sup>. When a new user is looking for followees, Twittomender can recommend profiles from our database of over 1 million Twitter users. Twittomender allows users to input a search query to find interesting potential followees or indeed a search query can be derived implicitly from that users own tweets. The terms of this query relate to the types of content the user would like to consume. These terms are used to find users who have tweeted these key terms frequently. These Twitter profiles are stored as documents and indexed by Lucene<sup>4</sup>. By using Lucene, a document search engine, we can search these documents in an information retrieval style by using queries to find matching documents. Twittomender uses 7 recommendation strategies, these include the content (tweets) and connections (ids) of a user. These fields are queried against the document corpus for similarity. Twittomender's recommendation strategies are comprised of two recommendation categorisations; **Content-based**: (1) *Users own tweets*, (2) *Followee's tweets*, (3) *Follower's tweets*, (4) *All tweets*, and **Collaborative-based**: (5) *Followee's Ids*, (6) *Follower's Ids*, (7) *All Ids*.

### 2.1 System Interaction

A typical user interaction with the system can be seen in Figure 11. Users of Twittomender are first asked to sync their Twitter account with Twittomender so that we can retrieve their profile information and form a user document. Each user in the Twittomender database is modeled as a document, with the documents content containing up to 200 of the most recent tweets of that user. Using the TF-IDF scoring metric, which examines the frequency of a given term within a document multiplied by its frequency score across all the documents in that index, search queries entered are scored for similarity amongst the document

<sup>2</sup> <http://twittomender.ucd.ie>

<sup>3</sup> <http://apiwiki.twitter.com>

<sup>4</sup> <http://lucene.apache.org/java/docs/>

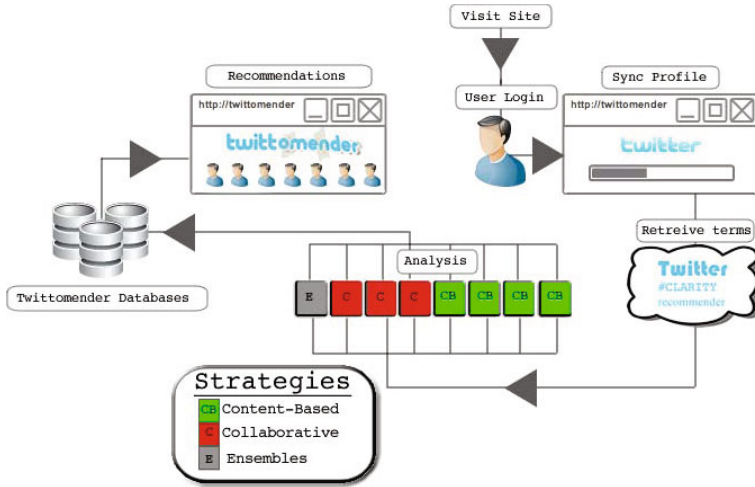


Fig. 1. The Twittomender System Control Flow Diagram

corpus to produce a result set. These search queries are either top terms from that user’s tweets (content-based) or their followees/followers ids (collaborative-based). For each strategy the top 20 of these document results are returned to the system as recommendations for followees. To form the final recommendations that are presented to a given user we then combine all our 7 recommendation strategies to form an ensemble approach that represents the most frequent suggestions across all strategies. The users recommended in the ensemble approach are then presented to the synced user with their profile pictures, user statistics and a term cloud of frequent terms from tweets to explain the recommendation and aid in selecting potential followees.

### 3 Evaluation

To evaluate Twittomender, we carried out two rounds of testing. The initial offline evaluation comprised of selecting 20,000 user documents from our database. This group of documents was split into two sets, a 1000 user test set and a 19,000 user recommendation pool to suggest users from, for each of the test users. Our precision metric for these offline evaluations was centered around whether or not a suggested user was already being followed by that test user, if so, it was then classed as a valid recommendation. There is a downside to this offline approach. How do we discern the validity of the other suggested users that are not being followed by the current test user. The solution to this is an online user study.

In the online study, we asked active users of Twitter to sync their Twitter profiles with Twittomender and each one was given a set of recommendations. The main aim behind this online trial was to find out whether the unknowns from a recommendation set were valid also. To this end we removed recommended

users that the synced user was already following to find out the number of unknowns they would follow from the recommendations.

Results from both the offline and online evaluations have proven successful. In the offline evaluations, each test user was presented with a recommendation list of 20 users. On average, across 1000 test users, 5 followees were identified for each user. This success rate is from our best performing strategy, user documents created from a users follower connections (collaborative-based). In the online trial the 34 participants indicated they would be willing to follow on average 6.9 new users from a recommendation list size of 30. These are users that they are not already following. Both these statistics bode well for the potential for Twittomender to recommend followees both old and new.

## 4 Conclusion

In this paper we have shown the Twittomender system as an effective way of finding new people to follow on Twitter. We have introduced 8 various recommendation strategies employed in Twittomender, namely 4 content, 3 collaborative and the 1 ensemble approach. Clearly the 20,000 users selected for our experiment only represent a snapshot of Twitters users, but we believe they form a diverse section of the Twitter community and future work will expand on these test sets. From recent user trials of the system there is a clear indication as to the efficiency of Twittomender to produce new connections for a user. As future work we aim to extend the live trial to more users and to compare against the recommendations of Twitter's own friend recommender system. Also we plan to examine the nature in which people follow others, as these strategies currently can not predict users who are followed due to some event or activity taking place in the real world.

## References

1. Chen, J., Geyer, W., Dugan, C., Muller, M., Guy, I.: Make new friends, but keep the old: recommending people on social networking sites. In: Proceedings of the 27th International Conference on Human Factors in Computing Systems, CHI 2009, pp. 201–210. ACM, New York (2009)
2. Guy, I., Ronen, I., Wilcox, E.: Do you know?: recommending people to invite into your social network. In: Proceedings of the 13th International Conference on Intelligent User Interfaces, IUI 2009, pp. 77–86. ACM, New York (2009)
3. Hannon, J., Bennett, M., Smyth, B.: Recommending twitter users to follow using content and collaborative filtering approaches. In: Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys 2010, pp. 199–206. ACM, New York (2010)
4. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Grouplens: applying collaborative filtering to usenet news. *Commun. ACM* 40, 77–87 (1997)
5. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) *Adaptive Web 2007*. LNCS, vol. 4321, pp. 325–341. Springer, Heidelberg (2007)

# Visual Exploration of Health Information for Children

Frans van der Sluis<sup>1</sup>, Sergio Duarte Torres<sup>2</sup>, Djoerd Hiemstra<sup>2</sup>,  
Betsy van Dijk<sup>1</sup>, and Frea Krusinga<sup>3</sup>

<sup>1</sup> Department of Human Media Interaction, University of Twente, P.O. Box 217,  
7500 AE Enschede, The Netherlands

<sup>2</sup> Database Group, University of Twente, P.O. Box 217, 7500 AE Enschede,  
The Netherlands

<sup>3</sup> Emma Children Hospital, University Medical Center Amsterdam, The Netherlands

**Abstract.** Children experience several difficulties retrieving information using current Information Retrieval (IR) systems. Particularly, children struggle to find the right keywords to construct queries given their lack of domain knowledge. This problem is even more critical in the case of the specialized health domain. In this work we present a novel method to address this problem using a cross-media search interface in which the textual data is searched through visual images. This solution aims to solve the recall and recognition problem which is salient for health information, by replacing the need for a vocabulary with the easy task of recognising the different body parts.

## 1 Introduction

Using health information is often problematic, being complex to find and difficult to read. For children, this becomes even more problematic. Often pediatric health information is not written at an appropriate readability level [1] and is difficult to retrieve for children because of a lack of domain knowledge [2], whilst the benefits of pediatric health information to children are extra salient: e.g., in particular for ill children, allowing to overcome uncertainty about their disease.

Van der Sluis and Van Dijk [2] identified four salient problems children have with IR systems, one of which is the vocabulary problem. Numerous studies show children have difficulties in choosing the right words. Often, they misspell their keywords or use keywords that are too broad or too narrow [3]. These findings are often attributed to a lack of vocabulary, which is a known problem in IR research; i.e., the vocabulary problem [4].

Visual search interfaces allow to alleviate all facets of the vocabulary problem for children through removing the free recall of words. Tag clouds are well-known examples of a visual search interface [5]. A tag cloud gives a representation of the word frequency for the most used words in the underlying corpus. However, in the case of health information for children, also the recognition of commonly used words is likely not to be helpful enough.

To overcome both the recognition and the recall problem, this demo will present a cross-media retrieval interface. This interface reduces the need for the recognition of words by using images to represent easy recognizable body parts. Moreover, to make more complex concepts available, users can zoom-in on certain body parts to make their search more specific. Hence, children can explore health related information through visually exploring the different body parts.

## 2 Design

The User Interface (UI) of the demo is shown in Figure 1. Two main functionalities are involved in this UI: visual search and results presentation, both will be further elaborated in the following paragraphs. Moreover, we will first describe the data set and the performed data enrichment allowing for a better mapping on the search metaphor.



Fig. 1. Illustration of the user interface

### 2.1 Data Set

The data utilized in this work consists of metadata describing the items available in the library of one of the largest children's hospitals in the Netherlands. These items represent a collection of diverse media types as books, dvds, coloring pages, etc. The purpose of the hospital's library is to provide trusted information for children about health, diseases, treatments and other physical and emotional issues that commonly arise in the hospital. Currently, the library consists of 560 items and each item is described with title, age appropriateness range, a short description, and an image.

To connect the interface with the search engine, we first determined the body parts and the level of granularity needed for the interface. For this purpose a dictionary of body parts was built based on the Wikipedia categories containing articles with body parts (e.g., *Anatomy*). The identification was carried out by simply matching the entries of the dictionary with the title and description of the items. Given that the recall of this approach is low since few items explicitly mention the body parts in the meta-data, we enriched the data by adding a list of body parts that are related to the content of the items.

**Table 1.** Top-5 frequencies per body part obtained by applying simple matching (a) and after enriching the data (b)

(a) Simple matching.		(b) Enriched data.	
Body part	Frequency	Body Part	Frequency
Brain	15	Brain	21
Body	16	Body	19
Head	6	Nose	12
Eyes	4	Lungs	9
Ears	3	Eyes	7

This process was performed automatically by first identifying the entities from the metadata using Wikipedia, in a similar fashion as in [6]. We establish a relation between an item and a body part if the Wikipedia article associated to one of its entities is connected in the Wikipedia link structure with the article associated to the body part. For example, if one of the entities mentioned in the item description is *deaf*, our system is able to relate the item with the body part *ear* since its Wikipedia article is referred from the article *Deaf*. Using this method allowed us to increase the coverage from 28% to 53%. Table 1 illustrates the most frequent body parts found in our dataset before and after applying the method described to enrich the data.

The search is performed by constructing a query based on the body part that is clicked. This query is sent to the search engine. We employed the Pf/Tijah engine [7] to index the title, description and the augmented body parts found. The communication between the UI and search engine is performed via the open search protocol [8] which also allows other parties to safely search our data.

## 2.2 Visual Search and Results Presentation

The search-part of the UI uses an illustration of a body as a search metaphor. As indicated in Section 1, such a metaphor is expected to reduce the vocabulary problem, which is particularly salient with pediatric health information. As Table 1 illustrates, selections can range from the whole body to the brain. This considerable difference in the level of detail creates the need for a zoom-in. We solved this need by using one image which contains a great level of detail, allowing for an image zoom-in to very specific parts (e.g., the ears). Moreover, the image changes when zooming in to highlight the relevant aspects at that level of details (e.g., the brains).

The interaction has been kept deliberately simple: a point and click paradigm is used for both selecting and zooming in, using feedback to make the functionality intuitive. Certain parts of the image are highlighted at any time, indicating a click on the highlighted part will give search results on that part. Moreover, when clicking on a highlighted area, the image will automatically zoom-in, leading to new highlighted areas.



### 3 Discussion and Future Work

This demo presents a novel way to solve a challenging retrieval problem: making often highly specialized health information searchable by less experienced users. Although the presented visual metaphor cannot cover all the information in the data set it is highly useful for exploring a particular domain of information.

The presented system can work for data written in any language, not relying on the use of words to search, and can work for different visual metaphors and thus different domains as well: any data set can be enriched with its association to a visualized concept using the Wikipedia-based method presented in Section 2.1. However, the reliance on a fixed visual metaphor makes the system fairly query-specific. The idea of basic level categories, indicating that people have common basic concepts for which clear representative images can be found [9], and the availability of resources such as open clip-art can alleviate this problem in the long run.

The presented design incorporated basic findings on interaction of children with IR systems, making it a truly user-centered design, aimed at some of the most salient problems in a particularly difficult domain. The presented demo is ongoing work and will be employed in a children's hospital where each patient has a touch screen to be used for entertainment and informative purposes.

### Acknowledgements

This work was part of the PuppyIR project, which is supported by a grant of the 7th Framework ICT Programme (FP7-ICT-2007-3) of the European Union.

### References

1. D'Alessandro, D.M., Kingsley, P., Johnson-West, J.: The Readability of Pediatric Patient Education Materials on the World Wide Web. *Arch. Pediatr. Adolesc. Med.* 155, 807–812 (2001)
2. Van der Sluis, F., Van Dijk, E.M.A.G.: A closer look at children's information retrieval usage: Towards child-centered relevance. In: *Proceedings of the Workshop on Accessible Search Systems held at the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York (2010)
3. Bilal, D.: Children's use of the yahooligans! web search engine: 1. cognitive, physical, and affective behaviors on fact-based search tasks. *Journal of the American Society for Information Science* 51, 646–665 (2000)
4. Furnas, G.W., Landauer, T.K., Gomez, L.M., Dumais, S.T.: The vocabulary problem in human-system communication. *Commun. ACM* 30, 964–971 (1987)
5. Hassan-Montero, Y., Víctor Herrero-solana, A.: Improving tag-clouds as visual information retrieval interfaces. In: *Merida, InSciT 2006 Conference* (2006)
6. He, J., de Rijke, M.: An exploration of learning to link with wikipedia: Features, methods and training collection. In: *Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009*. LNCS, vol. 6203, pp. 324–330. Springer, Heidelberg (2010)

7. Hiemstra, D., Rode, H., van Os, R., Flokstra, J.: Pftijah: text search in an xml database system. In: Proceedings of the 2nd International Workshop on Open Source Information Retrieval (OSIR), Seattle, WA, USA, Ecole Nationale Supérieure des Mines de Saint-Etienne, pp. 12–17 (2006)
8. Clinton, D.: Specification opensearch 1.1 draft 4 (2007)
9. Hoenkamp, E.: Why information retrieval needs cognitive science: A call to arms. In: Proceedings of the 27th Annual Conference of the Cognitive Science Society, vol. 2005, pp. 965–970 (2005)

# Author Index

- Addis, Andrea 32  
Agichtein, Eugene 67  
Aktolga, Elif 617  
Albakour, M-Dyaa 605  
Aldavert, David 314  
Alkahky, Ali 238  
Allan, James 617  
Alonso, Omar 153  
Altingovde, I. Sengor 510  
Andrade, Nazareno 189  
Arampatzis, Avi 117, 326, 759  
Arguello, Jaime 141  
Armano, Giuliano 32  
Azzopardi, Leif 716
- Badrinath, Rama 641  
Baeza-Yates, Ricardo 153  
Baillie, Mark 568  
Balog, Krisztian 580  
Baskaya, Feza 593  
Bellogín, Alejandro 301  
Bennett, Mike 448  
Bennett, Paul N. 226  
Berendt, Bettina 207  
Bortnikov, Edward 104  
Boves, Lou 491  
Bratsberg, Svein Erik 530  
Breslin, John G. 201  
Buffoni, David 743
- Cai, Peng 562  
Callan, Jamie 141  
Cambazoglu, B. Barla 510  
Campinas, Stéphane 555  
Carman, Mark 747  
Carrillo de Albornoz, Jorge 55  
Carterette, Ben 141, 543  
Carvalho, Vitor R. 466  
Castells, Pablo 301  
Chatzichristofis, Savvas A. 326, 759  
Chiluka, Nitin 189  
Choi, Yoonjung 7  
Clarke, Charles L.A. 460  
Collier, Rem W. 695  
Craswell, Nick 264, 497
- Crestani, Fabio 436, 747  
Croft, W. Bruce 387  
Cui, Yachao 19  
Cummins, Ronan 277
- Darling, William M. 629  
Darwish, Kareem 238  
Datta, Anwitaman 653  
Delbru, Renaud 555  
Demartini, Gianluca 264, 772  
Denoyer, Ludovic 411  
de Rijke, Maarten 251, 362  
De Roeck, Anne 605  
de Vries, Arjen P. 264  
Díaz, Alberto 55  
Diaz, Fernando 141  
Di Buccio, Emanuele 755  
Dinh, Duy 375  
Di Nunzio, Giorgio Maria 675  
Drosatos, George 117  
Duan, Huizhong 350  
Dulac-Arnold, Gabriel 411  
Dunnion, John 695  
Duric, Adnan 629
- Efraimidis, Pavlos S. 117  
Eibl, Maximilian 679  
Eickhoff, Carsten 776  
Elsweiler, David 568  
Elyan, Eyad 687
- Fasli, Maria 605  
Fetterly, Dennis 497  
Frommholz, Ingo 729, 751
- Gabrilovich, Evgeniy 4  
Gaizauskas, Robert 767  
Gallinari, Patrick 411, 743  
Gao, Wei 562  
Gatterbauer, Wolfgang 479  
Gerani, Shima 436, 747  
Gervás, Pablo 55  
Geva, Shlomo 460  
Glassey, Richard 776

- Gurevych, Iryna 712  
 Gyllstrom, Karl 80, 776
- Hagen, Matthias 503  
 Hanjalic, Alan 611, 699, 704  
 Hannon, John 784  
 Hauff, Claudia 691  
 He, Jing 338  
 He, Yulan 214, 283  
 Hefny, Ahmed 238  
 Herbert, Benjamin 712  
 Hiemstra, Djoerd 670, 788  
 Hofmann, Katja 251  
 Hou, Yuexian 721  
 Huang, Wei Chi 460
- Iofciu, Tereza 264  
 Iria, José 424  
 Itakura, Kelly Y. 460
- Jagarlamudi, Jagadeesh 226  
 Järvelin, Kalervo 1, 593  
 Jiang, Jing 338  
 Jin, Wei 19  
 Joachims, Thorsten 6  
 Jonassen, Simon 530  
 Jones, Gareth J.F. 683, 725  
 Jose, Joemon M. 738
- Kadar, Cristina 424  
 Kamps, Jaap 92  
 Karatzas, Dimosthenis 314  
 Kazai, Gabriella 165  
 Keikha, Mostafa 436, 747  
 Keskustalo, Heikki 593  
 Kim, Jinyoung 466  
 Kim, Kwang Hyun 387  
 Kim, Yunhyong 605  
 Kinsella, Sheila 201  
 Kofler, Christoph 611  
 Koolen, Marijn 92  
 Kraaij, Wessel 491  
 Kroon, Fred W. 629  
 Kruisinga, Frea 788  
 Kruschwitz, Udo 605  
 Kürsten, Jens 679
- Lalmas, Mounia 277, 729, 751  
 Larson, Martha 611, 699, 704  
 Lee, Joon Ho 387
- Lempel, Ronny 104  
 Leonard, David 695  
 Leveling, Johannes 675  
 Li, Chenliang 653  
 Li, Peng 19  
 Li, Xiaoming 338  
 Lillis, David 695  
 Lim, Ee-Peng 338  
 Lipka, Nedim 307  
 Liu, Qiaoling 67  
 Lladós, Josep 314  
 Llorens, Hector 767  
 Lopez, Patrice 725
- Macdonald, Craig 517  
 Magdy, Walid 683, 725  
 Magnani, Matteo 780  
 Mandl, Thomas 675  
 Massoudi, Kamran 362  
 Mayer, Rudolf 763  
 McCarthy, Kevin 448, 784  
 McDonald, Ryan 368  
 Melucci, Massimo 755  
 Moens, Marie-Francine 80  
 Montesi, Danilo 780  
 Moshfeghi, Yashar 738  
 Myaeng, Sung-Hyon 7
- Najork, Marc 497  
 Nanas, Nikolaos 605  
 Navarro, Borja 767  
 Naveed, Nasir 733  
 Neumayer, Robert 763  
 Nørvåg, Kjetil 763  
 Nunziante, Gabriele 780
- Oh, Heung-Seon 7  
 Okumura, Manabu 177  
 O'Riordan, Colm 277  
 Ounis, Iadh 517  
 Ozcan, Rifat 510
- Passant, Alexandre 201  
 Perego, Raffaele 665  
 Phelan, Owen 448  
 Piwowarski, Benjamin 751  
 Plaza, Laura 55  
 Polajnar, Tamara 776  
 Pouwelse, Johan 189
- Quénot, Georges 708

- Ārehūrek, Radim 289  
 Richardson, Matthew 399  
 Robertson, Stephen 129  
 Rossi, Luca 780  
 Rusiñol, Marçal 314  
 Ruthven, Ian 568  
  
 Safadi, Bahjat 708  
 Saquete, Estela 767  
 Schmitt, Ingo 729  
 Seo, Jangwon 387  
 Serdyukov, Pavel 399  
 Shi, Yue 699, 704  
 Silvestri, Fabrizio 665  
 Sizov, Sergej 733  
 Smirnova, Elena 580  
 Smith, David A. 617  
 Smyth, Barry 448, 784  
 Song, Dawei 605, 687, 721, 755  
 Song, Fei 629  
 Staab, Steffen 733  
 Stein, Benno 307, 503  
 Subašić, Ilija 207  
 Sun, Aixin 653  
 Szarvas, György 712  
  
 Täckström, Oscar 368  
 Takamura, Hiroya 177  
 Tamine, Lynda 375  
 Taylor, Mike 399  
 Thota, Sree Lekha 543  
 Tjin-Kam-Jet, Kien 670  
 Toledo, Ricardo 314  
 Tollari, Sabrina 743  
 Tonello, Nicola 665  
 Toolan, Fergus 695  
 Torres, Sergio Duarte 776, 788  
 Trieschnigg, Dolf 670, 691  
 Trotman, Andrew 460  
  
 Tsagkias, Manos 362  
 Tummarello, Giovanni 555  
  
 Ulusoy, Özgür 510  
  
 van der Sluis, Frans 788  
 van Dijk, Betsy 788  
 van Rijsbergen, Keith 716, 729, 751  
 Vargiu, Eloisa 32  
 Veni Madhavan, C.E. 641  
 Venkatasubramanian, Suresh 641  
 Verberne, Suzan 491  
 Vinay, Vishwa 399  
 Vornovitsky, Kolman 104  
  
 Wang, Bin 19  
 Wang, Jun 301  
 Wang, Lei 687  
 Wartena, Christian 43  
 Weerkamp, Wouter 362  
 Weng, Jianshu 338  
 White, Ryan W. 399  
 Whiteson, Shimon 251  
 Wibbels, Martin 43  
 Wong, Kam-Fai 562  
  
 Yan, Hongfei 338  
 Yokono, Hikaru 177  
  
 Zagoris, Konstantinos 326, 759  
 Zellhöfer, David 729  
 Zhai, Chengxiang 350  
 Zhang, Lusheng 695  
 Zhang, Peng 721  
 Zhao, Wayne Xin 338  
 Zhao, Xiaozhao 721  
 Zhou, Aoying 562  
 Zhou, Deyu 283  
 Zuccon, Guido 716