Djoerd Hiemstra · Marie-Francine Moens ·
Josiane Mothe · Raffaele Perego ·
Martin Potthast · Fabrizio Sebastiani (Eds.)

LNCS 12656

Advances in Information Retrieval

43rd European Conference on IR Research, ECIR 2021
Virtual Event, March 28 – April 1, 2021
Proceedings, Part I

Part I

Springer

Lecture Notes in Computer Science 12656

Djoerd Hiemstra · Marie-Francine Moens ·
Josiane Mothe · Raffaele Perego ·
Martin Potthast · Fabrizio Sebastiani (Eds.)

Advances in Information Retrieval

43rd European Conference on IR Research, ECIR 2021
Virtual Event, March 28 – April 1, 2021
Proceedings, Part I

Springer

Editors

Djoerd Hiemstra
Radboud University Nijmegen
Nijmegen, The Netherlands

Josiane Mothe
Toulouse Institute of Computer Science
Research
Toulouse, France

Martin Potthast
Leipzig University
Leipzig, Germany

Marie-Francine Moens
Department of Computer Science
Katholieke Universiteit Leuven
Heverlee, Belgium

Raffaele Perego
Istituto di Scienza e Tecnologie
dell'Informazione
Consiglio Nazionale delle Ricerche
Pisa, Italy

Fabrizio Sebastiani
Istituto di Scienza e Tecnologie
dell'Informazione
Consiglio Nazionale delle Ricerche
Pisa, Italy

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-030-72112-1 ISBN 978-3-030-72113-8 (eBook)
https://doi.org/10.1007/978-3-030-72113-8

LNCS Sublibrary: SL3 – Information Systems and Applications, incl. Internet/Web, and HCI

The use of general descriptive names, registered names, trademarks, service marks, 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.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

It is our great pleasure to welcome you to ECIR 2021, the 43rd edition of the annual BCS-IRSG European Conference on Information Retrieval.

ECIR 2021 was to be held in Lucca, Italy, but due to the COVID-19 pandemic emergence and the travel restrictions enforced worldwide, the conference was held entirely online. ECIR 2021 started on March 28 with a day of (full-day and half-day) tutorials, plus the Doctoral Consortium. The main conference took place in the three days that followed (March 28 – April 1). The technical program of the main conference included three exciting keynote talks, one per day: the first was presented by Francesca Rossi (IBM), the second by Ahmed Hassan Awadallah (Microsoft AI Research), as the winner of the BCS/Microsoft/BCS IRSG Karen Spärck Jones Award 2020, and the third by Ophir Frieder (Georgetown University). The technical program also consisted of research papers by contributors from Europe and the rest of the world. In total, 488 papers were submitted across all tracks, from 53 different countries. The program committees for the various tracks decided to accept 145 papers in total; the final scientific program thus included 50 full papers (a 24% acceptance rate), 39 short papers (25% acceptance rate), 15 demonstration papers (48% acceptance rate), and 11 reproducibility papers (52% acceptance rate). As in the previous edition, the technical program also included 12 "lab" (i.e., shared task) boosters from the CLEF 2021 conference, and the presentation of selected papers published in the 2020 issues of the Information Retrieval Journal. Symmetrically, the authors of a selection of ECIR 2021 papers will be invited to submit an extended version for publication in a special issue of the journal.

The last day of the conference (April 1) was devoted to 5 workshops and an exciting Industry Day. The workshops dealt with important topics such as algorithmic bias in search and recommendation (BIAS workshop), bibliometric-enhanced information retrieval (BIR workshop), conversational systems (MICROS workshop), online misinformation (ROMCIR workshop), and narrative extraction from texts (Text2Story workshop). This year the Industry Day was focused on the experience of Ph.D. interns in industrial contexts, and showcased success stories and positive experiences of former Ph.D. interns and former Ph.D. mentors. All submissions were peer reviewed by at least three international Program Committee members to ensure that only submissions of the highest quality were included in the final program. The acceptance decisions were further informed by discussions among the reviewers for each submitted paper, led by a senior Program Committee member or one of the track chairs. The accepted contributions covered the state of the art in IR: deep-learning–based information retrieval techniques, use of entities and knowledge graphs, recommender systems, retrieval methods, information extraction, question answering, topic and prediction models, multimedia retrieval, etc. In keeping with tradition, the ECIR 2021 program saw a high proportion of papers with students as first authors, and a balanced mix of papers from universities, public research institutes, and companies.

Putting everything together was hard teamwork. We want to thank everybody involved in making ECIR 2021 an exciting event. First and foremost, we want to thank our Program Chairs Djoerd Hiemstra and Marie-Francine (Sien) Moens for chairing the selection of the full papers. Many thanks also to the Short Papers Chairs Josiane Mothe and Martin Potthast, who managed not only the short paper submissions but also the CLEF papers submissions; to the Tutorials Chairs Richard McCreadie and Alejandro Moreo; to the Workshops Chairs Lorraine Goeuriot and Nicola Tonellotto; to the Reproducibility Track Chairs Maria Maistro and Gianmaria Silvello; to the Demo Chairs Nattiya Kanhabua and Franco Maria Nardini; to the Doctoral Consortium Chairs Claudio Lucchese and Guido Zuccon; to the Industry Day Chairs Roi Blanco and Fabrizio Silvestri; to the Sponsorship Chair Nicola Ferro; and to the Test-of-Time Award Chair Gabriella Pasi. Special thanks go also to our Publicity Chair Andrea Esuli and to our Proceedings Chair Ida Mele. All of them went to great lengths to ensure the high quality of this conference. Quite aside from the people who held chairing roles, lots of other people contributed to the scientific success of ECIR 2021: many thanks to the members of the Senior Program Committee, to the members of the Program Committees of the various tracks, to the mentors of the Doctoral Consortium Committee, and to all those who reviewed, in any capacity, full papers, short papers, reproducibility papers, tutorial and workshop proposals, and demo papers. Last but not least, we would like to thank all the members of the local organizing team at the National Research Council of Italy; in order to keep the registration fees as low as possible, no professional conference organization company was called in to help, which meant that this team took 100% of the organization upon them. We would thus like to thank our three Local Organization Chairs Cristina Muntean, Marinella Petrocchi and Beatrice Rapisarda. Thanks also to (in alphabetic order) Silvia Corbara, Andrea Esuli, Ida Mele, Alessio Molinari, Alejandro Moreo, Vinicius Monteiro de Lira, Franco Maria Nardini, Andrea Pedrotti, Nicola Tonellotto, Roberto Trani, and Salvatore Trani, for helping in various phases of the organization. They all invested tremendous efforts into making ECIR 2021 an exciting event by helping to create an enjoyable online and offline experience for authors and attendees. It is thanks to them that the organization of the conference was not just hard work, but also a pleasure. Finally, we would like to give heartfelt thanks to our sponsors and supporters: Bloomberg (platinum and best paper awards sponsor), Amazon, eBay, Google (gold sponsors), Textkernel (silver sponsor), Springer (test-of-time paper award sponsor), and Signal (industry impact award sponsor). We also gratefully acknowledge the generous support of the ACM Special Interest Group on Information Retrieval (ACM SIGIR) and of the ECIR 2020 organizers. We thank them all for their support and contributions to the conference, which allowed us to ask a low fee to paper authors only and to keep the registration free for all other attendees. Thanks also to the National Research Council of Italy, to the IMT School for Advanced Studies Lucca, to the British Computer Society's Information Retrieval Specialist Group (BCS-IRSG), and to the AI4Media project, for supporting our organizational work.

We hope you enjoy these proceedings of ECIR 2021!

March 28 to April 1, 2021

Raffaele Perego
Fabrizio Sebastiani

Organization

General Chairs

Raffaele Perego	ISTI-CNR, Italy
Fabrizio Sebastiani	ISTI-CNR, Italy

Program Chairs

Djoerd Hiemstra	Radboud University, The Netherlands
Marie-Francine (Sien) Moens	KU Leuven, Belgium

Short Papers Chairs

Josiane Mothe	Université de Toulouse, France
Martin Potthast	Leipzig University, Germany

Tutorials Chairs

Richard McCreadie	University of Glasgow, UK
Alejandro Moreo	ISTI-CNR, Italy

Workshops Chairs

Lorraine Goeuriot	Université Grenoble Alpes, France
Nicola Tonellotto	Università di Pisa, Italy

Reproducibility Track Chairs

Maria Maistro	University of Copenhagen, Denmark
Gianmaria Silvello	Università di Padova, Italy

Demo Chairs

Nattiya Kanhabua	Upwork, Thailand
Franco Maria Nardini	ISTI-CNR, Italy

Industry Day Chairs

Roi Blanco	Amazon Research, Spain
Fabrizio Silvestri	Facebook, UK

Doctoral Consortium Chairs

Claudio Lucchese	Università di Venezia, Italy
Guido Zuccon	University of Queensland, Australia

Sponsorships Chair

Nicola Ferro	Università di Padova, Italy

Test-of-Time Award Chair

Gabriella Pasi	Università di Milano-Bicocca, Italy

Publicity Chair

Andrea Esuli	ISTI-CNR, Italy

Proceedings Chair

Ida Mele	IASI-CNR, Italy

Webmaster and Social Media Manager

Beatrice Rapisarda	IIT-CNR, Italy

Local Organization Chairs

Cristina Muntean	ISTI-CNR, Italy
Marinella Petrocchi	IIT-CNR, Italy
Beatrice Rapisarda	IIT-CNR, Italy

Local Organization Committee

Silvia Corbara	ISTI-CNR, Italy
Alessio Molinari	ISTI-CNR, Italy
Vinicius Monteiro de Lira	ISTI-CNR, Italy
Roberto Trani	ISTI-CNR, Italy
Salvatore Trani	ISTI-CNR, Italy
Andrea Pedrotti	ISTI-CNR, Italy

Organizing Institutions

Consiglio Nazionale
delle Ricerche

ISTITUTO DI SCIENZA E TECNOLOGIE
DELL'INFORMAZIONE "A. FAEDO"

Istituto di
Informatica e Telematica

IMT SCHOOL
FOR ADVANCED
STUDIES
LUCCA

Program Committee

Ahmed Abdelali	Hamid Bin Khalifa University
Karam Abdulahhad	GESIS - Leibniz Institute for the Social Sciences
Dirk Ahlers	Norwegian University of Science and Technology
Qingyao Ai	University of Utah
Ahmet Aker	University of Duisburg-Essen
Navot Akiva	Bar-Ilan University
Mehwish Alam	FIZ Karlsruhe - Leibniz Institute for Information Infrastructure, AIFB Institute, KIT
Dyaa Albakour	Signal AI
Mohammad Aliannejadi	University of Amsterdam
Pegah Alizadeh	École Supérieure d'Ingénieurs Léonard da Vinci
Satya Almasian	Heidelberg University
Omar Alonso	Instacart
İsmail Sengör Altıngövde	Bilkent University
Giambattista Amati	Fondazione Ugo Bordoni
Giuseppe Amato	ISTI-CNR
Linda Andersson	Artificial Researcher IT GmbH, TU Wien
Hassina Aouidad Aliane	CERIST
Ioannis Arapakis	Telefonica Research
Jaime Arguello	The University of North Carolina at Chapel Hill
Mozhdeh Ariannezhad	University of Amsterdam
Maurizio Atzori	University of Cagliari
Ebrahim Bagheri	Ryerson University
Seyed Ali Bahreinian	IDSIA
Krisztian Balog	University of Stavanger
Alexandros Bampoulidis	Research Studio Data Science - RSA FG
Mitra Baratchi	Leiden University
Alvaro Barreiro	University of A Coruña
Alberto Barrón-Cedeño	University of Bologna
Alejandro Bellogin	Universidad Autònoma de Madrid
Patrice Bellot	Aix-Marseille Université - CNRS (LSIS)
Alessandro Benedetti	Sease
Klaus Berberich	Saarbrücken University of Applied Sciences (htw saar)
Catherine Berrut	LIG, Université Joseph Fourier Grenoble I
Sumit Bhatia	IBM
Paheli Bhattacharya	Indian Institute of Technology Kharagpur
Roi Blanco	Amazon
Gloria Bordogna	National Research Council of Italy - CNR
Larbi Boubchir	University of Paris 8
Pavel Braslavski	Ural Federal University
David Brazier	Edinburgh Napier University
Timo Breuer	TH Köln (University of Applied Science)
Paul Buitelaar	Insight Centre for Data Analytics, National University of Ireland Galway

Fidel Cacheda	Universidade da Coruña
Sylvie Calabretto	LIRIS
Pável Calado	INESC-ID, University of Lisbon
Rodrigo Calumby	University of Feira de Santana
Ricardo Campos	Ci2 - Polytechnic Institute of Tomar; INESC TEC
Fazli Can	Bilkent University
Iván Cantador	Universidad Autónoma de Madrid
Annalina Caputo	Dublin City University
Zeljko Carevic	GESIS Leibniz Institute for the Social Sciences
Ben Carterette	Spotify
Pablo Castells	Universidad Autónoma de Madrid
Shubham Chatterjee	University of New Hampshire
Despoina Chatzakou	Information Technologies Institute, Centre for Research and Technology Hellas
Long Chen	University of Glasgow
Max Chevalier	IRIT
Adrian-Gabriel Chifu	Aix Marseille Univ, CNRS, LIS
Konstantina Christakopoulou	Google
Malcolm Clark	The University of the Highlands & Islands
Vincent Claveau	IRISA - CNRS
Jérémie Clos	University of Nottingham
Paul Clough	The University of Sheffield
Alessio Conte	University of Pisa
Fabio Crestani	University of Lugano (USI)
Bruce Croft	University of Massachusetts Amherst
Arthur Câmara	Delft University of Technology
Tirthankar Dasgupta	Tata Consultancy Services
Martine De Cock	University of Washington
Hélène De Ribaupierre	Cardiff University
Arjen de Vries	Radboud University
Yashar Deldjoo	Polytechnic University of Bari
Elena Demidova	Bonn University
José Devezas	University of Porto
Emanuele Di Buccio	University of Padua
Giorgio Maria Di Nunzio	University of Padua
Gaël Dias	University of Caen Normandie
Liviu Dinu	University of Bucharest
Vlastislav Dohnal	Masaryk University
Inês Domingues	IPO Porto + Universidade de Coimbra
Dennis Dosso	University of Padua
Pan Du	University of Montreal
Mehdi Elahi	University of Bergen
Tamer Elsayed	Qatar University
Ludwig Englbrecht	University of Regensburg
Liana Ermakova	HCTI EA-4249, Université de Bretagne Occidentale

José Alberto Esquivel	Primer.ai
Andrea Esuli	Istituto di Scienza e Tecnologie dell'Informazione
Ralph Ewerth	L3S Research Center, Leibniz Universität Hannover
Alessandro Fabris	University of Padova
Erik Faessler	University of Jena
Anjie Fang	Amazon.com
Hui Fang	University of Delaware
Hossein Fani	University of Windsor
Nicola Ferro	University of Padova
Sébastien Fournier	LSIS
Christoph M. Friedrich	University of Applied Sciences and Arts Dortmund
Ingo Frommholz	University of Wolverhampton
Norbert Fuhr	University of Duisburg-Essen
Michael Färber	Karlsruhe Institute of Technology
Luke Gallagher	RMIT University
Debasis Ganguly	IBM Ireland Research Lab
Darío Garigliotti	Aalborg University
Anastasia Giachanou	Utrecht University
Giorgos Giannopoulos	IMSI Institute, "Athena" Research Center
Alessandro Giuliani	University of Cagliari
Lorraine Goeuriot	Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG
Marcos Gonçalves	Federal University of Minas Gerais
Julio Gonzalo	UNED
Kripabandhu Ghosh	IISER Kolkata
Michael Granitzer	University of Passau
Adrien Guille	Université de Lyon
Rajeev Gupta	Microsoft
Shashank Gupta	Flipkart
Cathal Gurrin	Dublin City University
Matthias Hagen	Martin-Luther-Universität Halle-Wittenberg
Lei Han	The University of Queensland
Allan Hanbury	Vienna University of Technology
Preben Hansen	Stockholm University
Donna Harman	NIST
Helia Hashemi	University of Massachusetts Amherst
Faegheh Hasibi	Radboud University
Claudia Hauff	Delft University of Technology
Jer Hayes	Accenture
Ben He	University of Chinese Academy of Sciences
Nathalie Hernandez	IRIT
Djoerd Hiemstra	Radboud University
Daniel Hienert	GESIS - Leibniz Institute for the Social Sciences
Gilles Hubert	IRIT
Ali Hürriyetoğlu	Koç University
Adrian Iftene	"Al.I.Cuza" University of Iasi

Dmitry Ignatov	National Research University Higher School of Economics
Bogdan Ionescu	University Politehnica of Bucharest
Radu Tudor Ionescu	University of Bucharest
Mihai Ivanovici	Transilvania University of Brașov
Adam Jatowt	University of Innsbruck
Jean-Michel Renders	Naver Labs Europe
Shiyu Ji	UCSB
Jiepu Jiang	University of Wisconsin-Madison
Gareth Jones	Dublin City University
Joemon Jose	University of Glasgow
Chris Kamphuis	Radboud University
Jaap Kamps	University of Amsterdam
Nattiya Kanhabua	Upwork
Jussi Karlgren	Spotify
Jaana Kekäläinen	Tampere University
Liadh Kelly	Maynooth University
Roman Kern	Graz University of Technology
Daniel Kershaw	Elsevier
Prasanna Lakshmi Kompalli	Gokaraju Rangaraju Institute of Engineering and Technology
Ralf Krestel	Hasso Plattner Institute, University of Potsdam
Kriste Krstovski	University of Massachusetts Amherst
Udo Kruschwitz	University of Regensburg
Vaibhav Kumar	Amazon Alexa AI, Carnegie Mellon University
Oren Kurland	Technion, Israel Institute of Technology
Saar Kuzi	University of Illinois at Urbana-Champaign
Léa Laporte	INSA Lyon - LIRIS
Teerapong Leelanupab	King Mongkut's Institute of Technology Ladkrabang
Jochen L. Leidner	University of Sheffield
Mark Levene	Birkbeck, University of London
Elisabeth Lex	Graz University of Technology
Jimmy Lin	University of Waterloo
Matteo Lissandrini	Aalborg University
Suzanne Little	Dublin City University
Haiming Liu	University of Bedfordshire
Fernando Loizides	Cardiff University
David Losada	University of Santiago de Compostela
Natalia Loukachevitch	Research Computing Center of Moscow State University
Claudio Lucchese	Ca' Foscari University of Venice
Bernd Ludwig	Universität Regensburg
Sean MacAvaney	University of Glasgow
Craig Macdonald	University of Glasgow
Andrew Macfarlane	City, University of London
Joel Mackenzie	The University of Melbourne

João Magalhães	Universidade NOVA de Lisboa
Walid Magdy	The University of Edinburgh
Marco Maggini	University of Siena
Shikha Maheshwari	Chitkara University
Maria Maistro	University of Copenhagen
Antonio Mallia	New York University
Thomas Mandl	University of Hildesheim
Behrooz Mansouri	University of Tehran
Jiaxin Mao	Renmin University of China
Stefano Marchesin	University of Padova
Rainer Martin	Institute of Communication Acoustics, Ruhr-Universität Bochum
Miguel Martinez	Signal AI
Bruno Martins	IST and INESC-ID - Instituto Superior Técnico, University of Lisbon
Fernando Martínez-Santiago	Universidad de Jaén
Yosi Mass	IBM Haifa Research Lab
Sérgio Matos	IEETA, Universidade de Aveiro
Philipp Mayr	GESIS
Richard McCreadie	University of Glasgow
Graham McDonald	University of Glasgow
Parth Mehta	IRSI
Edgar Meij	Bloomberg L.P.
Ida Mele	IASI-CNR
Massimo Melucci	University of Padova
Marcelo Mendoza	Universidad Técnica Federico Santa María
Zaiqiao Meng	University of Cambridge
Dmitrijs Milajevs	Queen Mary University of London
Malik Muhammad Saad Missen	The Islamia University of Bahawalpur
Bhaskar Mitra	Microsoft
Marie-Francine Sien Moens	Katholieke Universiteit Leuven
Mohand Boughanem	IRIT University Paul Sabatier Toulouse
Ludovic Moncla	LIRIS (UMR 5205 CNRS), INSA Lyon
Vinicius Monteiro de Lira	CNR - Pisa
Felipe Moraes	Delft University of Technology
José Moreno	IRIT/UPS
Alejandro Moreo	Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo"
Yashar Moshfeghi	University of Strathclyde
Josiane Mothe	Université de Toulouse
Philippe Mulhem	LIG-CNRS
Cristina Ioana Muntean	ISTI CNR
Henning Müller	HES-SO
Preslav Nakov	Qatar Computing Research Institute, HBKU
Franco Maria Nardini	ISTI-CNR

Wolfgang Nejdl	L3S and University of Hannover
Jian-Yun Nie	University of Montreal
Andreas Nürnberger	Otto-von-Guericke University of Magdeburg
Kjetil Nørvåg	Norwegian University of Science and Technology
Neil O'Hare	Yahoo Research
Douglas Oard	University of Maryland
Michel Oleynik	Medical University of Graz
Anaïs Ollagnier	University of Exeter
Teresa Onorati	Universidad Carlos III de Madrid
Salvatore Orlando	Università Ca' Foscari Venezia
Iadh Ounis	University of Glasgow
Mourad Oussalah	University of Oulu
Deepak P.	Queen's University Belfast
Jiaul Paik	IIT Kharagpur
João Palotti	MIT
Girish Palshikar	Tata Consultancy Services
Polina Panicheva	National Research University Higher School of Economics, St Petersburg
Panagiotis Papadakos	Information Systems Laboratory - FORTH-ICS
Javier Parapar	University of A Coruña
Dae Hoon Park	Yahoo Research
Arian Pasquali	University of Porto
Bidyut Kr. Patra	NIT Rourkela
Pavel Pecina	Charles University in Prague
Filipa Peleja	Levi Strauss & Co.
Gustavo Penha	Delft University of Technology
Raffaele Perego	ISTI-CNR
Giulio Ermanno Pibiri	ISTI-CNR
Jeremy Pickens	OpenText
Karen Pinel-Sauvagnat	IRIT
Benjamin Piwowarski	CNRS/Sorbonne University Pierre and Marie Curie Campus
Martin Potthast	Leipzig University
Animesh Prasad	Amazon Alexa
Chen Qu	University of Massachusetts Amherst
Navid Rekab-Saz	Johannes Kepler University (JKU)
Kaspar Riesen	University of Applied Sciences and Arts Northwestern Switzerland
Kirk Roberts	The University of Texas Health Science Center at Houston
Paolo Rosso	Universitat Politècnica de València
Eric Sanjuan	Laboratoire Informatique d'Avignon- Université d'Avignon
Kamal Sarkar	Jadavpur University, Kolkata
Ramit Sawhney	Tower Research Capital
Philipp Schaer	TH Köln (University of Applied Sciences)

Ralf Schenkel	Trier University
Fabrizio Sebastiani	ISTI-CNR
Florence Sedes	I.R.I.T. Univ. P. Sabatier
Thomas Seidl	Ludwig-Maximilians-Universität München (LMU Munich)
Giovanni Semeraro	University of Bari
Procheta Sen	Dublin City University
Gautam Kishore Shahi	University of Duisburg-Essen, Germany
Mahsa S. Shahshahani	University of Amsterdam
Azadeh Shakery	University of Tehran
Eilon Sheetrit	Technion - Israel Institute of Technology
Jialie Shen	Queen's University Belfast
Kai Shu	Arizona State University
Mário J. Silva	Universidade de Lisboa
Gianmaria Silvello	University of Padua
Fabrizio Silvestri	Facebook
Laure Soulier	Sorbonne Université-LIP6
Marc Spaniol	Université de Caen Normandie
Günther Specht	University of Innsbruck
Damiano Spina	RMIT University
Andreas Spitz	Ecole Polytechnique Fédérale de Lausanne
Efstathios Stamatatos	University of the Aegean
Hanna Suominen	The ANU
Lynda Tamine	IRIT
Carla Teixeira Lopes	University of Porto
Gabriele Tolomei	Sapienza University of Rome
Antonela Tommasel	ISISTAN Research Institute, CONICET-UNCPBA
Nicola Tonellotto	University of Pisa
Salvatore Trani	ISTI-CNR
Alina Trifan	University of Aveiro
Manos Tsagkias	Apple
Theodora Tsikrika	Information Technologies Institute, CERTH
Ferhan Ture	Comcast Labs
Yannis Tzitzikas	University of Crete and FORTH-ICS
Md Zia Ullah	CNRS
Julián Urbano	Delft University of Technology
Daniel Valcarce	Google
Julien Velcin	ERIC Lyon 2, EA 3083, Université de Lyon
Suzan Verberne	Leiden University
Manisha Verma	VerizonMedia
Karin Verspoor	The University of Melbourne
Vishwa Vinay	Adobe Research
Marco Viviani	Università degli Studi di Milano-Bicocca
Duc Thuan Vo	Ryerson University
Stefanos Vrochidis	Information Technologies Institute
Shuohang Wang	Singapore Management University

Xi Wang	University of Glasgow
Christa Womser-Hacker	University of Hildesheim
Grace Hui Yang	Georgetown University
Min Yang	The Chinese Academy of Sciences
Andrew Yates	Max Planck Institute for Informatics
Emine Yilmaz	University College London
Hai-Tao Yu	University of Tsukuba
Ran Yu	GESIS - Leibniz Institute for the Social Sciences
Reza Zafarani	Syracuse University
Eva Zangerle	University of Innsbruck
Fattane Zarrinkalam	Ryerson University
Sergej Zerr	Leibniz Universität Hannover
Weinan Zhang	Shanghai Jiao Tong University
Xiangyu Zhao	Michigan State University
Xinyi Zhou	Syracuse University
Xiaofei Zhu	Chongqing University of Technology
Guido Zuccon	The University of Queensland

Additional Reviewers

Amigó, Enrique
Anand, Mayuresh
Apte, Manoj
Auersperger, Michal
Bakhshi, Sepehr
Bannihatti Kumar, Vinayshekhar
Bartscherer, Frederic
Basile, Pierpaolo
Bedathur, Srikanta
Bondarenko, Alexander
Boughanem, Mohand
Breuer, Timo
Busch, Julian
Christophe, Clément
Cresci, Stefano
Dadwal, Rajjat
Dalal, Dhairya
de Freitas, João
De Ribaupierre, Hélène
Dessì, Danilo
Dsouza, Alishiba
Efimov, Pavel
Essam, Marwa
Feng, Haoyun
Fournier, Sebastien
Fröbe, Maik
Gabler, Philipp
Gerritse, Emma
Ghahramanian, Pouya
Gourru, Antoine
Haak, Fabian
Hakimov, Sherzod
Haouari, Fatima
Hasanain, Maram
Hingmire, Swapnil
Hoppe, Anett
Iovine, Andrea
Jatowt, Adam
Julka, Sahib
Jullien, Sami
Kanungsukkasem, Nont
Kondapally, Ranganath
Kosmatopoulos, Andreas
Lal, Yash Kumar
Lee, Kai-Zhan
Loizides, Fernando
Lucchese, Claudio
Mavropoulos, Thanassis
Mayerl, Maximilian
Moumtzidou, Anastasia

Muntean, Cristina Ioana
Murauer, Benjamin
Mussard, Stéphane
Musto, Cataldo
Nardini, Franco Maria
Nikas, Christos
Noullet, Kristian
Nurbakova, Diana
Otto, Christian
Parveen, Daraksha
Pasricha, Nivranshu
Patil, Sangameshwar
Pawar, Sachin
Pegia, Maria Eirini
Perego, Raffaele
Pibiri, Giulio Ermanno
Polignano, Marco
Poux-Médard, Gaël
Pérez Vila, Miguel Anxo
Qiao, Yifan
Rahmani, Hossein A.
Repke, Tim
Roy, Nirmal
Saleh, Shadi
Santana, Brenda
Schaer, Philipp
Semedo, David
Sen, Bipasha
Shah, Shalin
Sharma, Himanshu
Skopek, Ondrej
Strauß, Niklas
Su, Ting
Suryawanshi, Shardul
Suwaileh, Reem
Syamala, Rama
Tavares, Diogo
Tempelmeier, Nicolas
Tonellotto, Nicola
Trani, Roberto
Truchan, Hubert
Venturini, Rossano
Vötter, Michael
Wang, Benyou
Witschel, Frieder
Yang, Min
Yang, Yingrui
Zerhoudi, Saber
Zhang, Zixun
Zühlke, Monty-Maximilian

Platinum and Best Paper Awards Sponsor

Bloomberg
Engineering

Bloomberg is building the world's most trusted information network for financial professionals. Our 6,000+ engineers, developers, and data scientists are dedicated to advancing and building new solutions and systems for the Bloomberg Terminal and other products in order to solve complex, real-world problems. Improving search and discovery of relevant content, functionality, and insights are critical focus areas for Bloomberg. To this end, we use Machine Learning, Deep Learning, Natural Language Processing, Information Retrieval, and Knowledge Graph technology across Bloomberg in several applications, including search, question answering, data integration, recommender systems, etc. to quickly understand and respond to major world events in order to predict when or how breaking business news will move markets – and why.

Gold Sponsors

Google amazon | science ebay

Silver Sponsor

textkernel

Test-of-Time Best Paper Award Sponsor

Springer

Test-of-Time Best Paper Award Sponsor

SIGNAL

With Generous Support from

SIGIR
Special Interest Group
on Information Retrieval

Contents – Part I

Full Papers

Contents – Part II

Reproducibility Track Papers

Short Papers

Demo Papers

CLEF 2021 Lab Descriptions

Doctoral Consortium Papers

Workshops

Tutorials

Full Papers

Stay on Topic, Please: Aligning User Comments to the Content of a News Article

Jumanah Alshehri(✉), Marija Stanojevic, Eduard Dragut, and Zoran Obradovic

Center for Data Analytics and Biomedical Informatics, Temple University, Philadelphia, PA, USA
{jumanah.alshehri,marija.stanojevic,edragut, zoran.obradovic}@temple.edu

Abstract. Social scientists have shown that up to 50% of the comments posted to a news article have no relation to its journalistic content. In this study we propose a classification algorithm to categorize user comments posted to a news article based on their alignment to its content. The alignment seeks to match user comments to an article based on similarity of content, entities in discussion, and topics. We propose a BERTAC, BERT-based approach that learns jointly article-comment embeddings and infers the relevance class of comments. We introduce an ordinal classification loss that penalizes the difference between the predicted and true labels. We conduct a thorough study to show influence of the proposed loss on the learning process. The results on five representative news outlets show that our approach can learn the comment class with up to 36% average accuracy improvement comparing to the baselines, and up to 25% comparing to the BA-BC. BA-BC is our approach that consists of two models aimed to capture dis-jointly the formal language of news articles and the informal language of comments. We also conduct a user study to evaluate human labeling performance to understand the difficulty of the classification task. The user agreement on comment-article alignment is "moderate" per Krippendorff's alpha score, which suggests that the classification task is difficult.

Keywords: Text mining · Text classification · Online news · News comments · Relevancy · Understanding user-generated text

1 Introduction

The study of user comments is essential for social scientists, policymakers, and journalists since virtual discussions offer an insight into the public opinion.

J. Alshehri and M. Stanojevic—contributed equally.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 3–17, 2021.
https://doi.org/10.1007/978-3-030-72113-8_1

In 2020, people shifted more toward online discussions due to COVID-19. Many survey-based studies tried to understand the users' behavior by characterizing and categorizing comments in online news [23,28,33,38]. A salient outcome of these studies is that 20% to 50% of users' comments are irrelevant to the content or topic of those articles since users drift from the original topic to irrelevant sub-discussions [14,30]. Our goal in this work is to understand commenting behavior, more precisely, to automatically identify the subset of comments, from the set of comments an article receives, that are pertinent to the content of the article. The challenge is multi-fold: e.g., comments tend to be terse, colloquial, often non-literary, containing grammatical errors, misspellings, and punctuation misuse. Our premise is that users are inclined to write comments that diverge from the article topic to different extents, especially in lengthier discussions. This noise in the data affects downstream applications such as opinion mining.

Previous studies tried to remove the noise among comments by studying toxic comments [10,16], topic drifting [12,27], and understanding the quality of online news comments [7,11,25]. From NLP perspective, this problem is a supervised classification task to separate relevant from irrelevant comments.

In this paper, we introduce the Article-Comment Alignment Problem (ACAP). We aim to define a set of article-comment relevance classes and propose a methodology to classify article-comments pairs automatically. ACAP is a challenging task, for example, consider the article *"This is going to happen in the United States: Donald Trump calls for surveillance of Muslims and advocates waterboarding terror suspects after Brussels attack"*[1] from *Daily Mail* and the comment *"It's not Europe anymore. It's Eurabia."* Two human annotators rate the comment as *Irrelevant*, while the third annotator rates it as *Same Category.* The third annotator's label is the most appropriate, but choosing that category requires background knowledge on the political circumstances in Europe in 2016. In solving ACAP, we hypothesize the following: 1) It is possible to capture the extent of a connection and semantics between an article and its comments using globally pre-trained models, fine-tuned with local data. 2) Considering the natural order of labels during training will boost the algorithm learning process.

We test our hypotheses in the following practical scenarios: (1) limiting amount of labeled article-comment pairs (1K per dataset), (2) bounding the number of tokens from each document (article or comment), and (3) concomitantly working with formal text, in the form of news articles, and informal text, in the form of comments. The pairs are extracted from five online news outlets [20]: *Wall Street Journal* (WSJ), *Fox News* (FN), *Daily Mail* (DM), *The Guardian* (TG), and *Market Watch* (MW). This work makes the following contributions: 1) We introduce the Article-Comment Alignment Problem (ACAP) and analyze the hardness of ACAP using an agreement study on the classification of human annotators. 2) We propose BERTAC, which jointly learns embedding representations for articles and their comments, to solve ACAP. We also propose BA-BC, which consists of two models on trained on articles and the other on comments, which attempts to capture the difference in language style between them,

[1] Full article: https://dailym.ai/2Qz7RG9.

formal versus informal. We compare it to several approaches, including BA-BC, and show its superior performance. 3) We develop a novel ordinal classification loss for BERTAC that penalizes the difference between the predicted and true labels. The proposed loss exhibits similar performance to the original loss in terms of accuracy, however, it boosts the model performance when trained on high agreement examples. 4) We conduct extensive empirical studies on articles and comments from 5 representative news outlets.

2 Related Work

User comments are a powerful means to understand public opinion and reaction to emerging events. Many organizations invest in mining user comments to improve their decision making. News outlets and social platforms are recommending most relevant user posts to keep the attention of busy readers [33]. Many studies focus on mining the user opinion from social media [2,3,32] and online news comments [1,15,35,37]. Other works look into bias in the news, and its influence on user-generated content [31,36]. The main challenge in those studies is the unpredictable quality of user-generated content.

To solve this problem, a line of research focuses on comment drifting [12,27] by utilizing the temporal nature of comments. The older an article is, the more commentators it has, and the probability of exposure to topic drift is higher [25]. This phenomenon influences the quality of comments and their relevance.

Another line of work [5,17,29] investigates which part of an article a comment aligns with using statistical models, while other [11] use hand-crafted structural, lexical, syntactic, discourse, and relevance features as an input to the logistic regression. Even though their F1 score is in the range of 70–80% and their analysis measures correlation of the attributes with the label, hand-crafting features for each problem is difficult and time-consuming.

The work most related to ours attempts to automatically classify paragraph-comment agreement [25]. They labeled the data based on Likert scale categories [19], which is criticized for introducing bias. For instance, a number of works show that user responses are significantly affected by the order and direction of the rating scale [9,34].

Instead, we propose to use transformer pre-trained language approach [6,26]. We also create a new ordinal classification loss. As shown in the experimental study, our approach performs significantly better than the baselines on ACAP. We work with three annotators. Their labels give us support data to study the difficulty of the problem. We show that the annotators exhibit only fair agreement, indicating that ACAP is a difficult problem even for human beings.

3 Datasets

We collected news articles and their comments between 2015 and 2017 [13] from multiple news outlets. The dataset has over 19K articles with 9M comments. For this study, we chose five news outlets that are representative of the problem at hand. The dataset contains articles and comments with a broad range of lengths and different number of comments. This data allows us to test the behavior of the proposed models under varied settings. Table 1(A) shows the statistics of datasets.

Table 1. (A) Statistics by outlet. We randomly selected 1K article-comments pairs from each outlet and labeled them. ALA is the articles' average length and ALC is average comments' length, measured by number of words. (B) Classes proportions for each dataset. Outlets are sorted based on total number of available articles per outlet.

Outlet	(A) Dataset statistics				(B) Classes proportion			
	#Art.	#Comm.	ALA	ALC	Relevant	Same Ent.	Same Cat.	Irrelevant
FN	0.3K	72K	250	22	3%	21%	29%	47%
TG	1.6K	428K	797	54	5%	39%	32%	24%
MW	1.7K	65K	512	42	7%	51%	20%	22%
WSJ	3.6K	309K	164	57	8%	25%	34%	33%
DM	10K	1,012K	487	28	15%	17%	20%	48%

3.1 Labeling

We discard all articles without comments. We randomly select 1K article-comment pairs from each outlet. Then, annotators manually and independently label the pairs in four classes: Relevant, Same Entities, Same Category, and Irrelevant. Relevant class - the content of the comment discusses the same matter as the article. Same Entities class - the comment is not directly relevant, however, it mentions the same main entities within the same scope (category) of the article. For example, the article talks about a Real Madrid - F.C. Barcelona game, mentioning Ronaldo's performance in the game, and the comment talks about Ronaldo's best goal in the Portuguese team. Same Category class - comment in this class is not discussing the article, but it falls into the same category as the article. For example, both comment and article are discussing politics. Irrelevant class - a comment is in this class if it does not belong to any other class.

Figure 1 shows labeled examples of each class from WSJ. First column is part of the article; second column has four comment examples, each example represent a different class. The article in the table discusses Hillary Clinton's email story that came out before the 2016 U.S. election. The first comment is *Relevant*, the second comment does not discuss the main issue, however, it mentions some of the entities discussed in the article within the category of the article (politics). Hence, its class is *Same Entities*. The third comment does not refer to any named entity from the article, but it discussed another political issue. Thus, its class is *Same Category*. In the last comment, the user believes that he looks like *Joe Friday*. This has no connection with the article, therefore, the comment is deemed *Irrelevant*.

To obtain labeled instances, we asked three native English speakers, who were not involved in this work, to annotate the article-comments pairs.

First Part of the article	Comment	Class
The *Clinton Campaign* at *Obama Justice* Emails on *WikiLeaks* show a top federal lawyer giving *Hillary* a quiet heads up. *President Obama* and *Attorney General Loretta Lynchat* the *White House* in July. The most obnoxious pin of the 2016 campaign cames this week, as *Democrats*, their media allies and even *President Obama* accused the *FBI* of stacking the election. It's an extraordinary claim, coming as it does from the same that has -we now know – been stacking the crew election all along in the corridors of the *Justice Department*. This is the true November surprise. For four months, *FBI Director James Comey* has been the public face of the investigation into *Hillary Clinton*'s email server. He played that role so well, putting the *FBI* so front and center, that the country forgot about *Mr.Comey*'s bosses.	As a practicing lawyer this is just embarrassing. Lawyers are governed by a *Code of Professional Conduct*. The *Justice Department* has enormous power in our country. Politics is not supposed to be part of the equation. It is now abundantly clear that politics is now game on for the *Obama Justice Department*.	Relevant
	The news article that never was written is how *Obama* has corrupted the government agencies and how *Clinton* will continue the process? *Kim* has done the best job of placing the blame for the political corruption of these agencies? The *IRS State Dept. Justice Dept.* and compass because he has perpetuated this for political gain	Same Entity
	We allowed *binladen* family to fly out during 911 blackout as soon I read that in the news I swore never to vote *Bush* again.	Same Category
	I always look liked *Joe Friday*	Not Relevant

Fig. 1. Labeling example, entities are colored. The article category is politics.

We provide them with the following: 1) an article-comment pair without the surrounding context (i.e., the parent and child comments), and 2) the four label categories with an explanation and an example for each of them. We assign the final label using an averaging aggregation scheme. We map Irrelevant, Same Category, Same Entity, and Relevant to $0, 1, 2$, and 3, respectively. We average the (users) scores per pair and round to the nearest integer, which becomes the label of the pair. For example, a pair x-y receives the score $1, 1$, and 2 will have a label 1, which corresponds to "Same Category". Table 1(B) shows the proportion of each class per outlet. We also binarize labels, by assigning 0 to Irrelevant comments, and 1 to rest of the labels.

3.2 User Agreement Study

ACAP is not an easy task, using Fleiss Kappa statistic and Krippendorff's alpha coefficient we compare the agreement between annotators. *Fleiss Kappa* statistic [8] calculates agreement between multiple scorers as in Eq. 1. The interpretation of Kappa value is, <0 = poor agreement, [0.01, 0.20] = slight agreement, [0.21, 0.40] = fair agreement, [0.41, 0.60] = Moderate agreement, [0.61, 0.80] = substantial agreement, and [0.81, 1.00] = perfect agreement.

$$FK = \frac{\sum_{i=1}^{N}\sum_{j=1}^{k} v_{ij}^2 - Nm}{Nm(m-1)} \tag{1}$$

To account for the error magnitude that a scorer makes, we use *Krippendorff's alpha coefficient* [18], this statistic consider the distance between labels given by multiple scorers as in Eq. 2. $\alpha \in [0, 1]$, where 0 = random scoring, and 1 = perfect scoring.

$$\alpha = 1 - \frac{(n-1)\sum_i \sum_j o_{ij} \times \delta_{ij}^2}{\sum_i \sum_j v_i \times v_j \times \delta_{ij}^2} \tag{2}$$

Table 2 gives the agreement scores between annotators per dataset. We noticed that labeling WSJ is the hardest. The raters' agreement is "Fair" for WSJ, TG, DM, and MW and "Moderate" for FN, based on Fleiss Kappa. The Krippendorff's score[2] is between 42% and 66% across outlets. Results indicate the difficulty of assigning a category for comments in general.

Table 2. Agreement analysis for annotators labels.

Dataset	WSJ	TG	DM	MW	FN
Fleiss Kappa	0.22	0.36	0.37	0.40	0.45
Krippendorff's α	0.42	0.60	0.61	0.64	0.66

4 Methods

4.1 BERTAC Model - Joint Modeling of Article and Comments

BERTAC leverages BERT$_{base}$ architecture, which allows us to learn more expressive embeddings for articles and comments. To solve ACAP we combine an article and its comment into a pair of segments and separate them with the special token [SEP]. Our goal is to make use of BERT's self-attention mechanism and bidirectional cross attention in an end-to-end fashion to encode the relevance between an article and its comments. One challenge in this setting is that of determining the length (in words) of the input segments that allow the network architecture to encode useful article-comment relations. We explore multiple lengths for each dataset based on the average length of the articles.

4.2 BA-BC Model - Disjoint Modeling of Article and Comments

Our problem consists of two main parts. The first part is the article, where the language is formal and usually formed of long sequences. The second part is the comment, where comments are often written in informal language and consist of short sequences. We explore if a mixture of different pre-trained models can better solve ACAP, we call it BA-BC shown in Fig. 2. The model consists of two stages. The first stage *Fine-tune on News* has two sides: (BA) is a BERT$_{base}$ architecture trained and fine-tuned on articles; the second side (BC) is a BERTweet architecture trained and fine-tuned on comments. BERTweet is a pre-trained model proposed by [26] and trained on English Tweets, the underlie architecture is RoBERTa [21]. Their results show that BERTweet outperform RoBERTa$_{base}$ and XLM-R$_{base}$ [4] in many tasks.

In the second stage, *Classification* stage, the output from the first stage is fed into a fully-connected layer with a ReLU non-linearity. To get full advantage

[2] Calculated by http://dfreelon.org/utils/recalfront/recal-oir/ software using ordinal setting.

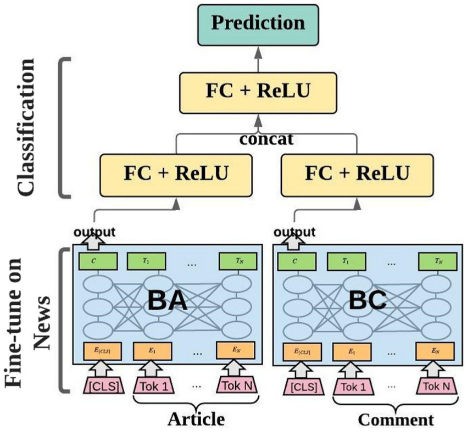


Fig. 2. BA-BC model for ACAP

of pre-trained models, we designed two versions of BA-BC. The first is called *BC-BA-Emb*, where the output from the *Fine-tune on News* stage contains the Embeddings created by both sides without seeing any training examples from our datasets. The second model, called *BC-BA-Fine-tune*, is additionally fine-tuned in the *Fine-tune on News* stage. Left-side (BA) is fine-tuned on articles and labels and right-side (BC) is fine-tuned on comments and labels. Then, last hidden state of fine-tuned parts is sent to the second, *Classification* stage.

4.3 Ordinal Classification Loss

We introduce the ordinal classification loss, that accounts for the distance between the predicted and the actual class. Here, we multiply the loss for each example with a weight that is calculated according to Eq. 3, where $k = 4$ (number of classes), y_i is the i^{th} actual label and $\bar{y_i}$ is the i^{th} predicted label.

$$weight = 1 + \frac{|\bar{y_i} - y_i|}{k - 1} \tag{3}$$

If the algorithm chooses the right class, the weight is 1, which is equal to original loss. If the model predicts a wrong category, the classification loss is multiplied by 2, 3, or 4 based on the distance between the real class and predicted class. This loss depends on the difference between the predicted and the correct class, and the softmax error during predicting the actual class. We incorporated the proposed loss to BERTAC and compare it to original loss in Sect. 6.4.

5 Experimental Setup

5.1 Environment

We run all deep models on four large nodes with 512 GB of DDR4 2400 MHz RAM. Each has two sockets with Intel Xeon E5-2667 v4 3.2 GHz processors, and every node contains two NVIDIA Tesla P100 PCIe 12 GB GPUs and SSDs as local hard drives. We ran doc2vec on a 64-bit processor, Intel Core i7-6700 CPU @ 2.60 GHz with four cores and 16.0 GB RAM.

5.2 Model Evaluation

To evaluate the models performance, we use both simple and weighted accuracy since our labels are ordinal. With simple accuracy metric, which is calculated as the percent of correct predictions, predicting 0 or 2 for label 3 are counted as equal mistakes. Instead, we use weighted accuracy to calculate error by summing the absolute difference between predicted class $\widetilde{s_i}$ and ground truth $\bar{s_i}$. Model's error on a dataset is calculated by dividing that error by the number of examples and max difference D between predicted classes. This constant is 3 in the given multi-class settings. The following formula computes the weighted accuracy, where m is the number of examples:

$$WACC = 1 - \frac{\sum_{i=1}^{m} |\widetilde{s_i} - \bar{s_i}|}{mD} \tag{4}$$

For all supervised models, experiments are repeated five times on different randomized split. The dataset is split into 70:20:10 ratio for training, testing, and cross-validation, respectively. We report the mean and standard deviation.

5.3 Comparison Models

A key challenge in solving ACAP is to establish a similarity of article-comment pair that is indicative of the relevance of the comment to the message of the article. We seek models that can learn long text representations using context and capitalize on the sequential nature of words in a comment. Besides, a model has to be able to embed two types of sequences: articles, which follow formal language, and comments, which may follow colloquial language.

First, *Doc2vec*. Since it is unsupervised we use all data, comments, and articles available in each outlet to learn the documents embedding. We learn separate embeddings per outlet to account for the linguistic style accommodations and other biases across outlets [22]. Then, we calculate the cosine similarity for all labeled pairs and assign a class for each pair based on the rules written below. A represents articles, and C represents comments. We experiment with thresholds in increments of 0.1. The ones used below give the best performance:

$$f(A_i, C_i) = \begin{cases} 0, & \text{if } cos(A_i, C_i) \leq 0.4 \\ 1, & \text{otherwise, if } cos(A_i, C_i) \leq 0.6 \\ 2, & \text{otherwise, if } cos(A_i, C_i) \leq 0.8 \\ 3, & \text{otherwise.} \end{cases}$$

The second baseline is *Siamese LSTM*, which consists of two LSTM that learn representations of articles and comments in separate modules. On top of these modules, there is a joint loss computation module, which computes the similarity between vectors and uses a dense layer to predict the label. Following

[24], we use the Manhattan distance to calculate the similarity. We utilize a sparse categorical cross-entropy with a softmax activation function.

BA-BC: To produce the vector representation of articles and comments two main steps are required, embeddings and classifications. In *BA-BC-Emb* model each side learns the embeddings by fine-tuning on articles and comments, respectively, in an unsupervised manner (without using labels). However, for *BA-BC-Fine-tune*, each side is fine-tuned on articles and comments in a supervised way (using their associated labels). Later on, the vector representation of the last hidden layer is injected into the classification stage. The *Fine-tune on News* stage is repeated for 3 epochs and the final representation is fed into the *Classification* stage, in which cross-entropy with a softmax activation function is applied as a loss. The classification stage is repeated for 300 epochs.

BERTAC: We leverage BERT$_{base}$ where it consists of 12 layers, hidden layer size is 768, number of self-attention heads is 12, and the total number of parameters is 110M. BERTAC is trained in two modes, cased and uncased, where letter casing is considered in the first while all letters are converted to small letters in the later. We trained the model for 6 epochs.

6 Results and Discussion

We study the complexity of this problem. In addition, we thoroughly study the multi-class datasets by employing and analyzing multiple models and evaluation measures to understand their behavior and identify the ones that better capture the semantic between an article and its comments. We also analyze the effect of the proposed Ordinal Classification Loss.

6.1 Binary Versus Multiclass ACAP

To characterize the complexity of our problem we compare a binary dataset and multiclass dataset using BERTAC. In Table 3, we can see that the model maximal performance is around 92% when the problem is binary. The accuracy drops between 13%–23% when we have 4 classes. Even though having multiple classes helps people understand the relationship between an article and its comments better, it becomes harder for a model to capture the semantics and knowledge that is needed to distinguish some labels.

Table 3. Test accuracy (in %) for BERTAC. The *B* row represents binary dataset and *M* row represents multiclass dataset. The average test accuracy of 5 experiments is reported with standard deviation.

Model	Dataset	FN	TG	MW	WSJ	DM
BERTAC	B	88.30(1.42)	92.45(1.45)	88.64(2.50)	85.07(0.97)	90.46(1.75)
	M	75.60(1.81)	74.58(6.49)	75.26(4.52)	63.17(2.44)	67.36(3.46)

6.2 Models Comparisons on Multiclass ACAP

As shown in Fig. 3, Doc2Vec performance is the worst, despite training on much larger corpus of unsupervised data. Siamese LSTM accuracy exceeds Doc2Vec with a boost between 1%–27%. BA-BC-Emb and BA-BC-Finetune outperform Siamese LSTM, the current SOTA for this problem. Both have an increase of 4%–17% in accuracy and 2%–4% in weighted accuracy over Siamese LSTM. Compering BA-BC-Emb and BA-BC-Finetune we observe that they have similar performance. BERTAC however outperforms all other models in both metrics when trained with the original loss function. In addition, we experiment with increasing the number of training points by merging the datasets. We note that increasing training examples does not improve any of the proposed models over BERTAC, which aligns with our hypothesis that BERTAC can outperform other models using only a small number of training examples.

Our experimental design is such that the number of articles varies between 300 and 10, 000 across outlets as shown in Table 1. However, we label a fixed number of random pairs. Therefore, the chance of selecting multiple comments for a single article is much larger at FN, TG and MW compared to WSJ and DM. A higher average accuracy is obtained on FN, TG and MW than on WSJ and DM. This suggests that when the model is trained on the same article with different comments it can learn the pattern and make better predictions.

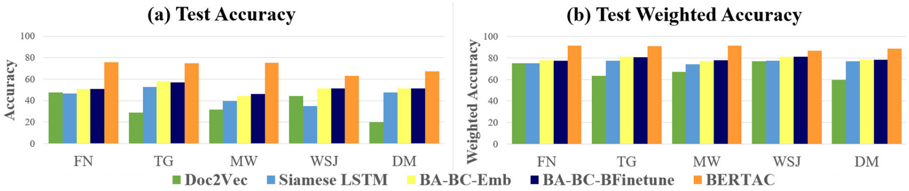


Fig. 3. The average test accuracy of 5 experiments (in %) for all models. (a) shows the accuracy results and (b) shows the weighted accuracy given by Eq. 4.

6.3 Weighted Versus Un-Weighted Accuracy

Comparing the algorithms outcomes on the weighted and unweighted accuracy, we note that their relative performance is unchanged: the model that has the lowest unweighted accuracy has the smallest weighted accuracy performance as well. The same relation stands for the highest results. There are a couple of explanations, first, most of the wrongly classified instances are mixed with its neighboring classes. Second, a proportional number of stronger misclassifications (where the distance between the actual and predicted category is larger than 1) is present across news outlets.

The analysis of weighted accuracy helps to gain additional insight into the models and the problem hardness. For instance, by comparing the weighted and unweighted accuracy scores, we get a better idea of how well a model learns, since it accounts for the strength of the error, penalizing more the mistakes on harder examples. We observe a large gap between accuracy and weighted accuracy results, where the error is between 2 or 3 times smaller in most cases. This indicates that when a model misclassifies an example, often, the model predicts one of the neighboring classes to the correct class. For example, in *WSJ*, *Doc2Vec*'s accuracy = 44.41% which is higher than of *Siamese LSTM*'s accuracy = 34.81%. However, the weighted accuracy shows the opposite: *Doc2Vec*'s accuracy = 76.88% and *Siamese LSTM*'s accuracy = 77.60%. This indicates that *Siamese LSTM* is able to understand the problem better and address the natural order of the classes during training.

6.4 Ordinal Classification Loss

We designed this experiment to investigate the effect of the *ordinal loss* on BERTAC. We hypothesize that BERTAC trained with the proposed *ordinal loss* will outperform the original loss.

We find that *proposed ordinal loss* has no significant advantage compared to *original loss*, where both losses have similar performance. To better understand this problem we investigate those instances where the annotators highly agree with each other in the labeling task: σ between the annotators's labels is either 0, which means that they all agree, or 0.5, which means that only one annotator disagree, with difference of 1 and this does not affect the final label after aggregating the annotators labels. We call this the *high agreement experiment* indicated by BERTAC$_{high}$ in Fig. 4. On the other hand, the *low agreement experiment* indicated by BERTAC$_{low}$, which contains only examples where σ between the annotators' labels, is higher than 0.5. We find that for some datasets BERTAC$_{low}$ accuracy is slightly higher than BERTAC$_{all}$ and BERTAC$_{high}$. However, looking into the high σ we can see that the model is not consistent compared to BERTAC$_{high}$, and the number of examples are much fewer that BERTAC$_{all}$. Analyzing the both losses for BERTAC$_{high}$, where annotators highly agree with each other, we find that the ordinal loss is higher but not significantly. The improvement in accuracy is between 1%–5% and 1%–3% in the weighted accuracy. However, if we study the behavior across the models, we can see that ordinal loss behaves somehow differently across experiments. For examples in Fig. 4, we can see that both BERTAC$_{high}$ and BERTAC$_{low}$ agree that ordinal loss is equal or better than original, where BERTAC$_{all}$ disagree. This brings the following question: *if the model were capable to vote for the best possible prediction from different model would this improve results?*

To answer the previous question, we calculate the average vote prediction from different models in order to obtain the best prediction. We consider the predictions from BERTAC uncased trained with ordinal loss and original loss, and BERTAC cased trained ordinal loss. Table 4 shows that the voting system improves the results with respect to accuracy and standard division.

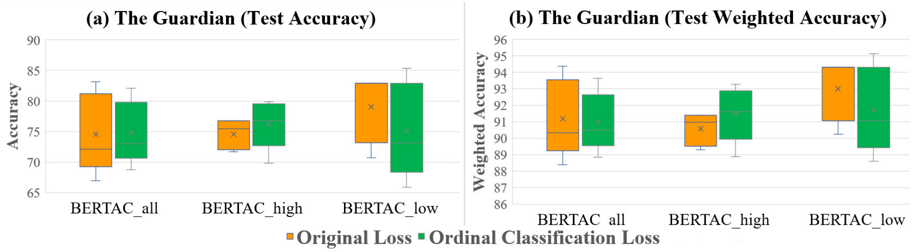


Fig. 4. The Guardian average test accuracy of 5 experiments (a) show the accuracy results and (b) weighted accuracy. The subscript beside BERTAC indicates the experiment type: *all* = trained on all labeled examples were used, *high* = trained on examples with high agreement score, and *low* = trained on examples with low agreement score.

Table 4. Accuracy results in % for BERTAC trained with ordinal loss (BERTAC$_{ord}$) and BERTAC trained with different settings and losses (BERTAC$_{vote}$).

Model	FN	TG	MW	WSJ	DM
BERTAC$_{ord}$	75.08 (4.19)	74.78 (5.15)	71.08 (3.47)	64.45 (3.36)	68.42 (1.49)
BERTAC$_{vote}$	**76.73 (2.15)**	**76.00 (6.16)**	**74.00 (3.40)**	**64.00 (2.77)**	**69.02 (1.87)**

7 Conclusion

In this work, we define the article-comment alignment problem (ACAP) and propose an effective approach to predict the level of relatedness between a comment and an article. We compare Doc2Vec, Siamese LSTM, BA-BC, and BERTAC models and study the performance improvement across them. The results reported in this work show that a joint modeling of article-comments, i.e., BERTAC, is able to capture a deeper level of semantic relatedness between comments and news articles, and help predict better the relevance level of a comment to the content of an article than the current state-of-the-art and other proposed methods.

Even though accuracy values are close, detailed analysis shows that BERTAC trained with proposed ordinal loss perform better than BERTAC on the original BERT loss. With the proposed loss, we can identify common mistakes by annotators and potentially use them to improve the performance of downstream applications, which we will explore in the future.

Acknowledgements. This research was supported in part by the NSF grant IIS-8142183. In addition, this research includes calculations carried out on HPC resources supported in part by the National Science Foundation through major research instrumentation grant number 1625061 and by the US Army Research Laboratory under contract number W911NF-16-2-0189.

References

1. Almoqbel, M.Y., Wohn, D.Y., Hayes, R.A., Cha, M.: Understanding Facebook news post comment reading and reacting behavior through political extremism and cultural orientation. Comput. Hum. Behav. **100**, 118–126 (2019). https://doi.org/10.1016/j.chb.2019.06.006. http://www.sciencedirect.com/science/article/pii/S0747563219302250
2. Bastos, M., Mercea, D.: Parametrizing Brexit: mapping Twitter political space to parliamentary constituencies. Inf. Commun. Soc. **21**(7), 921–939 (2018). https://doi.org/10.1080/1369118X.2018.1433224
3. Celli, F., Stepanov, E.A., Poesio, M., Riccardi, G.: Predicting Brexit: classifying agreement is better than sentiment and pollsters. In: Nissim, M., Patti, V., Plank, B. (eds.) Proceedings of the Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media, PEOPLES@COLING 2016, Osaka, Japan, 12 December 2016, pp. 110–118. The COLING 2016 Organizing Committee (2016). https://www.aclweb.org/anthology/W16-4312/
4. Conneau, A., et al.: Unsupervised cross-lingual representation learning at scale. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8440–8451. Association for Computational Linguistics (July 2020). https://doi.org/10.18653/v1/2020.acl-main.747. https://www.aclweb.org/anthology/2020.acl-main.747
5. Das, M.K., Bansal, T., Bhattacharyya, C.: Going beyond Corr-LDA for detecting specific comments on news & blogs. In: Carterette, B., Diaz, F., Castillo, C., Metzler, D. (eds.) 7th ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, 24–28 February 2014, pp. 483–492. ACM (2014). https://doi.org/10.1145/2556195.2556231
6. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, vol. 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/n19-1423
7. Diakopoulos, N., Naaman, M.: Towards quality discourse in online news comments. In: Hinds, P.J., Tang, J.C., Wang, J., Bardram, J.E., Ducheneaut, N. (eds.) Proceedings of the 2011 ACM Conference on Computer Supported Cooperative Work, CSCW 2011, Hangzhou, China, 19–23 March 2011, pp. 133–142. ACM (2011). https://doi.org/10.1145/1958824.1958844
8. Fleiss, J.L.: Measuring nominal scale agreement among many raters. Psychol. Bull. **76**(5), 378–382 (1971). https://doi.org/10.1037/h0031619
9. Friedman, H.H., Amoo, T., Rating the rating scales: Rating the rating scales. J. Mark. Manage. **9**, 114–123 (1999). https://ssrn.com/abstract=2333648
10. Georgakopoulos, S.V., Tasoulis, S.K., Vrahatis, A.G., Plagianakos, V.P.: Convolutional neural networks for toxic comment classification. In: Proceedings of the 10th Hellenic Conference on Artificial Intelligence, SETN 2018, Patras, Greece, 09–12 July 2018, pp. 35:1–35:6. ACM (2018). https://doi.org/10.1145/3200947.3208069
11. Gottipati, S., Jiang, J.: Finding thoughtful comments from social media. In: Kay, M., Boitet, C. (eds.) Proceedings of the 24th International Conference on Computational Linguistics (Technical Papers), COLING 2012, Mumbai, India, 8–15 December 2012, pp. 995–1010. Indian Institute of Technology Bombay (2012). https://www.aclweb.org/anthology/C12-1061/

12. Gruetze, T., Krestel, R., Naumann, F.: Topic shifts in StackOverflow: ask it like socrates. In: Métais, E., Meziane, F., Saraee, M., Sugumaran, V., Vadera, S. (eds.) NLDB 2016. LNCS, vol. 9612, pp. 213–221. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41754-7_18
13. He, L., Han, C., Mukherjee, A., Obradovic, Z., Dragut, E.: On the dynamics of user engagement in news comment media. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **10**(1), 1342 (2020). https://doi.org/10.1002/widm.1342
14. He, L., Shen, C., Mukherjee, A., Vucetic, S., Dragut, E.: Cannot predict comment volume of a news article before (a few) users read it. In: ICWSM. AAAI Press (2021)
15. Hille, S., Bakker, P.: Engaging the social news user. J. Pract. **8**(5), 563–572 (2014). https://doi.org/10.1080/17512786.2014.899758
16. Hosseini, H., Kannan, S., Zhang, B., Poovendran, R.: Deceiving Google's perspective API built for detecting toxic comments, CoRR abs/1702.08138 (2017). http://arxiv.org/abs/1702.08138
17. Hou, L., Li, J., Li, X., Tang, J., Guo, X.: Learning to align comments to news topics. ACM Trans. Inf. Syst. **36**(1), 91–931 (2017). https://doi.org/10.1145/3072591
18. Krippendorff, K.: Computing Krippendorff's alpha-reliability. Scholarly Commons (2011)
19. Likert, R.: A technique for the measurement of attitudes. Arch. Psychol. **22**, 5–55 (1932)
20. Liu, Q., Dragut, E., Mukherjee, A., Meng, W.: FLORIN: a system to support (near) real-time applications on user generated content on daily news. Proc. VLDB Endow. **8**(12), 1944–1947 (2015)
21. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach, CoRR abs/1907.11692 (2019). http://arxiv.org/abs/1907.11692
22. Mahendiran, A., et al.: Discovering evolving political vocabulary in social media. In: 2014 International Conference on Behavioral, Economic, and Socio-Cultural Computing, BESC 2014, pp. 1–7 (2014). https://doi.org/10.1109/BESC.2014.7059504
23. Mishne, G., Glance, N.: Leave a reply: an analysis of weblog comments. In: 3rd Annual Workshop on the Weblogging Ecosystem (2006)
24. Mueller, J., Thyagarajan, A.: Siamese recurrent architectures for learning sentence similarity. In: Schuurmans, D., Wellman, M.P. (eds.) Proceedings of the 13th AAAI Conference on Artificial Intelligence, 12–17 February 2016, Phoenix, Arizona, USA. pp. 2786–2792. AAAI Press (2016). http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12195
25. Mullick, A., Ghosh, S., Dutt, R., Ghosh, A., Chakraborty, A.: Public sphere 2.0: targeted commenting in online news media. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11438, pp. 180–187. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15719-7_23
26. Nguyen, D.Q., Vu, T., Nguyen, A.T.: BERTweet: a pre-trained language model for English tweets. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 9–14. Association for Computational Linguistics (October 2020). https://doi.org/10.18653/v1/2020.emnlp-demos.2. https://www.aclweb.org/anthology/2020.emnlp-demos.2
27. Park, A., Hartzler, A., Huh, J., Hsieh, G., McDonald, D.W., Pratt, W.: "How did we get here?" topic drift in online health discussions. J. Med. Internet Res. **18**(11), 284 (2016). https://doi.org/10.2196/jmir.6297. http://www.jmir.org/2016/11/e284/

28. Ruiz, C., Domingo, D., Micó, J.L., Díaz-Noci, J., Meso, K., Masip, P.: Public sphere 2.0? The democratic qualities of citizen debates in online newspapers. Int. J. Press Polit. **16**(4), 463–487 (2011). https://doi.org/10.1177/1940161211415849
29. Sil, D.K., Sengamedu, S.H., Bhattacharyya, C.: Readalong: reading articles and comments together. In: Srinivasan, S., Ramamritham, K., Kumar, A., Ravindra, M.P., Bertino, E., Kumar, R. (eds.) Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, 28 March–1 April 2011 (Companion Volume), pp. 125–126. ACM (2011). https://doi.org/10.1145/1963192.1963256
30. Singer, J.B.: Separate spaces: discourse about the 2007 Scottish elections on a national newspaper web site. Int. J. Press/Polit. **14**(4), 477–496 (2009). https://doi.org/10.1177/1940161209336659
31. Stanojevic, M., Alshehri, J., Dragut, E., Obradovic, Z.: Biased news data influence on classifying social media posts. In: Proceedings of the 3rd International Workshop on Recent Trends in News Information Retrieval, co-located with 42nd International ACM Conference on Research and Development in Information Retrieval (SIGIR 2019), Paris, France, 25 July 2019. CEUR Workshop Proceedings, vol. 2411, pp. 3–8. CEUR-WS.org (2019). http://ceur-ws.org/Vol-2411/paper1.pdf
32. Stanojevic, M., Alshehri, J., Obradovic, Z.: Surveying public opinion using label prediction on social media data. In: International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2019, Vancouver, British Columbia, Canada, 27–30 August 2019, pp. 188–195. ACM (2019). https://doi.org/10.1145/3341161.3342861
33. Weber, P.: Discussions in the comments section: factors influencing participation and interactivity in online newspapers' reader comments. New Media Soc. **16**(6), 941–957 (2014). https://doi.org/10.1177/1461444813495165
34. Yan, T., Keusch, F.: The effects of the direction of rating scales on survey responses in a telephone survey. Publ. Opin. Q. **79**(1), 145–165 (2015). https://doi.org/10.1093/poq/nfu062
35. Yang, F., Dragut, E., Mukherjee, A.: Claim verification under positive unlabeled learning. In: ASONAM (2020)
36. Yang, F., Dragut, E., Mukherjee, A.: Predicting personal opinion on future events with fingerprints. In: COLING (December 2020)
37. Zhang, Y., Yang, F., Zhang, Y., Dragut, E., Mukherjee, A.: Birds of a feather flock together: satirical news detection via language model differentiation (2020)
38. Ziegele, M., Quiring, O.: Conceptualizing online discussion value: a multidimensional framework for analyzing user comments on mass-media websites. Ann. Int. Commun. Assoc. **37**(1), 125–153 (2013). https://doi.org/10.1080/23808985.2013.11679148

An E-Commerce Dataset in French for Multi-modal Product Categorization and Cross-Modal Retrieval

Hesam Amoualian[3], Parantapa Goswami[1](✉), Pradipto Das[2],
Pablo Montalvo[1], Laurent Ach[1], and Nathaniel R. Dean[2]

[1] Rakuten Institute of Technology (RIT), Paris, France
{parantapa.goswami,pablo.montalvo,laurent.ach}@rakuten.com
[2] Rakuten Institute of Technology (RIT), Boston, USA
{pradipto.das,nathaniel.dean}@rakuten.com
[3] Tessella Altran, Paris, France
hesam.amoualian@altran.com

Abstract. A multi-modal dataset of ninety nine thousand product listings are made available from the production catalog of Rakuten France, a major e-commerce platform. Each product in the catalog data contains a textual title, a (possibly empty) textual description and an associated image. The dataset has been released as part of a data challenge hosted by the SIGIR ECom'20 Workshop. Two tasks are proposed, namely a principal large-scale multi-modal classification task and a subsidiary cross-modal retrieval task. This real world dataset contains around 85K products and their corresponding product type categories that are released as training data and around 9.5K and 4.5K products are released as held-out test sets for the multi-modal classification and cross-modal retrieval tasks respectively. The evaluation is run in two phases to measure system performance, first on 10% of the test data, and then on the rest 90% of the test data. The different systems are evaluated using macro-F1 score for the multi-modal classification task and recall@1 for the cross-modal retrieval task. Additionally, a robust baseline system for the multi-modal classification task is proposed. The top performance obtained at the end of the second phase is 91.44% macro-F1 and 34.28% recall@1 for the two tasks respectively.

Keywords: E-commerce dataset · Multimodal classification · Cross-modal retrieval

1 Introduction

Rakuten Multi-modal Product Data Classification and Retrieval challenge is organized by Rakuten Institute of Technology, the research and development department of Rakuten Group. This challenge focuses on the topic of

H. Amoualian—Most of the work was performed while at RIT-Paris.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 18–31, 2021.
https://doi.org/10.1007/978-3-030-72113-8_2

large-scale multi-modal (text and image) classification, where the goal is to predict each product's *type code* as defined in the catalog of Rakuten France, and cross-modal retrieval, aiming to retrieve the most relevant image of a product given the textual title and description. The cataloging of product listings through some type of text or image categorization is a fundamental problem for any e-commerce marketplace, with applications ranging from personalized search and recommendations to query understanding. Manual and rule-based approaches to categorization are not scalable since commercial products are organized in many and sometimes thousands of classes. When actual users categorize product data, it has often been seen that not only the text of the title and description of the product is useful but also the associated images.

Advances in this area of research have been limited due to the lack of real world large-scale multi-modal product data from actual commercial catalogs. This data challenge presents several interesting research aspects due to the intrinsic noisy nature of the product labels and images, the size of modern e-commerce catalogs, and a highly skewed data distribution.

The dataset is made publicly available through Rakuten Data Release Platform[1]. We hope that by making the data publicly available, our proposed tasks will attract more research institutions and industry practitioners, who do not have the opportunity to contribute their ideas due to the lack of actual commercial e-commerce catalog data.

Principle contributions of this article:

- We release a real-world e-commerce multimodal product dataset.
- Two tasks are proposed on this dataset, namely cross-modal retrieval and classification, as well as a baseline.
- We report methods and results provided by the data challenge participants who outperformed the baseline using both textual and visual modalities.

2 Related Work

In recent years, multi-modal learning for various tasks has attracted significant attention of researchers. The most common task in this domain, which corresponds to the second task of this challenge, is multi-modal retrieval. The seminal work of [16] and [9] for joint embedding learning brought insights from how text and image encoders could be trained together using modified triplet losses. More recently transformers have been employed to this use-case [22,23] which achieved state-of-the-art performances on retrieval. Another study in [3] mentions attribute extraction for fashion items, in a supervised way. Multi-modal classification for social media data is explored by [7], whereas [4] tackles the problem of noisy labels in their weakly supervised approach on realistic datasets. Still on weakly supervised multi-modal approaches, the work of [10] explores how using attention from visual features improves clustering of textual attributes. Recently, AutoKnow from [6] purposes a way to build a broad knowledge graph

[1] Rakuten France Multimodal Dataset in https://rit.rakuten.co.jp/data_release/.

for a thousand types of products. It includes a set of machine learning methods to automate knowledge enrichment and ontology construction for a large number of products.

Retail product datasets are typically split between product price datasets, product search relevancy and product reviews datasets. Many such datasets are publicly available. However, some otherwise significant contributions on image search such as [27] unfortunately cannot share their proprietary dataset. For this data challenge, our dataset is product-oriented. Most publicly available product datasets are either uniquely image-based (e.g. [1]) or uniquely non-visual (e.g. [2]). One can find small excerpts of product datasets from different companies in the competitive machine learning website Kaggle.com (some example uses are listed in [25]). Notably Amazon [21] provides both image features extracted by a CNN and metadata for recommendation. Rakuten has also organized an item classification data challenge [19] based on textual title and description of the items. However our data challenge further extends the scope by introducing multiple modalities. To the best of our knowledge, the dataset to be released publicly as part of our hosted data challenge, is the only existing large-scale multi-modal retail product dataset available in French containing product titles, top level categories, descriptions and images.

3 Challenge Description

In the taxonomy of Rakuten France, products sharing the same product type code share the same exact array of attributes fields and possible values. Product type codes are numbers that match a generic product name, such as 1500 - `Watches`, 120 - `Laptops`, and so on. In that sense, the type code of a product is its category label.

For example, in the product catalog of Rakuten France, a product with a French title *Klarstein Présentoir 2 Montres Optique Fibre* is associated with an image and sometimes with an additional description. This product is categorized with a product type code of **1500**, signifying watches. There are other products with different titles, images and with possible descriptions, which are under the same product type code. Given these information on the products, like the example above, this challenge proposes that participating teams build and submit systems that classify previously unseen products into their corresponding product type codes.

The main tasks for this challenge are as follows:

Task 1 - The primary **Multi-modal classification** task: Given a training set of products and their product type codes, the aim is to predict the corresponding product type codes for an unseen held out test set of products. The systems are free to use the available textual titles and/or descriptions whenever available and additionally the images to allow for true multi-modal learning.

Task 2 - The secondary **Cross-modal retrieval** task: Here the systems have to predict the correct image for a product given its textual content. Doubtlessly this task is more challenging than the classification task.

The difficulty in solving the tasks stems from the following observations:

- Highly imbalanced number of samples within the classes.
- Length of titles in terms of words can vary considerably, from two or three words to about fifty words.
- Descriptions, when present, may be a verbose representation of the product rather than a very specific one with precisely defined attributes for the product.
- Images may not be "clean". Some images could be of low quality, while some images may have text in them as often found in a banner or book/media covers.

4 Data Description

Rakuten France has released approximately 99K product listings in `tsv` format, including a training (84,916) and two test sets (9,372 samples for the classification and 4,440 samples for the retrieval task). The training and test splits have been obtained using random sampling stratified by product type codes, i.e. the product categories. Each product in the dataset consists of product title, product description, product image and its corresponding product type code. The dataset is distributed over 27 unique product type categories.

The complete catalog of products of Rakuten France is much larger than 99 thousand listings and contains much more than 27 product type codes. Among all the available product type codes, initially 27 are manually identified based on how often the products belonging to these type codes need to be categorized and how much GMV[2] they generate. This choice makes the dataset more grounded in reality, as high classification performance is easily correlated with strong business impact. We do stratified sampling so that the dataset represents as accurately as possible the original distribution of items. For each of these identified type codes, a 10% sample of the entire product catalog is randomly selected to create the dataset for our data challenge.

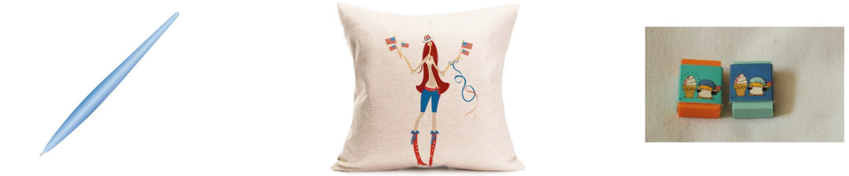
The training data file is in a tab-separated values (`tsv`) format where each line contains a product title, (possibly empty) description, product id, id of the associated image and its corresponding product type code. Additionally an image folder is supplied containing all the training images. One can use the image id and product id to obtain the associated image file from the image folder. The test data file for the primary task of multi-modal classification contains all the fields as training data file except the product type code, and similarly image id and product id can be used to obtain the corresponding image files from the

[2] Gross Merchandise Volume (GMV) is the total monetary value for merchandise sold through a particular marketplace over a certain period of time.

test image folder. The test data file for the cross-modal retrieval task contains only product title, (possibly empty) description and the product id. Image files for the products for this test set are also provided, but the link between the products and their corresponding image files are not provided in this case.

Table 1. Three samples from the training dataset for multi-modal classification.

Integer_id	Title	Description	Image_id	Product_id
2	Grand Stylet Ergonomique Bleu Gamepad . . .	PILOT STYLE Touch Pen . . .	938777978	201115110
40001	Drapeau Américain Vintage Oreiller . . .	Vintage American Flag Pillow Cases . . .	1273112704	3992402448
84915	Gomme De Collection 2 Gommes Pinguin . . .	NaN	684671297	57203227



(a) Image filename: image_938777978_product_201115110.jpg; Category: Entertainment

(b) Image filename: image_1273112704_product_3992402448.jpg; Category: Household

(c) Image filename: image_684671297_product_57203227.jpg; Category: Books

Fig. 1. Images of the three example products shown in Table 1.

Table 1 displays three different lines of the training file, and Fig. 1 shows the corresponding images for these three products. The examples are selected from the head, torso, and tail of the distribution. Two of which have descriptions, one has not. The images in Fig. 1 show the hardness of the task, especially the cross-modal retrieval task. The hardness stems from the fact that the most prominent part of the images may not be representative of the product, but the totality of the object silhouette needs to be considered.

Also, a tab-separated file containing the mapping between each product type code (abbreviated `Prdtypecode`) and its top level category in English is provided. For example:

It should be noted that the product titles and descriptions are for the vast majority written in French (99%), although, one can find some outlying samples in other languages like English, German, and Spanish. Almost 35% of the products contain an empty description. The images are all squares of dimensions $500 \times 500\ px^2$, which can have white or black borders included.

Prdtypecode	Top level category
2280	Books
1280	Child

4.1 Data Characteristics

The product listing distribution in this dataset over the 27 product type code classes is highly imbalanced, although not following a typical long tail distribution. Figure 2 shows the distribution of the product counts in the training and test dataset across all the product type codes. The largest class contains 12% of the products in the entire training dataset, whereas the smallest one contains only 0.9%.

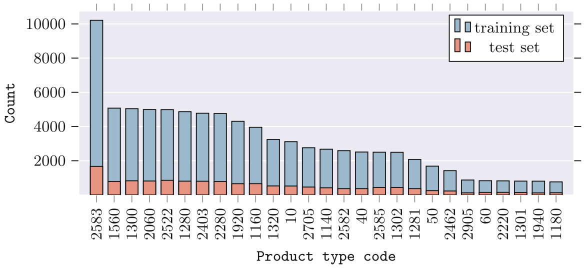


Fig. 2. Product type code frequency distribution in the training and test set

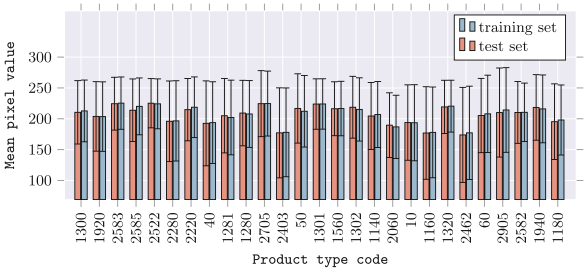


Fig. 3. Mean pixel values for the Red channel of images in the train set (in blue) and the test set (in red), per product type code. The standard deviation is shown as an error bar in the respective colors. The training and test distributions are very similar with regards to these values. (Color figure online)

Figure 3 shows the mean pixel value of the Red channel of images. The similarity of the heights of the bars corresponding to the training and the test sets for each product type code shows that the splits do not suffer from distributional misalignment and hence image models developed using a development set from the training data should generalize to the test set. Similar plots are obtained for the Green and Blue channels as well. Measuring possible discrepancies between image datasets is a hard problem and only proxies exist as images are loaded with intrinsic semantic information that usually need a human to decode. However, in this particular case, the training and test sets *do* come from the same source of data. Hence, the image data do not exhibit domain shift between these two distributions and a high performance on the test set released in phase 1 (see Sect. 5.2), is a good indication of final system performance in the data challenge.

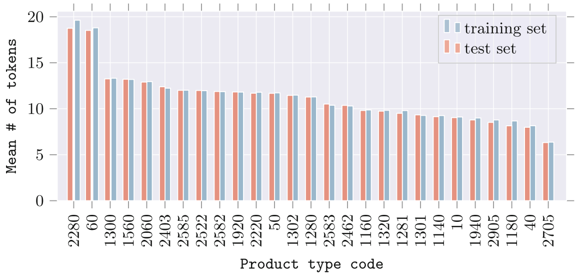


Fig. 4. The average number of tokens per category in the titles. This distribution doesn't show any distinct tail phenomenon.

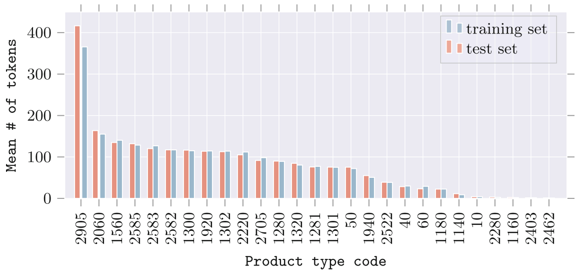


Fig. 5. The average number of tokens per category in the descriptions. This distribution does show a distinct head-torso-tail phenomenon with a substantial probability mass on the torso.

Figures 4 and 5 show the distribution of average token frequency in the training and test datasets across all the product type codes. In this case, too, the respective histograms from the test set closely follow those from the training set. The average product title length in the training set is twelve tokens with the maximum title length of forty tokens. The maximum length of concatenated title and description is 512.

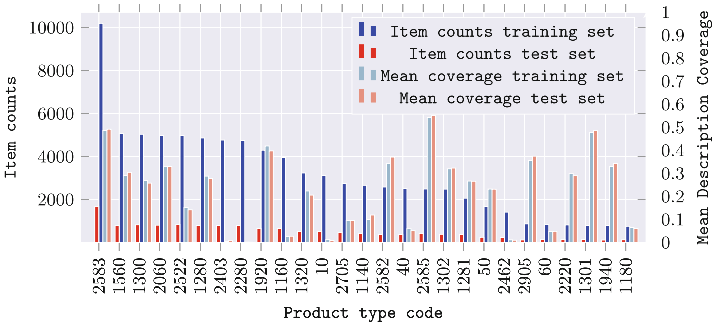


Fig. 6. Mean Description Coverage: Mean of the number of tokens in the titles covered by descriptions in each category and normalized by title lengths.

Another interesting statistics that we mined from the dataset is shown in Fig. 6. In this figure, we summarize the plots in Figs. 2, 4 and 5. For the same ordering of categories in Fig. 2 and across all products in a category, we compute the token-wise coverage of the title words with those in the corresponding product description and then compute the coverage mean with each coverage being normalized by title length. This *mean description coverage* tracks the relevance of the information content in the titles to the verbosity of the corresponding descriptions or a lack thereof.

Figure 6 shows two things – one is that the test distribution of the mean coverage follows the training distribution for the different product types. Second and most importantly, the correlation of the coverage means with the item counts is weak. For instance, product type code 2905 has the highest mean number of tokens in description but ranks only fourth from last in Fig. 4. This is an interesting scenario in e-commerce catalogs and novel research is needed to model the *generation of relevant descriptions from titles and images* so that a consistent catalog can be generated for all products without manual curation. Note that the x-axes in Figs. 2, 4, 5 and 6 are all sorted in descending order of the variables that correspond to the y-axes that are to the left of the plots.

5 Evaluation

5.1 Evaluation Metric

Since in this challenge, we are dealing with many classes with highly asymmetric number of samples, an item weighted metric to evaluate the systems is incapable of revealing the deficiencies of the classification algorithms.

Task 1. The **macro-F1 score** is adopted to evaluate product type code classification on held out test samples. This score is understood as the arithmetic average of per-product type code F1 score. The reason for choosing Macro-F1 is that it is a more unbiased estimate of classification performance for highly imbalanced data, unlike Micro-F1.

Task 2. For the cross-modal retrieval task, the systems are evaluated on **recall at 1 (R@1)** on held out test samples. This is indeed a very strict measure for the task. This score can be defined as the average of the per-sample scoring of 1 if the image returned matches the corresponding title and 0 otherwise.

5.2 Evaluation Phases and Timeline

This data challenge has been held in two phases which includes model building and model evaluation. In each phase, there is a separate test set for each task.

Phase 1 - Model Building. Participants built and tested models on the training data. The models are evaluated on a 10% subset of the test set. This phase was open for a little under three months.

Phase 2 - Model Evaluation. The final models are evaluated on the remaining 90% of the test set. This phase was open for eight days.

6 Baseline Models

Our baseline models for both tasks are dubbed *RIT-Paris Baseline*. The baseline model for product classification is based on Multi-modal BiTransformers [15]. This model combines two pre-trained networks. For the image network, it utilizes ResNet-152 [11]. Input images are normalized, center-cropped, resized at 224 × 224, and each embedding vector has 2048 dimensions. The lingual model uses a bidirectional transformer architecture with pre-trained BERT [5] embeddings.

For the implementation of this model, we use the MMBT library from Pytorch Transformers[3] [26]. Multilingual DistilBert[4] [24] is used for the lingual modality and ResNet-152 for the image modality. All components in the baseline model

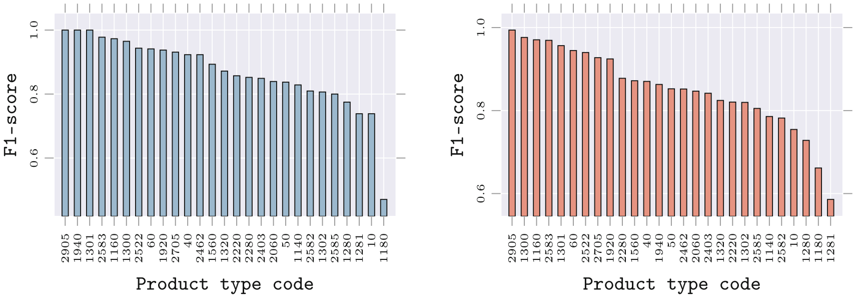


Fig. 7. F1 scores from the proposed baseline model on the test sets from phase 1 (left) and 2 (right) for the classification task.

[3] https://huggingface.co/transformers/summary.html.

[4] https://huggingface.co/distilbert-base-multilingual-cased.

are used with default parameter settings that are recommended by the authors and *no tuning* has been performed for the multi-modal classification task.

The macro-F1 scores obtained using the proposed baseline model for the test set released during the first and the second phases of our data challenge on the multi-modal classification task is shown in the table next.

	Phase 1	Phase 2
Score	0.8705	0.8536

Figure 7 shows that the baseline model can achieve high scores for most of the categories. However, not all categories are easy to classify. Specifically, products related to Child (1280, 1281) and Entertainment (1180) top-level categories, have the worst scores (0.73, 0.59, and 0.66 respectively).

The baseline method for the secondary task of cross-modal retrieval is chosen to be a very simple one – given text about the product, we choose an image from the test set randomly at uniform. We next briefly describe the *documented* systems that perform superior to our baselines.

7 Short Descriptions of Top Performing Systems

Team Synerise AI: For the classification task, a 2-stage scheme (separate pre-training for each modality using an efficiently trainable density estimator model [8] with multi-modal fusion) is used yielding a 89.78 macro F1 score. For the cross-modal retrieval task also, the team employs a 2-stage scheme (Optical Character Recognition with the efficiently trainable density estimator model). This methodology is able to yield a 34.28 recall@1 score and place the team at the first position for the secondary retrieval task.

Team Beantown: They first fine-tune feature extractors from French text using CamemBERT [20] and image modalities using BiT [17] respectively, then applies Highway Network based fusion to obtain multi-modal features. These features are then used to train a classifier for the classification task. For the retrieval task, a similarity search method using the FAISS library [14] has been used to retrieve product images from their text titles. This system resulted in 90.22 macro F1 score for the first task and 23.3 recall@1 score for the second task. *They clearly show that the performance for classification is worse when individual modalities are considered separately.*

Team pa_curis: They also use pre-trained CamemBERT for text and pre-trained ResNet152 [12] for image modality to learn uni-modal features and then deploys late decision level fusion to combine the modalities. Using different versions of text and image classifiers and fusion techniques 12 classifiers are obtained and with majority vote based classification decision. Their system won the multi-modal classification task with 91.44 macro F1 score.

Team Alto: They use ResNet to extract image embeddings, a combination of BERT-based transformer and bi-LSTM to encode text and finally a co-attention block is used to correspond between words and images. The learnt image and text embeddings are then concatenated. Furthermore an ensemble is created by stacking multi-modal models with different base architectures and then using another learning strategy that leverages individual model's strengths. This yields a macro F1 of 90.87 for the classification task.

Team Transformers: They extract text features using two different transformer models, namely CamemBERT and FlauBERT [18]. Images are extracted using SE-ResNeXt [13]. These features are then combined using addition, concatenation, and attention maps. Finally boosted late-fusion is used to combine predictions from the models.

7.1 System Performances

Altogether around hundred teams participated in this data challenge. Among them fifteen teams submitted final system results at the end of phase 2. The submitted systems are scored against the gold standard using the metrics defined in Sect. 5.1. Section 5.2 describes the evaluation phases.

Table 2. Top 10 system scores from phase 1 of the evaluation stage.

Task 1: Multimodal classification			Task 2: Cross-modal retrieval		
Rank	Team	Macro-F1	Rank	Team	Recall@1
1	Transformers	91.94	1	Synerise AI	50.23
2	zenit84	91.63	2	changer	46.85
3	Alto	91.63	3	pa_curis	41.89
4	Beantown	90.89	4	Beantown	38.96
5	Synerise AI	89.72	5	Alto	38.29
6	pa_curis	89.65	6	MMG_AI_TEAM	27.25
7	RIT-Paris Baseline	87.05	7	RIT-Paris Baseline	≈1.50
8	tester	86.94	8	Team 11	1.35
9	testers	85.87			
10	MMG_AI_TEAM	84.81			

The results of the final submissions by the various teams are shown in Tables 2 and 3. Table 2 shows the scores of the systems when a 10% random sample of the test set has been released previously with the training set i.e. phase 1 of the evaluation. Tables 3 shows the scores of the systems when the rest 90% of the test set is released i.e. phase 2 of the evaluation. Not all teams submitted their system results for the final and much larger test set released in phase 2. The ranks for the primary multi-modal classification task (task 1) does not change much between phases 1 and 2, showing generalization to typically hold.

Table 3. System scores from the main part (phase 2) of the evaluation stage.

Task 1: Multimodal classification			Task 2: Cross-modal retrieval		
Rank	Team	Macro-F1	Rank	Team	Recall@1
1	pa_curis	91.44	1	Synerise AI	34.28
2	Alto	90.87	2	changer	31.93
3	Transformers	90.53	3	Beantown	23.30
4	zenit84	90.39	4	Alto	19.99
5	Beantown	90.22	5	pa_curis	19.74
6	Synerise AI	89.78	6	MMG_AI_TEAM	15.77
7	MMG_AI_TEAM	86.94	7	RIT-Paris Baseline	≈2.00
8	RIT-Paris Baseline	85.36			
9	Team MLG	64.48			

A bootstrap sampling procedure, similar to the one in [19], is used to evaluate statistical significance of the submitted systems. The bar plots in the Fig. 8 show the medians of the macro-F1 and Recall@1 scores, for tasks 1 and 2 respectively, for each of the submitted systems. The median scores are calculated after sampling the predictions with replacement. The bars in the plot corresponding to the various system submissions are arranged in descending order of the median scores. We assign the same color to bars that overlap in confidence intervals, however, Fig. 8 reveals that all bars are color coded uniquely. Using this statistical test, with 95% confidence, all submitted systems are different.

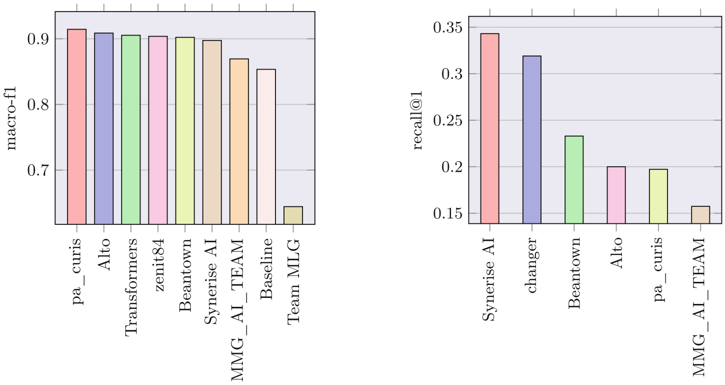


Fig. 8. Results of `bootstrap sampling` performed on the final phase 2 submissions of the participating teams: **median** F1 scores for task 1 (left) and **median** recall@1 for task 2 (right) for each team. Best viewed in color. (Color figure online)

8 Conclusion

The two tasks in our data challenge have been designed to have gradation in difficulty. The classification task is a much easier task and the result from our baseline shows a macro-F1 score of 82% (in the second phase) by just using Multilingual DistilBert on the textual modality. It also shows that using both modalities is helpful to improve the final classification score (85.3%).

The majority of submissions have used pre-trained models that serve as better priors for initialization of embedding vectors. CamemBERT [20] and FlauBERT [18] have been on the forefront of such model choices due to the underlying French corpora on which these models have been trained. Similarly the majority of model choices for image modeling has been using ResNet and its variants [12] in addition to some very recent models such as Google's Big Transfer (BiT) model [17] for pre-trained initialization. Finally the top scoring systems have shown some novelty in fusing the outputs from the uni-modal models using co-attention, highway networks and gradient boosted trees.

In conclusion, we hope that this new dataset can be a de-facto resource for multi-modal classification and cross-modal retrieval on *real world e-commerce data*, which alleviates some of the scarcity issues. The dataset is publicly available through Rakuten Data Release Platform (https://rit.rakuten.co.jp/data_release/) under the name "Rakuten France Multi-modal Product Dataset".

References

1. Fashion-MNIST. https://github.com/zalandoresearch/fashion-mnist
2. Innerwear data from victoria's secret and others. https://www.kaggle.com/PromptCloudHQ/innerwear-data-from-victorias-secret-and-others
3. Cardoso, Â., Daolio, F., Vargas, S.: Product characterisation towards personalisation: learning attributes from unstructured data to recommend fashion products. In: Proceedings of the 24th ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD), pp. 80–89 (2018)
4. Corbiere, C., Ben-Younes, H., Rame, A., Ollion, C.: Leveraging weakly annotated data for fashion image retrieval and label prediction. In: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW) (October 2017). https://doi.org/10.1109/iccvw.2017.266
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding (2018)
6. Dong, X., et al. AutoKnow: self-driving knowledge collection for products of thousands of types. arXiv arXiv:2006.13473 (2020)
7. Duong, C.T., Lebret, R., Aberer, K.: Multimodal classification for analysing social media, CoRR abs/1708.02099 (2017)
8. Dąbrowski, J., et al.: An efficient manifold density estimator for all recommendation systems (2020)
9. Faghri, F., Fleet, D.J., Kiros, J.R., Fidler, S.: VSE++: improved visual-semantic embeddings, CoRR abs/1707.05612 (2017)
10. Han, X., et al.: Automatic spatially-aware fashion concept discovery (2017)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)

12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015). http://arxiv.org/abs/1512.03385
13. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
14. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs. arXiv preprint arXiv:1702.08734 (2017)
15. Kiela, D., Bhooshan, S., Firooz, H., Testuggine, D.: Supervised multimodal bitransformers for classifying images and text (2019)
16. Kiros, R., Salakhutdinov, R., Zemel, R.S.: Unifying visual-semantic embeddings with multimodal neural language models, CoRR abs/1411.2539 (2014)
17. Kolesnikov, A., et al.: Big transfer (BiT): general visual representation learning (2019)
18. Le, H., et al.: FlauBERT: unsupervised language model pre-training for French. In: Proceedings of the 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, 11–16 May 2020, pp. 2479–2490. European Language Resources Association (2020)
19. Lin, Y.C., Das, P., Trotman, A., Kallumadi, S.: A dataset and baselines for e-commerce product categorization. In: Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR 2019, pp. 213–216. Association for Computing Machinery, New York (2019)
20. Martin, L., et al.: CamemBERT: a tasty French language model. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 7203–7219. Association for Computational Linguistics (July 2020). https://www.aclweb.org/anthology/2020.acl-main.645
21. McAuley, J., Targett, C., Shi, Q., van den Hengel, A.: Image-based recommendations on styles and substitutes (2015)
22. Park, G., Han, C., Yoon, W., Kim, D.: MHSAN: multi-head self-attention network for visual semantic embedding, CoRR abs/2001.03712 (2020)
23. Qi, D., Su, L., Song, J., Cui, E., Bharti, T., Sacheti, A.: ImageBERT: cross-modal pre-training with large-scale weak-supervised image-text data, CoRR abs/2001.07966 (2020)
24. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter (2019)
25. Sidorov, M.: Attribute extraction from ecommerce product descriptions. CS229 (2018)
26. Wolf, T., et al.: Huggingface's transformers: state-of-the-art natural language processing. arXiv arXiv:1910.03771 (2019)
27. Yang, F., et al.: Visual search at eBay. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (August 2017). https://doi.org/10.1145/3097983.3098162

FedeRank: User Controlled Feedback with Federated Recommender Systems

Vito Walter Anelli, Yashar Deldjoo, Tommaso Di Noia, Antonio Ferrara(✉), and Fedelucio Narducci

Politecnico di Bari, Bari, Italy
{vitowalter.anelli,yashar.deldjoo,tommaso.dinoia,antonio.ferrara,
fedelucio.narducci}@poliba.it

Abstract. Recommender systems have shown to be a successful representative of how data availability can ease our everyday digital life. However, data privacy is one of the most prominent concerns in the digital era. After several data breaches and privacy scandals, the users are now worried about sharing their data. In the last decade, Federated Learning has emerged as a new privacy-preserving distributed machine learning paradigm. It works by processing data on the user device without collecting data in a central repository. We present FedeRank (https://split.to/federank), a federated recommendation algorithm. The system learns a personal factorization model onto every device. The training of the model is a synchronous process between the central server and the federated clients. FedeRank takes care of computing recommendations in a distributed fashion and allows users to control the portion of data they want to share. By comparing with state-of-the-art algorithms, extensive experiments show the effectiveness of FedeRank in terms of recommendation accuracy, even with a small portion of shared user data. Further analysis of the recommendation lists' diversity and novelty guarantees the suitability of the algorithm in real production environments.

Keywords: Recommender systems · Collaborative filtering · Federated learning · Learning to rank

1 Introduction

Recommender Systems (RSs) are well-known information-filtering systems widely adopted for mitigating the information-overload problem. Indeed, the broad amount of items and services has caused a cognitive impairment that takes the name of over-choice, or choice overload. RSs have proved to be very useful in making possible personalized access to these catalogs of items. These systems are generally hosted on centralized servers and train their models by exploiting massive proprietary and sensitive data. However, public awareness related to data collection was spurred and increased. In recent years, an increasing number of countries have introduced regulations to protect user privacy and data security.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 32–47, 2021.
https://doi.org/10.1007/978-3-030-72113-8_3

Representative examples are the GDPR in the European Union [15], the CCPA in California [8], and the Cybersecurity Law in China [42]. Such policies prohibit free data circulation and force personal data to remain isolated and fragmented.

In this context, Google has recently proposed Federated Learning (FL) as a privacy-by-design technique which tackles data isolation while meeting the need for big data [23,33]. FL trains a global machine-learning model by leveraging both users' data and personal devices' computing capabilities. Unlike previous approaches, it keeps data on the devices (e.g., smartphones, tablets, etc.) without sharing it with a central server. Today, FL is considered the best candidate to face the data privacy, control and property challenges posed by the data regulations.

Among the recommendation paradigms proposed in the literature, Collaborative Filtering (CF) demonstrated a very high accuracy [32,47]. The strength of CF recommendation algorithms is that users who expressed similar tastes in the past tend to agree in the future as well. One of the most prominent CF approaches is the Latent Factor Model (LFM) [26]. LFMs uncover users and items latent representation, whose linear interaction can explain observed feedback.

In this paper, we introduce FedeRank, a novel factorization model that embraces the FL paradigm. A disruptive effect of employing FedeRank is that users participating in the federation process can decide if and how they are willing to disclose their private sensitive preferences. Indeed, FedeRank mainly leverages non-sensitive information (e.g., items the user has not experienced). Here, we show that even only a small amount of sensitive information (i.e., items the user has experienced) lets FedeRank reach a competitive accuracy. How incomplete data impacts the performance of the system is an entirely unexplored field. Analogously, it is still to establish the minimum amount of data necessary to build an accurate recommendation system [46]. At the same time, preserving privacy at the cost of a worse tailored recommendation may frustrate users and reduce the "acceptance of the recommender system" [35]. In this work, instead of focusing on how to protect personal information from privacy breaches (that is explored in other active research fields), we investigate how to guarantee the users the control and property of their data as determined by regulations. The work's contributions are manifold due to the number of open challenges that still exist with the FL paradigm. To summarize, our contributions in this paper include:

- the development of the first, to the best of our knowledge, federated pair-wise recommendation system;
- an analysis of the impact of client-side computation amount;
- an investigation on the existing relation between incomplete data and recommendation accuracy, and an analysis of the algorithmic bias on the final recommendation lists, based on the data deprivation amount.

To this extent, we have carried out extensive experiments on three real-world datasets (*Amazon Digital Music*, *LibraryThing*, and *MovieLens 1M*) by considering two evaluation criteria: (a) the accuracy of recommendations measured by exploiting precision and recall, (b) beyond-accuracy measures to evaluate the novelty, and the diversity of recommendation lists. The experimental evaluation shows that FedeRank provides high-quality recommendations, even though it leaves users in control of their data.

2 Related Work

In the last decades, academia and industry have proposed several competitive recommendation algorithms. Among the Collaborative Filtering algorithms, the most representative examples are undoubtedly Nearest Neighbors systems, Latent Factor Models, and Neural Network-based recommendation systems. The Nearest Neighbors scheme has shown its competitiveness for decades. After them, factorization-based recommendation emerged thanks to the disruptive idea of Matrix Factorization (MF). Nevertheless, several generalized/specialized variants have been proposed, such as FM [37], SVD++ [24], PITF [40], FPMC [39]. Unfortunately, rating-prediction-oriented optimization, like SVD, has shown its limits in the recommendation research [34]. Consequently, a new class of *Learning to Rank* algorithms has been developed in the last decade, mainly ranging from point-wise [28] to pair-wise [38], through list-wise [41] approaches. Among pair-wise methods, BPR [38] is one of the most adopted, thanks to its outstanding capabilities to correctly rank with an acceptable computational complexity. Finally, in the last years, methods exploiting the architectures of deep neural networks have established themselves in search and recommendation research.

To make RSs work properly (easing the user decision-making process and boosting the business), they need to collect user information related to attributes, demands, and preferences [20], jeopardizing the user's privacy. In this scenario—and, more generally, in any scenario with a system learning from sensitive data—FL was introduced for meeting privacy shortcomings by horizontally distributing the model's training over user devices [33]. Beyond privacy, FL has posed several other challenges and opened new research directions [21]. In the last years, it has extended to a more comprehensive idea of privacy-preserving decentralized collaborative ML approaches [45], ranging from horizontal federations, where different devices (and local datasets) share the same feature space, to vertical federations, where devices share training samples that differ in feature space.

Some researchers focused the attention on the decentralized and distributed matrix-factorization approaches [12,16]. However, in this work, we focus on federated learning principles theoretically and practically different from classical distributed approaches. Indeed, FL assumes the presence of a coordinating server and the use of private and self-produced data on each node. In general, distributed approaches do not guarantee these assumptions. Ammad-ud-din *et al.* [3] propose a federated implementation of collaborative filtering, whose security limits are analyzed in [11], which uses the SVD-MF method for implicit feedback [19]. Here, the training is a mixture of Alternating Least Squares (ALS) and Stochastic Gradient Descent (SGD) for preserving users' privacy. Nevertheless, incomprehensibly, almost no work addressed top-N recommendation exploiting the "Learning to rank" paradigm. In this sense, one rare example is the work by Kharitonov *et al.* [22], who recently proposed to combine evolution strategy optimization with a privatization procedure based on differential privacy. The previous work focuses neither on search or recommendation. Perhaps, like ours, it can be classified as a federated learning-to-rank algorithm. Finally, Yang *et al.* [46] identified some recent FL challenges and open research directions.

3 Approach

In this section, we introduce the fundamental concepts regarding the Collaborative Filtering recommendation using a Federated Learning scheme. Along with the problem definition, the notation we adopt is presented.

The recommendation problem over a set of users $\mathcal{U}$ and a set of items $\mathcal{I}$ is defined as the activity of finding for each user $u \in \mathcal{U}$ an item $i \in \mathcal{I}$ that maximizes a utility function $g : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ [36]. Let $\mathbf{X} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$ be the user-item matrix containing for each element x_{ui} an implicit feedback (e.g., purchases, visits, clicks, views, check-ins) of user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$. Therefore, $\mathbf{X}$ only contains binary values, $x_{ui} = 1$ and $x_{ui} = 0$ denoting whether user u has consumed or not item i, respectively.

The recommendation model is based on Factorization approach, originally introduced by Matrix Factorization [27], that became popular in the last decade thanks to its state-of-the-art recommendation accuracy [29]. This technique aims to build a model Θ in which each user u and each item i is represented by the embedding vectors $\mathbf{p}_u$ and $\mathbf{q}_i$, respectively, in the shared latent space $\mathbb{R}^F$. Let assume $\mathbf{X}$ can be factorized such that the dot product between $\mathbf{p}_u$ and $\mathbf{q}_i$ can explain any observed user-item interaction x_{ui}, and any non-observed interaction can be estimated as $\hat{x}_{ui}(\Theta) = b_i(\Theta) + \mathbf{p}_u^T(\Theta) \cdot \mathbf{q}_i(\Theta)$ where b_i is a term denoting the bias of the item i.

Among pair-wise approaches for learning-to-rank the items of a catalog, Bayesian Personalized Ranking [38] is the most broadly adopted, thanks to its capabilities to correctly rank with *acceptable* computational complexity. Given a training set defined by $\mathcal{K} = \{(u, i, j) \mid x_{ui} = 1 \wedge x_{uj} = 0\}$, BPR minimizes the ranking loss by exploiting the criterion $\max_{\Theta} G(\Theta)$, with $G(\Theta) = \sum_{(u,i,j) \in \mathcal{K}} \ln \ \sigma(\hat{x}_{uij}(\Theta)) - \lambda \|\Theta\|^2$, where $\hat{x}_{uij}(\Theta) = \hat{x}_{ui}(\Theta) - \hat{x}_{uj}(\Theta)$ is a real value modeling the relation between user u, item i and item j, $\sigma(\cdot)$ is the sigmoid function, and λ is a model-specific regularization parameter to prevent overfitting. Pair-wise optimization applies to a wide range of recommendation models, including factorization. Hereafter, we denote the model $\Theta = \langle \mathbf{P}, \mathbf{Q}, \mathbf{b} \rangle$, where $\mathbf{P} \in \mathbb{R}^{|\mathcal{U}| \times F}$ is a matrix whose u-th row corresponds to the vector $\mathbf{p}_u$, and $\mathbf{Q} \in \mathbb{R}^{|\mathcal{I}| \times F}$ is a matrix in which the i-th row corresponds to the vector $\mathbf{q}_i$. Finally, $\mathbf{b} \in \mathbb{R}^{|\mathcal{I}|}$ is a vector whose i-th element corresponds to the value b_i.

3.1 FedeRank

FedeRank redesigns the original factorization approach for a federated setting. Indeed, the initial factorization model and its variants use a single, centralized model, which does not guarantee users to control their data. FedeRank splits the pair-wise learning model Θ among a central server S and a federation of users $\mathcal{U}$. Federated learning aims to optimize a global loss function by using data distributed among a federation of users' devices. The rationale is that the server no longer collects private users' data. Rather, it aggregates the results of some steps of local optimizations performed by clients, preserving privacy, ownership, and locality of users' data [6]. Formally, let Θ be the machine learning model parameters, and $G(\Theta)$ be a loss function to minimize. In Federated learning, the users $\mathcal{U}$ of a federation collaborate to minimize G (under the coordination of a central server S) without sharing or exchanging their raw data. From an algorithmic point of view, S shares Θ with the federation of devices. Then, the optimization problem of minimizing G is locally solved. Since each user participates to the federation with her personal data and with her personal client device, we will interchangeably use the terms "client", "user", and "device".

To set up the framework, we consider the central server S holding a model $\Theta_S = \langle \mathbf{Q}, \mathbf{b} \rangle$, where $\mathbf{Q} \in \mathbb{R}^{|\mathcal{I}| \times F}$ is a matrix in which i-th row represents the embedding $\mathbf{q}_i$ for item i in the catalog, while the element b_i of $\mathbf{b} \in \mathbb{R}^{|\mathcal{I}|}$ is the bias of item i. That is, the information on S only characterizes the items of the catalog. On the other hand, each user $u \in \mathcal{U}$ holds a local model $\Theta_u = \langle \mathbf{p}_u \rangle$, where $\mathbf{p}_u \in \mathbb{R}^F$ corresponds to the representation of user u in the latent space of dimensionality F. Each user holds a private interaction dataset $\mathbf{x}_u \in \mathbb{R}^{|\mathcal{I}|}$, which—compared to a centralized recommender system—corresponds to the $\mathbf{X}$'s u-th row. The user u leverages her private dataset $\mathbf{x}_u$ to build the local training set $\mathcal{K}_u = \{(u, i, j) \mid x_{ui} = 1 \wedge x_{uj} = 0\}$. Finally, the overall number of interactions in the system can be obtained by exploiting the local datasets. Let us define it as $X^+ = \sum_{u \in \mathcal{U}} |\{x_{ui} | x_{ui} = 1\}|$.

The training procedure iterates for E *epochs*, in each of which *rpe rounds of communication* between the server and the devices are performed. A round of communication envisages a **Distribution to Devices** → **Federated Optimization** → **Transmission to Server** → **Global Aggregation** sequence. The notation $\{\cdot\}_S^t$ denotes an object computed by the server S at round t, while $\{\cdot\}_u^t$ indicates an object computed by a specific client u at round t.

(1) **Distribution to Devices.** Let $\{\mathcal{U}^-\}_S^t$ be a subset of $\mathcal{U}$ with cardinality m, containing m clients $u \in \mathcal{U}$. The set $\{\mathcal{U}^-\}_S^t$ can be either defined by S, or the result of a request for availability sent by S to clients in $\mathcal{U}$. Each client $u \in \{\mathcal{U}^-\}_S^t$ receives from S the latest snapshot of $\{\Theta_S\}_S^{t-1}$.

(2) **Federated Optimization.** Each user $u \in \{\mathcal{U}^-\}_S^t$ generates the set $\{\mathcal{K}_u^-\}_u^t$ containing T random triples (u, i, j) from $\mathcal{K}_u$. It is worth underlining that Rendle [38] suggests, for a centralized scenario, to train the recommendation model by randomly choosing the training triples from $\mathcal{K}$, to avoid data is traversed item-wise or user-wise, since this may lead to slow convergence. Conversely, in a federated approach, we require to train the model user-wise. Indeed, the learning is separately performed on each device (u), that only knows the data in $\mathcal{K}_u$. Thanks to the user-wise traversing, FedeRank can decide who controls (the designer or the user) the number of triples T in the training set $\{\mathcal{K}_u^-\}_u^t$, to tune

the degree of local computation. With the local training set, the user u can compute her contribution to the overall optimization of Θ_S with the following update:

$$\{\Delta\Theta_S\}_u^t = \{\Delta\langle\mathbf{Q},\mathbf{b}\rangle\}_u^t := \sum_{(u,i,j)\in\{\mathcal{K}_u^-\}_u^t} \frac{\partial}{\partial\Theta_S} \ln\ \sigma(\hat{x}_{uij}(\{\Theta_S\}_S^{t-1};\{\Theta_u\}_u^{t-1})), \tag{1}$$

plus a regularization term. At the same time, the client u updates its local model Θ_u, and incorporates it in the current model by using:

$$\{\Delta\Theta_u\}_u^t = \{\Delta\langle\mathbf{p}_u\rangle\}_u^t := \sum_{(u,i,j)\in\{\mathcal{K}_u^-\}_u^t} \frac{\partial}{\partial\Theta_u} \ln\ \sigma(\hat{x}_{uij}(\{\Theta_S\}_S^{t-1};\{\Theta_u\}_u^{t-1})), \tag{2}$$

plus a regularization term. The partial derivatives in Eq. 1 and 2 are straightforward, and can be easily computed by following the scheme proposed by Rendle *et al.* [38]. At the end of the federated computation, given a shared learning rate α, each client can update its local model Θ_u—containing the user profile—by aggregating the computed update:

$$\{\Theta_u\}_u^t := \{\Theta_u\}_u^{t-1} + \alpha\{\Delta\Theta_u\}_u^t. \tag{3}$$

(3) **Transmission to Server.** In a purely distributed architecture, each user in $\mathcal{U}^-$ returns to S the computed update. Here, instead of sending $\{\Delta\Theta_S\}_u^t$, each user transmits a modified version $\{\Delta\Theta_S^\Phi\}_u^t$. To introduce this aspect of FedeRank, let us define $\mathcal{F} = \{i, \forall(u,i,j) \in \{\mathcal{K}_u^-\}_u^t\}$, and a randomized object $\Phi = \langle\mathbf{Q}^\Phi,\mathbf{b}^\Phi\rangle$, with $\mathbf{Q}^\Phi \in \mathbb{R}^{|\mathcal{I}|\times F}$, and $\mathbf{b}^\Phi \in \mathbb{R}^{|\mathcal{I}|}$. Each row $\mathbf{q}_i^\Phi$ of $\mathbf{Q}^\Phi$ and each element b_i^Φ of $\mathbf{b}^\Phi$ assume their value according to the probabilities:

$$\begin{gathered} P(\mathbf{q}_i^\Phi = \mathbf{1}, b_i^\Phi = 1 \mid i \in \mathcal{F}) = \pi, \quad P(\mathbf{q}_i^\Phi = \mathbf{0}, b_i^\Phi = 0 \mid i \in \mathcal{F}) = 1-\pi, \\ P(\mathbf{q}_i^\Phi = \mathbf{1}, b_i^\Phi = 1 \mid i \notin \mathcal{F}) = 1 \end{gathered} \tag{4}$$

Based on $\{\mathbf{Q}^\Phi\}_u^t$ and $\{\mathbf{b}^\Phi\}_u^t$, $\Delta\Theta_S^\Phi$ can be computed as it follows:

$$\{\Delta\Theta_S^\Phi\}_u^t = \{\Delta\Theta_S\}_u^t \odot \{\Phi\}_u^t := \langle\{\Delta\mathbf{Q}\}_u^t \odot \{\mathbf{Q}^\Phi\}_u^t, \{\Delta\mathbf{b}\}_u^t \odot \{\mathbf{b}^\Phi\}_u^t\rangle, \tag{5}$$

where the operator $\odot$ denotes the Hadamard product. This transformation is motivated by a possible privacy issue. The update $\Delta\mathbf{Q}$ computed in Eq. 1 by user u is a matrix whose rows are non-zero in correspondence of the items i and j of all the triples $(u,i,j) \in \mathcal{K}_u^-$ [38]. An analogous behavior can be observed for the elements of $\Delta\mathbf{b}$. Focusing on the non-zero elements, we observe that, for each triple $(u,i,j) \in \mathcal{K}_u^-$, the updates $\{\Delta\mathbf{q}_i\}_u^t$ and $\{\Delta\mathbf{q}_j\}_u^t$, as well as $\{\Delta b_i\}_u^t$ and $\{\Delta b_j\}_u^t$, show the same absolute value with opposite sign [38]. In fact, this makes completely distinguishable for the server the consumed and the non-consumed items of user u, allowing S to reconstruct $\mathcal{K}_u^-$, thus raising a privacy issue. Since our primary goal is to put users in control of their data, we leave users the possibility to choose a fraction π of positive item updates to send.

The remaining positive item updates (a fraction $1 - \pi$) are masked by setting them to zero, by means of the transformation in Eq. 5. Conversely, the negative updates are always sent to S, since their corresponding rows are always multiplied by a $\mathbf{1}$ vector. Indeed, these updates are related to non-consumed items, which are indistinguishably negative or missing values, assumed to be *non-sensitive* data.

(4) **Global Aggregation.** Once S has received $\{\Delta\Theta_S^{\Phi}\}_u^t$ from all clients $u \in \mathcal{U}^-$, it aggregates the received updates in $\mathbf{Q}$ and $\mathbf{b}$ to build the new global model, with α being the learning rate:

$$\{\Theta_S\}_S^t := \{\Theta_S\}_S^{t-1} + \alpha \sum_{u \in \mathcal{U}^-} \{\Delta\Theta_S^{\Phi}\}_u^t. \qquad (6)$$

Although Federated Learning was conceived as a privacy-by-design paradigm for distributed machine learning, it still does not provide formal privacy guarantees. Malicious actors might acquire different information. They may be able to analyze remote devices or communication flows in the network or infer users' private data by inspecting updates received on the server [21]. Possible solutions include the encryption of data on local devices, the network traffic, or the adoption of secure multi-party computation [7]. Moreover, local differential privacy can guarantee that even if an adversary can inspect the communication between a user and the central server, she can learn only limited information [13,14,43]. To date, FedeRank explicitly provides the needed APIs to work, out of the box, with encryption communication libraries, thus providing state-of-the-art privacy guarantees. We have chosen this solution since discussing privacy and security implications in FL is beyond our scope. In this way, the system designer can choose the privacy solution while disregarding the underlying machine learning model. Moreover, FedeRank can also be easily adapted to guarantee local differential privacy. Indeed, it is not due to chance the choice of putting the user in control of Φ. Suppose the Φ object also considers sending fake information. In that case, FedeRank becomes utterly compliant with the randomized response technique, which guarantees differential privacy [44].

4 Experiments

Datasets. We have investigated the performance of FedeRank considering three well-known datasets: *Amazon Digital Music* [31], *LibraryThing* [48], and *MovieLens 1M* [18]. The former includes the users' satisfaction feedback for a catalog of music tracks available with Amazon Digital Music service. It contains 1,835 users and 41,488 tracks, with 75,932 ratings ranging from 1 to 5. *LibraryThing* collects the users' ratings on a book catalog. It captures the interactions of 7,279 users on 37,232 books. It provides more than two million ratings with 749,401 unique ratings in a range from 1 to 10. The latter is *MovieLens 1M* dataset, which collects users' ratings in the movie domain: it contains 1,000,209 ratings ranging from 1 to 5, 6,040 users, and 3,706 items. We have filtered out users

Table 1. Characteristics of the datasets used in the offline experiments: $|\mathcal{U}|$ is the number of users, $|\mathcal{I}|$ the number of items, X^+ the amount of positive feedback.

Dataset	$\|\mathcal{U}\|$	$\|\mathcal{I}\|$	X^+	$\frac{X^+}{\|\mathcal{U}\|}$	$\frac{X^+}{\|\mathcal{I}\|}$	$\frac{X^+}{\|\mathcal{I}\|\cdot\|\mathcal{U}\|}\%$
Amazon DM	1,835	41,488	75,932	41.38	1.83	0.000997%
LibraryThing	7,279	37,232	749,401	102.95	20.13	0.002765%
MovieLens 1M	6,040	3,706	1,000,209	165.60	269.89	0.044684%

with less than 20 ratings (considering them as cold-users). Table 1 shows the characteristics of the resulting datasets adopted in the experiments.

Baseline Algorithms. We compared FedeRank with representative centralized algorithms to position its performance with respect to the state-of-the-art techniques: **VAE** [30], a non-linear probabilistic model taking advantage of Bayesian inference to estimate the model parameters; **User-kNN** and **Item-kNN** [25], two neighbor-based CF algorithms, that exploit cosine similarity to compute similarity between users or items; **BPR-MF** [38], the centralized vanilla BPR-MF implementation; and **FCF** [3], the only federated recommendation approach, to date, based on MF[1]. We have evaluated FedeRank considering $|\mathcal{U}^-| = 1$. That is, in each round of communication we involve only a single client to avoid noisy results. We thereby guarantee the sequential training, needed to compare against centralized pir-wise techniques. We have investigated with two different FedeRank settings. In the **first setting**, we have set $T = 1$, i.e., each client extracts solely one triple (u, i, j) from its dataset when asked for training the model; with this special condition, we test whether FedeRank is effectively comparable to BPR. Moreover, to make the comparison completely fair, we extract triples as proposed by Rendle *et al.* [38]. The **second setting** follows a real Federated scenario where the client local computation is not limited to a single triple. Specifically, the number T of triples extracted by each client is set to $\frac{X^+}{|\mathcal{U}|}$.

Reproducibility and Evaluation Metrics. To train FedeRank, we have adopted a realistic temporal hold-out 80–20 splitting for training set and test set [17]. We have further split the training set adopting a temporal hold-out strategy on a user basis to pick the last 20% of interactions as a validation set. Hence, we have explored a grid in the range $\{0.005, \ldots, 0.5\}$. Then, to ensure a fair comparison, we have used the same learning rate to train FedeRank. We have set up the remaining parameters as follows: the user- and positive item-regularization parameter is set to 1/20 of the learning rate; conversely, the negative item-regularization parameter is set to 1/200 of the learning rate as suggested by Anelli *et al.* [4]. Moreover, for each setting, we have selected the best model in the first 20 epochs. Finally, the number of latent factors is equal to 20. This value reflects a trade-off between latent factors' expressiveness and memory space limits (given by a realistic Federated Learning environment). We have measured

[1] Since no source code is available, we reimplemented it in the reader's interest.

the recommendation accuracy by exploiting: Precision ($P@N$) (the proportion of relevant items in the recommendation list), and Recall ($R@N$), that measures the relevant suggested items. Regarding diversity, we have adopted Item Coverage (IC) and Gini Index (G). The former provides the overall number of diverse recommended items, and it highlights the degree of personalization expressed by the model [1]. The latter measures how unequally an RS provides users with different items [10], being higher values corresponding to more tailored lists.

4.1 Performance of Federated Learning to Rank

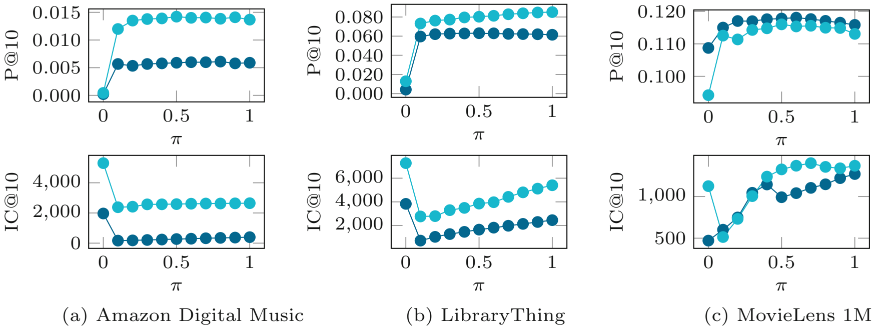
We begin our experimental evaluation by investigating the efficacy of FedeRank, and we assess whether its performance is comparable to baseline algorithms. Table 2 depicts the results in terms of accuracy and diversity. The Table is visually split into two parts. The algorithms in the bottom part (BPR-MF, FCF, and the two settings of FedeRank) are the factorization-based models. The upper part provides the positioning of FedeRank to the other state-of-the-art approaches. Focusing on the factorization-based methods, we can note that BPR-MF outperforms FedeRank for $T = 1$, but it remains at about 67% and 88% of the centralized algorithm for *Amazon Digital Music* and *LibraryThing*, respectively. However, the realistic Federated setting is with $T = X^+/|\mathcal{U}|$. Here, FedeRank consistently improves the recommendation performance with respect to BPR-MF and FCF, over the three datasets. Actually, for *Amazon Digital Music* and *LibraryThing* FedeRank improves accuracy metrics of about 50% and 25% with respect to BPR-MF. The achievement can be explained as an advantage brought by the increased local computation. It is worth noticing that these results partially contradict Rendle *et al.* [38] since they hypothesize that traversing user-wise the training triples would worsen the recommendation performance. The same accuracy improvements are not visible in *MovieLens 1M*, where we witness results comparable or worse than BPR-MF, probably due to the overfitting caused by the very high ratio between ratings and items. FedeRank with increased computation still results robust with respect to the IC metric, since, in general, it outperforms or remains comparable to FCF and BPR-MF.

Table 2. Recommendation performance for baselines and FedeRank on the three datasets. For each value of T, the experiment with the best π is shown.

	Amazon Digital Music				LibraryThing				MovieLens 1M			
	P@10	R@10	IC@10	G@10	P@10	R@10	IC@10	G@10	P@10	R@10	IC@10	G@10
Random	0.00005	0.00005	14186	0.28069	0.00054	0.00028	31918	0.60964	0.00871	0.00283	3666	0.85426
Most popular	0.00469	0.00603	24	0.00023	0.05013	0.03044	36	0.00031	0.10224	0.03924	118	0.00569
User-kNN	0.01940	0.02757	4809	0.04115	0.14193	0.10115	3833	0.01485	0.12613	0.06701	737	0.04636
Item-kNN	0.02147	0.03171	4516	0.03801	0.20214	0.14778	12737	0.09979	0.08873	0.05475	2134	0.19292
VAE	0.01580	0.02289	3919	0.04179	0.10834	0.07711	7800	0.04638	0.11735	0.06192	1476	0.09259
BPR-MF	0.00921	0.01298	739	0.00415	0.07009	0.04303	3082	0.01359	**0.11911**	0.05817	**1444**	**0.08508**
FCF	0.00839	0.01222	**2655**	0.01861	**0.10760**	0.04392	829	0.01305	0.10760	0.04392	829	0.01305
FedeRank												
$T = 1$	0.00610	0.00889	349	0.00136	0.06309	0.03738	1650	0.00512	0.11805	**0.05902**	1041	0.06608
$T = X^+/\|\mathcal{U}\|$	**0.01422**	**0.02060**	2586	**0.02153**	0.08512	**0.05627**	**5404**	**0.02784**	0.11599	0.05571	1326	0.02513

4.2 Analysis of Positive Feedback Transmission Ratio

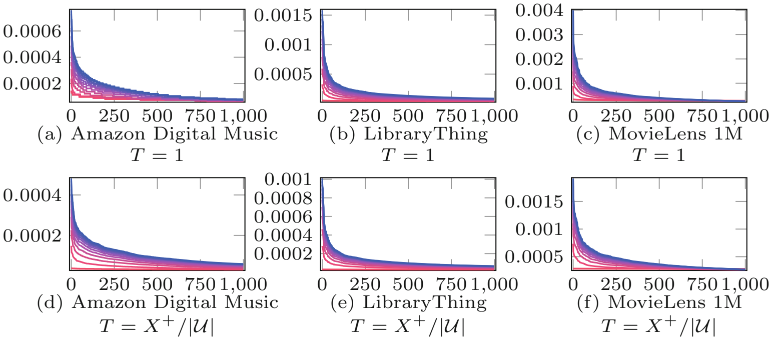
We have extensively analyzed the behavior of FedeRank when tuning π for sending progressive fractions of positive feedback in $[0.0, \ldots, 1.0]$ with step 0.1. We believe that the most important dimensions for this analysis are accuracy (Precision), and aggregate diversity (Item Coverage). Figure 1 reports the results for the two experimented settings. Even here, *Amazon Digital Music* and *LibraryThing* show similar trends. The accuracy of the recommendation progressively increases reaching the maximum with fractions 0.8 and 0.5, respectively, for $T = 1$, and with fractions 0.9 and 1.0 for $T = X^+/|\mathcal{U}|$. First, this suggests that, at the beginning of the training, some positive feedback is needed for establishing the value of an item. Notwithstanding, even with $\pi = 0.1$ (i.e., sharing just 10% of private information), we witness a jump in recommendation accuracy (one order of magnitude), reaching up to 92% of the best accuracy. We should also observe another significant behavior. With a fraction of 0.0, we observe a high value of *IC*, with poor recommendation accuracy. It suggests that the system could not capture population preferences, and it behaves similarly to Random. However, even with a small fraction of positive feedback like 0.1, we observe a significant decrease in diversity metrics. The system learns which items are popular and starts suggesting them. Moreover, if we observe large fractions, we may notice that diversity increases as we feed the system with more information. For *MovieLens 1M*, it is worth noticing that FedeRank shows accuracy performance extremely close to the best value by sharing only 10% of positive interactions. This behavior may be due to several reasons. Firstly, *MovieLens 1M* is a relatively dense dataset in the recommendation scenario (it has a sparsity of 0.955). Secondly, it shows a very high user-item ratio [2] (i.e., 1.63) compared to *Amazon Digital Music* (0.04), and *LibraryThing* (0.20), and it shows high values for the average number of ratings per user (132.87), and ratings per item $(216, 56)$. All these clues suggest that the system learns how to rank items even without the need for the totality of ratings. If we focus on diversity metrics, *IC* and Gini, we may notice that diversity is progressively increasing from fraction 0.1 to 1.0.



(a) Amazon Digital Music (b) LibraryThing (c) MovieLens 1M

Fig. 1. F1 performance at different values of π in the range $[0.1, 1]$. Dark blue is $T = 1$, light blue is $T = X^+/|\mathcal{U}|$. (Color figure online)

It suggests that the system recommends a small number of popular items with a fraction of 0.1, while it provides more diversified recommendation lists considering larger portions of positive user feedback. At this stage of the analysis, we can draw an interesting consideration: in general, the highest accuracy values do not correspond to the fraction of 1.0. Specifically, the experiments show that, initially, the recommender struggles to suggest relevant items without positive feedback (fraction 0.0). However, with a small injection of feedback, the system starts to work well. Nonetheless, in *Amazon Digital Music* and *LibraryThing*, if we increase the fraction, we witness an increase concerning accuracy only until a certain value of π. Although this consideration, we observe an increase in diversity metrics when we continue to increase the value of π. Since it has a small or even detrimental impact on accuracy, those items might be unpopular items erroneously suggested to users.



(a) Amazon Digital Music $T = 1$

(b) LibraryThing $T = 1$

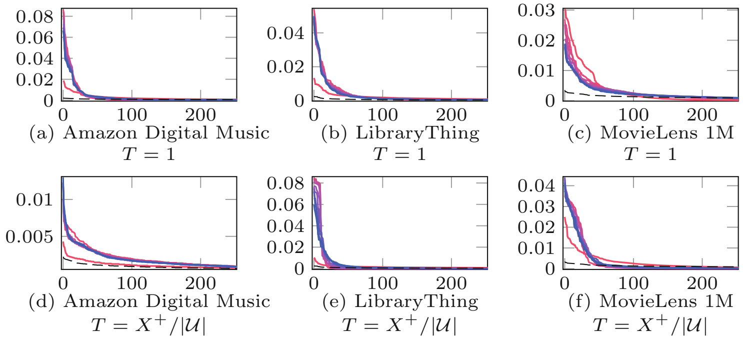
(c) MovieLens 1M $T = 1$

(d) Amazon Digital Music $T = X^+/|\mathcal{U}|$

(e) LibraryThing $T = X^+/|\mathcal{U}|$

(f) MovieLens 1M $T = X^+/|\mathcal{U}|$

Fig. 2. Normalized number of item updates during the training: the 1,000 most updated items for different values of π (from $\pi = 0.0$ in red to $\pi = 1.0$ in blue). (Color figure online)



(a) Amazon Digital Music $T = 1$

(b) LibraryThing $T = 1$

(c) MovieLens 1M $T = 1$

(d) Amazon Digital Music $T = X^+/|\mathcal{U}|$

(e) LibraryThing $T = X^+/|\mathcal{U}|$

(f) MovieLens 1M $T = X^+/|\mathcal{U}|$

Fig. 3. Normalized number of recommendations for each item (colored curves from $\pi = 0.0$ in red to $\pi = 1.0$ in blue) vs. normalized amount of positive feedback per item (black dashed curve). The 250 most popular items are shown. (Color figure online)

4.3 Study on FedeRank Algorithmic Bias

In this section, we study how incomplete transmission of user feedback affects the item popularity in the recommendations and during the learning process. It is essential to discover whether the exploitation of a FL approach influences the algorithmic bias, determining popular items to be over-represented [5,9]. We have re-trained FedeRank with all the previously considered π. For each experiment, we analyzed the data flow between the clients and the server. Afterward, we have extracted the number of updates for each item. Figure 2 illustrates the occurrences for the $1,000$ most updated items. In the Figure, the curve colors denote the different π, while the values represent the update frequency during the training process for each item on the horizontal axis. Analogously, we considered the final top-10 recommendation list of each user. Following the same strategy, we analyzed the occurrences of the items in the recommendation. Then, we ordered items from the most to the least recommended, and we plotted the occurrences of the first 250 in Fig. 3. To compare the different datasets, we have normalized the values considering the overall dataset occurrences. Figure 2 shows that data disclosure, i.e., the value of π, highly influences the information exchanged during the training process. Additionally, the update frequency curve exhibits a constant behavior for all the datasets, when $\pi = 0.0$. This trend suggests that items are randomly updated without taking into account any information about item popularity. This behavior explains the high *IC* entirely observed in Fig. 1 for $\pi = 0.0$. The curve for $\pi = 0.1$ shows that the exchanged data is enough to provide the system with information about item popularity. The curves suggest that the information on item popularity is being injected into the system. By increasing the value of π, the trend becomes more evident. Due to the original rating distribution, the system initially exchanges more information about the very popular items. To analyze the algorithmic bias, we can observe Fig. 3, where the colored curves represent the frequency of item recommendation on the horizontal axis, and the black dashed curve the amount of positive feedback for that item in the dataset. Remarkably, item popularity in recommendation lists does not vary as we may expect based on the previous analysis. The setting $\pi = 0.0$ is an exception, as extensively explained before. Since in *Amazon Digital Music* and *LibraryThing* the updates sent by the clients are randomly selected among the negative items, FedeRank acts like a Random recommender. The system cannot catch popularity information and it struggles to make the right items popular. Finally, we can focus on the curves for $\pi > 0$. It is noteworthy that the π curves behave similarly, and they propose the same proportion of popular items. The curves show the model absorbs the initial variation in exchanged item distribution, unveiling an unknown aspect of factorization models.

5 Conclusion and Future Work

In this paper, we have tackled the problem of putting users in control of their private data for a recommendation scenario. Witnessing the growing concern about privacy, users might want to exploit their sensitive data and share only a small fraction of it. In such a context, classic CF approaches are no more feasible. To overcome these problems, we have proposed FedeRank, a novel recommendation framework that respects the FL paradigm. With FedeRank, private user feedback remains on user devices unless they decide to share it. Nevertheless, FedeRank ensures high-quality recommendations despite the constrained setting. We have extensively studied the performance of FedeRank by comparing it with other state-of-the-art methods. We have then analyzed the impact of progressive reduction of user feedback and studied the effects on the diversity of the recommendation results. Finally, we have investigated whether the federated algorithm imposes an algorithmic bias to the recommended lists. The study paves the way for further research directions. On the one hand, the results' analysis suggests that centralized recommender systems are not performing at their best. Feeding recommender systems with all the available feedback, without any filtering, may lead to a performance worsening. On the other hand, the competitive results of FedeRank suggest that the FL-based algorithms show a recommendation quality that makes them suitable to be adopted on a massive scale.

References

1. Adomavicius, G., Kwon, Y.: Improving aggregate recommendation diversity using ranking-based techniques. IEEE Trans. Knowl. Data Eng. **24**(5), 896–911 (2012)
2. Adomavicius, G., Zhang, J.: Impact of data characteristics on recommender systems performance. ACM Trans. Manage. Inf. Syst. **3**(1), 31–317 (2012)
3. Ammad-ud-din, M., et al.: Federated collaborative filtering for privacy-preserving personalized recommendation system, CoRR, abs/1901.09888 (2019)
4. Anelli, V.W., Noia, T.D., Sciascio, E.D., Pomo, C., Ragone, A.: On the discriminative power of hyper-parameters in cross-validation and how to choose them. In: Proceedings of the 13th ACM Conference on Recommender Systems, pp. 447–451. ACM (2019)
5. Baeza-Yates, R.: Bias in search and recommender systems. In: 14th ACM Conference on Recommender Systems, RecSys 2020, Virtual Event, Brazil, 22–26 September 2020, p. 2 (2020)
6. Bonawitz, K., et al.: Towards federated learning at scale: system design, CoRR, abs/1902.01046 (2019)
7. Bonawitz, K., et al.: Practical secure aggregation for privacy-preserving machine learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, 30 October–03 November 2017, pp. 1175–1191 (2017)
8. California State Legislature: The California Consumer Privacy Act of 2018 (2018)
9. Cañamares, R., Castells, P.: Should I follow the crowd?: a probabilistic analysis of the effectiveness of popularity in recommender systems. In: 2018 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08–12, pp. 415–424 (2018)
10. Castells, P., Hurley, N.J., Vargas, S.: Novelty and diversity in recommender systems. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 881–918. Springer, Boston (2015)
11. Chai, D., Wang, L., Chen, K., Yang, Q.: Secure federated matrix factorization. IEEE Intell. Syst., 1 (2020)

12. Duriakova, E., et al.: PDMFRec: a decentralised matrix factorisation with tunable user-centric privacy. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, 16–20 September 2019, pp. 457–461 (2019)
13. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci. **9**(3–4), 211–407 (2014)
14. Erlingsson, Ú., Pihur, V., Korolova, A.: RAPPOR: randomized aggregatable privacy-preserving ordinal response. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014, pp. 1054–1067 (2014)
15. European Commission. 2018 reform of EU data protection rules (2018)
16. Fierimonte, R., Scardapane, S., Uncini, A., Panella, M.: Fully decentralized semi-supervised learning via privacy-preserving matrix completion. IEEE Trans. Neural Netw. Learn. Syst. **28**(11), 2699–2711 (2017)
17. Gunawardana, A., Shani, G.: Evaluating recommender systems. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 265–308. Springer, Boston (2015)
18. Harper, F.M., Konstan, J.A.: The MovieLens datasets: history and context. ACM Trans. Interact. Intell. Syst. (TIIS) **5**(4), 1–19 (2015)
19. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Proceedings of the 8th IEEE International Conference on Data Mining, ICDM 2008, 15–19 December 2008, Pisa, Italy, pp. 263–272. IEEE Computer Society (2008)
20. Jeckmans, A.J.P., Beye, M., Erkin, Z., Hartel, P.H., Lagendijk, R.L., Tang, Q.: Privacy in recommender systems. In: Ramzan, N., van Zwol, R., Lee, J., Clüver, K., Hua, X. (eds.) Social Media Retrieval, Computer Communications and Networks, pp. 263–281. Springer, London (2013)
21. Kairouz, P.: Advances and Open Problems in Federated Learning (2019)
22. Kharitonov, E.: Federated online learning to rank with evolution strategies. In: Proceedings of the 12th ACM International Conference on Web Search and Data Mining, pp. 249–257 (2019)
23. Konecný, J., McMahan, H.B., Ramage, D., Richtárik, P.: Federated optimization: distributed machine learning for on-device intelligence, CoRR, abs/1610.02527 (2016)
24. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, 24–27 August 2008, pp. 426–434 (2008)
25. Koren, Y.: Factor in the neighbors: scalable and accurate collaborative filtering. ACM Trans. Knowl. Discov. Data (TKDD) **4**(1), 1–24 (2010)
26. Koren, Y., Bell, R.M.: Advances in collaborative filtering. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 77–118. Springer, Heidelberg (2015)
27. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Comput. **42**(8), 30–37 (2009)
28. Koren, Y., Sill, J.: OrdRec: an ordinal model for predicting personalized item rating distributions. In: Mobasher, B., Burke, R.D., Jannach, D., Adomavicius, G. (eds.) Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, 23–27 October 2011, pp. 117–124. ACM (2011)
29. Kumar Bokde, D., Girase, S., Mukhopadhyay, D.: Role of matrix factorization model in collaborative filtering algorithm: a survey, CoRR, abs/1503.07475 (2015)

30. Liang, D., Krishnan, R.G., Hoffman, M.D., Jebara, T.: Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 World Wide Web Conference, pp. 689–698 (2018)
31. McAuley, J., Targett, C., Shi, Q., Van Den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 43–52 (2015)
32. McFee, B., Barrington, L., Lanckriet, G.R.G.: Learning content similarity for music recommendation. IEEE Trans. Audio Speech Lang. Process. **20**(8), 2207–2218 (2012)
33. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., et al.: Communication-efficient learning of deep networks from decentralized data. arXiv preprint arXiv:1602.05629 (2016)
34. McNee, S.M., Riedl, J., Konstan, J.A.: Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: CHI'06 Extended Abstracts on Human Factors in Computing Systems, pp. 1097–1101 (2006)
35. Muhammad, K., et al.: FedFast: going beyond average for faster training of federated recommender systems. In: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2020, Virtual Event, CA, USA, 23–27 August 2020, pp. 1234–1242 (2020)
36. Ning, X., Desrosiers, C., Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 37–76. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6_2
37. Rendle, S.: Factorization machines. In: The 10th IEEE International Conference on Data Mining, ICDM 2010, Sydney, Australia, 14–17 December 2010, pp. 995–1000 (2010)
38. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: Bilmes, J.A., Ng, A.Y. (eds.) Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009, Montreal, QC, Canada, 18–21 June 2009, pp. 452–461. AUAI Press (2009)
39. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, 26–30 April 2010, pp. 811–820 (2010)
40. Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the 3rd International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, 4–6 February 2010, pp. 81–90 (2010)
41. Shi, Y., Larson, M., Hanjalic, A.: List-wise learning to rank with matrix factorization for collaborative filtering. In: Proceedings of the 4th ACM Conference on Recommender Systems, pp. 269–272 (2010)
42. Standing Committee of the National People's Congress of Popular Republic of China. China internet security law (2017)
43. Differential Privacy Team: Learning with Privacy at Scale (2017). https://machinelearning.apple.com/research/learning-with-privacy-at-scale. Accessed Jan 2021

44. Wang, Y., Wu, X., Hu, D.: Using randomized response for differential privacy preserving data collection. In: Palpanas, T., Stefanidis, K. (eds.) Proceedings of the Workshops of the EDBT/ICDT 2016 Joint Conference, EDBT/ICDT Workshops 2016, Bordeaux, France, 15 March 2016, vol. 1558 of CEUR Workshop Proceedings. CEUR-WS.org (2016)
45. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM TIST **10**(2), 12:1–12:19 (2019)
46. Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., Yu, H.: Federated Learning. Morgan & Claypool Publishers, San Rafael (2019)
47. Yuan, J., Shalaby, W., Korayem, M., Lin, D., AlJadda, K., Luo, J.: Solving cold-start problem in large-scale recommendation engines: a deep learning approach. In: 2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, 5–8 December 2016, pp. 1901–1910. IEEE Computer Society (2016)
48. Zhao, T., McAuley, J., King, I.: Improving latent factor models via personalized feature projection for one class recommendation. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management, pp. 821–830 (2015)

Active Learning for Entity Alignment

Max Berrendorf[1](✉), Evgeniy Faerman[1], and Volker Tresp[1,2]

[1] Ludwig-Maximilians-Universität München, Munich, Germany
{berrendorf,faerman}@dbs.ifi.lmu.de
[2] Siemens AG, Munich, Germany
volker.tresp@siemens.com

Abstract. In this work, we propose a novel framework for labeling entity alignments in knowledge graph datasets. Different strategies to select informative instances for the human labeler build the core of our framework. We illustrate how the labeling of entity alignments is different from assigning class labels to single instances and how these differences affect the labeling efficiency. Based on these considerations, we propose and evaluate different active and passive learning strategies. One of our main findings is that passive learning approaches, which can be efficiently precomputed, and deployed more easily, achieve performance comparable to the active learning strategies. In the spirit of reproducible research, we make our code available at https://github.com/mberr/ea_active_learning.

Keywords: Entity alignment · Active learning · Knowledge graphs

1 Introduction

A knowledge graph (KG) is a way to store information (semi-)structurally to enable automatic data processing and data interpretation. KGs are utilized in various Information Retrieval related applications requiring semantic search of information [1,11]. While there exist various large open-source KGs, such as YAGO-3 [25], Wikidata [38], or ConceptNet [33], they often contain orthogonal information, and have their respective strength and weaknesses. Hence, being able to combine information from different knowledge graphs is required in many applications. An important subtask is identifying matching entities across several graphs, called *entity alignment* (EA). Recent years witnessed substantial advances regarding the methodology, in particular involving graph neural networks (GNNs) [6,7,19,28,34–37,40,42,44,46]. Common among these approaches is that they use a set of given seed alignments and infer the remaining ones. While several benchmark datasets are equipped with alignments, acquiring them in practice is cumbersome and expensive, often requiring human annotators. To address this problem, we propose to use *active learning* for entity alignment. In summary, our contributions are as follows:

M. Berrendorf and E. Faerman—equal contribution.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 48–62, 2021.
https://doi.org/10.1007/978-3-030-72113-8_4

- To the best of our knowledge, we are the first to propose using active learning for entity alignment in knowledge graphs. We investigate and formalize the problem, identify critical aspects, and highlight differences to the classical active learning setting for classification.
- A specialty of entity alignment is that learning is focused on information about aligned nodes. We show how to additionally utilize information about exclusive nodes in an active learning setting, which leads to significant improvements.
- We propose several different heuristics, based upon node centrality, graph and embedding coverage, Bayesian model uncertainty, and certainty matching.
- We thoroughly evaluate and discuss the heuristics' empirical performance on a well-established benchmark dataset using a recent GNN-based model. Thereby, we show that state-of-the-art heuristics for classification tasks perform poorly compared to surprisingly simple node centrality based approaches.

2 Problem Setting

We study the problem of entity alignment for knowledge graphs (EA). A knowledge graph can be represented by the triple $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ is a set of entities, $\mathcal{R}$ a set of relations, and $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ a set of triples. The alignment problem now considers two such graphs $\mathcal{G}^L, \mathcal{G}^R$ and seeks to identify entities common to both, together with their mapping. The mapping can be defined by the set of matching entity pairs $\mathcal{A} = \{(e, e') \mid e \in \mathcal{E}^L, e' \in \mathcal{E}^R, e \equiv e'\}$, where $\equiv$ denotes the matching relation. While some works are using additional information such as attributes or entity labels, we solely consider the graph structure's relational information. Thus, a subset of alignments $\mathcal{A}_{train} \subseteq \mathcal{A}$ is provided, and the task is to infer the remaining alignments $\mathcal{A}_{test} := \mathcal{A} \setminus \mathcal{A}_{train}$. With $\mathcal{A}^L := \{e \in \mathcal{E}^L \mid \exists e' \in \mathcal{E}^R : (e, e') \in \mathcal{A}\}$ we denote the set of entities from $\mathcal{G}^L$ which do have a match in $\mathcal{A}$, and $\mathcal{A}^R$ analogously. With $\mathcal{X}^L = \mathcal{E}^{\mathcal{L}} \setminus \mathcal{A}^L$ we denote the set of exclusive entities in the graph $\mathcal{G}^L$ which occur neither in train nor test alignment, and $\mathcal{X}^R$ analogously.

In practice, obtaining high-quality training alignments means employing a human annotator. As knowledge graphs can become large, annotating a sufficient number of alignment pairs may require significant labeling efforts and might be costly. Thus, we study strategies to select the most informative alignment labels to achieve higher performance with fewer labels, commonly referred to as active learning. The following section surveys existing literature about active learning with a particular focus on graphs and reveals differences in our setting.

3 Related Work

Classical active learning approaches [31] often do not perform well in batch settings with neural network architectures. Therefore, developing active learning heuristics for neural networks is an active research area. New approaches were proposed for image [2,16,18,30,39,43], text [32,45] and relational [5,17,23,27,41] data. Active learning algorithms aim to select the most informative training instances. For instance, the intuition behind uncertainty sampling [22] is that instances about which the model is unconfident comprise new or not yet explored information. However, the estimation of neural networks' uncertainty is not a trivial task since neural networks are often overconfident about their predictions [15]. One approach to tackle this problem is to use Monte-Carlo dropout to estimate the uncertainty for active learning heuristics [16,27,32]. Alternatively, [2] demonstrated that ensembles of different models lead to better uncertainty estimation and consequently better instance selection. The method described in [23] adopts a different approach and queries labels for instances for which it is the most certain that they are unlabeled. For this assessment, the authors propose an adversarial framework, where the discriminator differentiates between labeled and unlabeled data.

Geometric or density-based approaches [5,17,18,30,41,43], on the other hand, aim to select the most representative instances. Therefore, unlabeled instances are selected for labeling, such that labeled instances cover unlabeled data in the embedding space. Other approaches to estimate the informativeness of unlabeled samples use, e.g., the expected length of gradient [45].

Active learning approaches with neural networks on relational data were so far applied to the classification of nodes in homogeneous graphs [5,17,23,41] and link prediction in knowledge graphs [27]. In [8,9,26] authors propose active learning approaches for the graph matching problem, where the matching costs are known *in advance*, and the goal is to minimize assignment costs. Note that this is different from our task, where the goal is to learn meaningful representations of the entities.

4 Methodology

In this section, we introduce our proposed labeling setting and describe data post-processing to leverage exclusive nodes. Moreover, we propose numerous new labeling strategies: Some strategies take inspiration from existing state-of-the-art heuristics for classification. Others are developed entirely new based on our intuitions. Finally, we present our evaluation framework for the evaluation of different heuristics.

4.1 Labeling Setting

Since we are dealing with matching KGs, where entities have meaningful labels, we assume that human annotators use these entity names for matching. Therefore, we see two different possibilities to formulate the labeling task:

1. The system presents annotators with possible matching pairs, and they label it as `True` or `False`
2. The system presents annotators a node from one of the two KGs, and the task is to find all matching nodes in the other KG.

It is easier to label a single instance in the first scenario, as it is a yes/no question. However, since each node can have more than one matching node in the other KG, $|\mathcal{E}^L| \times |\mathcal{E}^R|$ queries are necessary to label the whole dataset. In contrast, in the second scenario, human annotators need a similar qualification but the time spent per labeled instance increases because they have to search for possible matchings. However, there are the following advantages of the second scenario:

First, there are only $|\mathcal{E}^L| + |\mathcal{E}^R|$ possible queries. Second, in both scenarios, the learning algorithm needs positive matchings to start training. Assuming $|\mathcal{A}^L| \approx |\mathcal{A}^R| \approx |\mathcal{A}|$ and $|\mathcal{E}^R| \approx |\mathcal{E}^L| \approx |\mathcal{E}|$, the probability to select a match with a random query is in the first scenario $|\mathcal{A}|/|\mathcal{E}|^2$, whereas for the second scenario it is $|\mathcal{A}|/|\mathcal{E}|$. Additionally, in the second scenario, it is possible to start with some simple graph-based heuristics, e.g., based on a graph centrality score like degree or betweenness. For many KGs, it is a valid assumption that the probability of having a match is higher for more central nodes. Cold-start labeling performance is especially relevant when the labeling budget is restricted. Third, in the classical active learning scenario, there is the assumption that each query returns a valid label. However, for EA, the information that two nodes do not match is limited since negative examples can also be obtained by negative sampling. In contrast, in the second scenario, we can use information about missing matchings to adapt the dataset, see Sect. 4.2.

In this paper, we focus on the second scenario. However, heuristics relying on information from the matching model described in Sect. 4.3 can also be applied in the first scenario.

4.2 Dataset Adjustment

The EA task's main motivation is either the fusion of knowledge into a single database or exchanging information between different databases. In both cases, the primary assumption is that there is information in one KG, which is not available in the other. This information comes in relations between aligned entities, relations with exclusive entities, or relations between exclusive entities. While larger differences between the KGs increase their fusion value, they also increase the difficulty of matching processes. One possibility to partially mitigate this problem is to enrich both KGs independently using link prediction and transfer links between aligned entities in the training set [6,23]. As this methodology does only deal with missing relations between shared entities, in this work, we go a step further: Since we control the labeling process, we naturally learn about exclusive nodes from the annotators. Therefore, we propose to remove the exclusive nodes from the KGs for the matching step. After the matching is finished, the exclusive nodes can be re-introduced. In the classical EA setting, where the KGs and partial alignments are already given, and there is no control over dataset creation, the analogous removal of exclusive nodes is not possible: To determine whether a node is exclusive or just not contained in the training alignment requires access to the test alignments, hence representing a form of test leakage.

4.3 Active Learning Heuristics

The main goal of active learning approaches is to select the most informative set of examples. In our setting, each query either results in matches or verified exclusiveness, both providing new information. Nodes with an aligned node in the other KG contribute to the signal for the supervised training. State-of-the-art GNN models for EA learn by aggregating the k-hop neighborhood of a node. Two matching nodes in training become similar when their aggregated neighborhood is similar. Therefore, the centrality of identified alignments or their coverage is vital for the performance. On the other hand, exclusive nodes improve training by making both KGs more similar. Since it is not clear from the outset, what affects the final performance most, we analyze heuristics with different inductive biases.

Node Centrality – Selecting nodes with high centrality in the graph has the following effects: (a) a higher probability of a match in the opposite graph, and (b) updates for a larger number of neighbors if a match or significant graph changes when being exclusive. Although there is a large variety of different centrality measures in graphs [10], we observed in initial experiments that they perform similarly. Therefore, in this work, we evaluate two heuristics based on the nodes' role in the graph. The first, *degree* heuristic (denoted as *deg*), orders nodes by their degree, and the nodes with a higher degree are selected first. The second, *betweenness* heuristic (*betw*), works similarly and relies on the betweenness centrality measure.

Graph Coverage – Real-World graphs tend to have densely connected components [12]. In this case, if nodes for labeling are selected according to some centrality measure, there may be a significant overlap of neighborhoods. At the same time, large portions of the graph do receive no or infrequent updates. Therefore, we propose a heuristic, seeking to distribute labels across the graph. We adopt an *approximate vertex cover* algorithm [29] to define an active learning heuristic for entity alignment. Each node is initialized with a weight equal to its degree. Subsequently, we select the node from both graphs with the largest weight, remove it from the candidate list, and decrease all its neighbors' weight by one. We denote this heuristic as *avc*.

Embedding Space Coverage – The goal of embedding space coverage approaches is to cover the parts of the embedding space containing data as well as possible. Here we adapt the state-of-the art method *coreset* [30] (denoted as *cs*) for the EA task. Thereby, we aim to represent each graph's embedding space by nodes with *positive* matchings. We adopt a greedy approach from [30], which in each step selects the object with the largest distance to the nearest neighbor among already chosen items. Its performance was similar to the mixed-integer program algorithm while being significantly faster. In the process of node selection, it is not known whether nodes in the same batch have matchings or are exclusive. Thereby, in each step, each candidate node is associated with a score according to its distance to the nearest positive matching or the nodes already selected as potential positives in the same batch. The node with the largest distance to the closest positive point is added to the batch.

Embedding Space Coverage by Central Nodes – The possible disadvantage of *coreset* heuristic in the context of entity alignment is that selected nodes may have low centrality and therefore affect only a small portion of the graph. Intuitively, it is possible because each next candidate is maximally distant from all nodes with positive matchings, which are expected to be more or less central. In this heuristic, we try to remedy this effect and sample nodes with high centrality in different parts of embedding space. Therefore, in each step, we perform clustering of node representations from both graphs in the joint space, c.f. Fig. 1. We count already labeled nodes in each cluster and determine the number of candidates selected from this specific cluster. This number is inversely proportional to the number of already labeled nodes in the cluster. We then use a node centrality based heuristic to select the chosen number of candidates per cluster. We denote this heuristic by *esccn*.

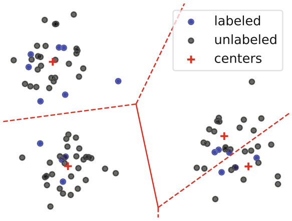


Fig. 1. Schematic visualization of the *esccn* heuristic. The labeled nodes per cluster are counted and used to derive how many samples to draw from this cluster. Another heuristic is then used to select the specific number from the given clusters, e.g., a graph-based *degree* heuristic.

Uncertainty Matching – Uncertainty-based approaches are motivated by the idea that the most informative nodes are those for which the model is most uncertain about the final prediction. We reformulate EA as a classification problem: The number of classes corresponds to the number of matching candidates, and we normalize the vector of similarities to the matching candidates with the softmax operation. A typical uncertainty metric for classification is *Shannon entropy* computed over the class probability distribution, where large entropy corresponds to high uncertainty. We can employ *Monte-Carlo Dropout* to compute a Bayesian approximation of the softmax for the entropy similarly to [17]. However, the repeatable high entropy across multiple dropout masks indicates the *prediction uncertainty*, where the model is *certain* that a right prediction is impossible. In the context of entity alignment, we expect high prediction uncertainty for the exclusive nodes since a model may be *certain* about lacking good matchings. Therefore we opt for model uncertainty for the entity alignment. The model uncertainty is high if the model makes different (certain) decisions for the same instances in multiple runs [14]. We employ *BALD* [21] with Monte-Carlo Dropout [17]. The heuristic computes the expected difference between the entropy of single model prediction and expected entropy. Note that numerous classes may lead to similar entropy and BALD values for the whole dataset. To mitigate this effect, we employ softmax temperature [20].

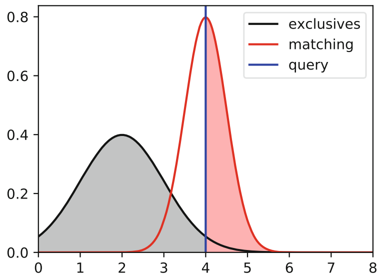


Fig. 2. Visualization of scoring method of the *prexp* heuristic. We fit two normal distributions for matching and exclusive nodes. Each distribution models the maximum similarity these nodes have to any node in the other graph ($s_{max}(q)$). To assess the quality of a query q, we get its maximum similarity, and evaluate $P_{match}(s_{max}(e) \leq s_{max}(q)) - P_{excl}(s_{max}(e) \geq s_{max}(q))$, i.e. the black area minus the red one. (Color figure online)

Certainty Matching – A distinctive property of EA is that the supervised learning signal is provided only by the part of the labeled nodes that have a matching partner in the other graph. Therefore, we propose a heuristic that prefers nodes having matches in the opposite graph, named *previous-experience-based (prexp)*. As the model is trained to have high similarities between matching nodes, the node with maximum similarity is the most likely matching partner for a given node. Moreover, we expect that higher similarity values indicate a better match, such that we can utilise this maximum similarity as a matching score: $s_{max}(e) = \max_{e' \in \mathcal{E}^R} similarity(e, e')$ for $e \in \mathcal{E}^L$. Thus, we hypothesize that the distribution of maximum similarity values between exclusive nodes and those having a matching partner differ and can be used to distinguish those categories. However, we note that the similarity distribution for already labeled nodes may differ from those that are not labeled, as the labeled nodes directly receive updates by a supervised loss signal. Hence, we use *historical* similarity values acquired when we selected unlabeled nodes for labeling, and the ground truth information about them having a match received after the labeling. Based on these, we fit two normal distributions for maximum similarities: The first distribution with the probability function P_{match} describes the distribution of maximal similarity score of nodes with matchings. Similarly, the function P_{excl} computes the probability that the maximal similarity score belongs to an exclusive node. For each entity in question e, we take its maximal similarity score to the candidate in other graph and compute a difference between two probabilities $P_{match}(s_{max}(e) \leq x) - P_{excl}(s_{max}(e) \geq x)$ as heuristic score, c.f. Fig. 2. This score is large if the maximal similarity of exclusive nodes is smaller than that of nodes with matchings. We keep only entities with the score greater than threshold t, where t is a hyperparameter. This way, we make sure that the score is used only if matching and exclusive nodes are distinguishable. If there are not enough entities that fulfill this requirement, we use some simple fallback heuristic, e.g., degree, for the remaining nodes.

5 Evaluation Framework

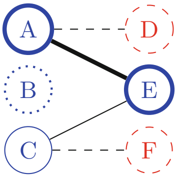


Fig. 3. Visualization of node categorisation for $\mathcal{E}^L = \{A, B, C\}$, and $\mathcal{E}^R = \{D, E, F\}$. Solid lines represent training alignments, whereas dashed ones denote test alignments. Node B is the only exclusive node. All blue nodes are in the initial pool $\mathcal{P}_0$. The red dashed nodes D and F may not be requested for labeling as they neither are exclusive nor participate in a training alignment. When node A is requested, only the alignment (A, E) is returned, and A, as well as E, become unavailable. The second training alignment (C, E) can still be obtained by requesting C. (Color figure online)

To evaluate active learning heuristics in-vitro, an alignment dataset comprising two graphs and labeled alignments is used. These alignments are split into training alignments $\mathcal{A}_{train}$ and test alignments $\mathcal{A}_{test}$. We employ an incremental batch-wise pool-based framework. At step i, there is a pool of potential queries $\mathcal{P}_i \subseteq \left(\mathcal{E}^L \cup \mathcal{E}^R\right)$, from which a heuristic selects a fixed number of elements $\mathcal{Q}_i \subseteq \mathcal{P}_i$, where $b = |\mathcal{Q}_i|$ is often called the budget. These queries are then passed to an alignment oracle $\mathfrak{O}$ simulating the labeling process and returning $\mathfrak{O}(\mathcal{Q}_i) = (\mathcal{A}_i, \mathcal{X}_i^L, \mathcal{X}_i^R)$, where the first component comprises the discovered alignments $\mathcal{A}_i = \{(a, a') \in \mathcal{A}_{train} \mid \{a, a'\} \cap \mathcal{Q}_i \neq \emptyset\}$, and the last components the exclusive nodes $\mathcal{X}_i^L = \mathcal{X}^L \cap \mathcal{Q}_i$, and $\mathcal{X}_i^R$ analogously. Afterward, the labeled nodes are removed from the pool, i.e. $\mathcal{P}_{i+1} = \mathcal{P}_i \setminus \left(\mathcal{A}_i^L \cup \mathcal{A}_i^R \cup \mathcal{X}_i^L \cup \mathcal{X}_i^R\right)$. Note that when dealing with 1:n matchings, we remove all matches from the set of available nodes, despite some of them having additional alignment partners. As each alignment edge can be retrieved using any of its endpoints, this does not pose a problem. Now, the model is trained with all already found alignments, denoted by $\mathcal{A}_{\leq i}$, and without all exclusive nodes discovered so far, denoted by $\mathcal{X}_{\leq i}^L, \mathcal{X}_{\leq i}^R$, given as

$$\mathcal{A}_{\leq i} = \bigcup_{j \leq i} \mathcal{A}_j, \quad \mathcal{X}_{\leq i}^L = \bigcup_{j \leq i} \mathcal{X}_j^L, \quad \mathcal{X}_{\leq i}^R = \bigcup_{j \leq i} \mathcal{X}_j^R.$$

Following [27,32], we do not reset the parameters but warm-start the model with the previous iteration's parameters. The pool is initialized with $\mathcal{P}_0 := \mathcal{A}^L_{train} \cup \mathcal{A}^R_{train} \cup \mathcal{X}^L \cup \mathcal{X}^R$. We exclude nodes that are not contained in the training alignment, but in the test alignments, as in this case, either a test alignment has to be revealed, or a node has to be unfaithfully classified as exclusive. An illustration of the pool construction and an example query of size one is given in Fig. 3.

6 Experiments

6.1 Setup

For evaluation, we use both subsets of the WK3l-15k dataset [7][1]. Similarly to [28] we extract additional entity alignments from the triple alignments. Besides using the official train-test split, we perform an additional 80-20 train-validation split shared across all runs. We additionally evaluate the transferability of the hyperparameter settings. One of the challenges in active learning is that hyperparameter search for a new dataset is not possible because of the lack of labeled data at the beginning. Therefore, for the evaluation of the second subset `en-fr`, we use the best hyperparameter settings which we obtained using `en-de` and compare how consistent are results for both subsets.

We employ a GNN-based model, GCN-Align [40]. We use the best settings as found in [3]. To allow for Monte-Carlo Dropout estimation for the Bayesian heuristics, we additionally add a dropout layer between the embeddings and the GCN and vary the dropout rate. We employ a margin-based matching loss, and we exclude so far identified exclusive nodes from the pool of negative samples. Following [2], we use 25 runs with different dropout masks for Bayesian approaches. As evaluation protocol, we always retrieve 200 queries from the heuristic, update the exclusives and alignments using the oracle, and train the model for up to 4k epochs with early stopping on validation mean reciprocal rank (MRR) evaluated every 20 epochs, with a patience value of 200 epochs. There are different scores for the evaluation of entity alignment, which evaluate different performance aspects [4]. In this work, we report Hits@1 (H@1) on the test alignments since this metric is most relevant for the applications. We selected the heuristics' hyperparameters according to the AUC of the step vs. validation H@1 score. Using the best hyperparameter configuration, we re-ran the experiments five times and report the mean and the standard deviation of the results on the test set.

6.2 Results

Removal of Exclusives – Figure 4 shows the test performance of the random selection baseline heuristic compared to the number of queries, with the

[1] Note that the frequently used DBP15k dataset is not suitable for our experiments due to its construction. Exclusive nodes in DBP15K are exactly those having a degree of one and are therefore trivial to identify.

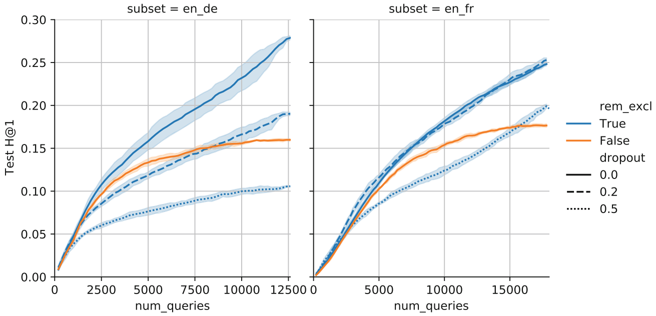


Fig. 4. Performance vs number of queries for random baseline with different levels of dropout, and when removing exclusive nodes from message passing. Removing exclusives significantly improves the final performance.

standard deviation across five runs shown as shaded areas. As can be seen by comparing the two solid lines, removing exclusives is advantageous, in particular, when many queries are performed, i.e., many exclusives are removed. Therefore, we focus the subsequent analysis only on the case, when found exclusives are removed from the graph. Moreover, we can see that using a high dropout value of 0.5 is disadvantageous on both datasets. While a dropout value of 0.2 also hurts performance for the `en-de` subset, it does not have a negative influence on `en-fr`.

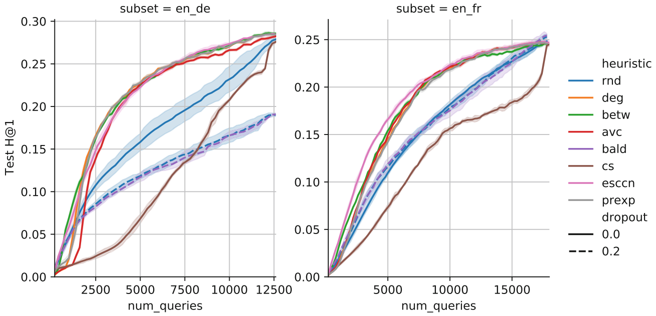


Fig. 5. Performance on test alignments vs. number of queries for different heuristics.

Comparison of Different Heuristics – Figure 5 compares the performance of different heuristics through all steps. Since there is a large overlap across different heuristics, we additionally compute AUC for each heuristic and report it in Table 1. From the results, we observe that our expectations about the performance of different heuristics are mostly confirmed. Most of the heuristics perform significantly better than random sampling. Our intuitions about possible problems with *coreset* in the context of entity alignment are also verified:

Table 1. Mean and standard deviation of AUC number of queries vs. test hits @ 1 aggregated from five different runs for each heuristic and subset. The * symbol indicates significant results compared to the `rnd` baseline according to unequal variances t-test (Welch's t-test) with $p < 0.01$.

Subset	en-de	en-fr
avc	0.2020 ± 0.0005*	0.1748 ± 0.0005*
bald	0.1222 ± 0.0039*	0.1514 ± 0.0013
betw	0.2134 ± 0.0005*	0.1773 ± 0.0004*
cs	0.1117 ± 0.0011*	0.1185 ± 0.0016*
deg	0.2105 ± 0.0005*	0.1741 ± 0.0005*
esccn	0.2114 ± 0.0006*	0.1828 ± 0.0021*
prexp	0.2103 ± 0.0009*	0.1733 ± 0.0009*
rnd	0.1605 ± 0.0040	0.1510 ± 0.0019

The heuristic performs consistently worse than the random sampling baseline. On the other hand, our new *esccn* heuristic, which also tries to cover embedding space, but uses most central nodes instead, is one of the best performing heuristics. We also observe an inferior performance of the uncertainty-based heuristic, which performance is comparable with the random heuristic. Note, that we also evaluated softmax entropy and maximal variation ratio heuristics from [17] and their performance was similar. Overall, we see similar patterns for both subsets: There is a set of good performing heuristics and their performance is very similar.

Performance in Earlier Stages – In many real-life applications, the labeling budget is limited; therefore, the model performance in the first steps is of higher relevance. Therefore, in Fig. 6, we analyze the model performance in the first

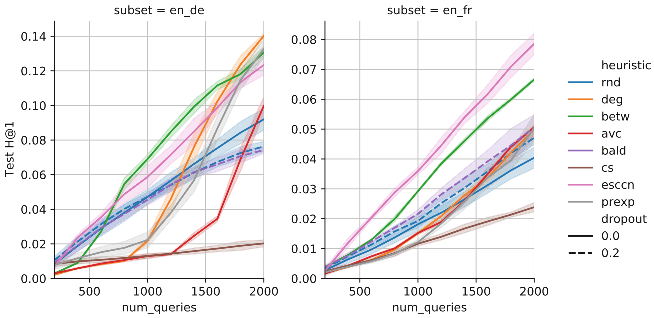


Fig. 6. Performance on test alignments vs. number of queries for different heuristics. This figure shows only queries up to 2,000, i.e., the region where not many alignments have been found so far.

2,000 iterations. We observe that the *escnn* and *betw* heuristics compete for first prize and that towards the end, they are superseded by other heuristics.

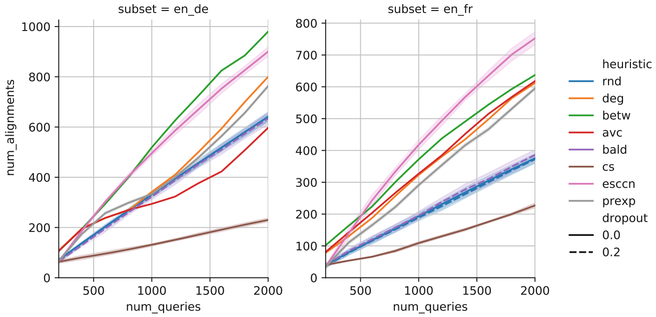


Fig. 7. Number of found training alignments vs. number of queries for different heuristics. This figure shows only queries up to 2,000, i.e., the region where not many alignments have been found so far.

Influence of Positive Matchings – In Fig. 7, we show the number of alignment pairs identified by each heuristic in the first 2,000 steps. For most heuristics, the plots look very similar to the plots in Fig. 6 above with the performance on the y axis. In Fig. 4, we also saw that the removal of exclusive nodes affects the performance only at later iterations. Therefore, we can conclude that finding nodes with matches is especially important in the early training stages.

On the whole, we can conclude that node centrality based heuristics like *betw* are the right choice for active learning for entity alignment. It achieves performance comparable with model-based approaches and does not require access to model predictions during the labeling process. The labeling ordering can be precomputed and does not change, also facilitating to parallelize the labeling process for a fixed budget to multiple annotators, e.g., using systems such as Amazon Mechanical Turk.

7 Conclusion

In this paper, we introduced the novel task of active learning for entity alignment and discussed its differences to the classical active learning setting. Moreover, we proposed several different heuristics, both, adaptions of existing heuristics used for classification, as well as heuristics specifically designed for this particular task. In a thorough empirical analysis, we showed strong performance of simple centrality and graph cover heuristics, while adaptations of state-of-the-art heuristics for classification showed inferior performance. For future work, we envision transferring our approaches to other graph matching problems, such as matching road networks [13] or approximating graph edit distance [24]. Moreover, we aim to study the generalization of our findings to other datasets and models.

Acknowledgement. This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibilities for its content.

References

1. Bast, H., Björn, B., Haussmann, E.: Semantic search on text and knowledge bases. Found. Trends Inf. Retrieval **10**(2–3), 119–271 (2016)
2. Beluch, W.H., Genewein, T., Nürnberger, A., Köhler, J.M.: The power of ensembles for active learning in image classification. In: CVPR, pp. 9368–9377. IEEE Computer Society (2018)
3. Berrendorf, M., Faerman, E., Melnychuk, V., Tresp, V., Seidl, T.: Knowledge graph entity alignment with graph convolutional networks: lessons learned. arXiv preprint arXiv:1911.08342 (2019)
4. Berrendorf, M., Faerman, E., Vermue, L., Tresp, V.: Interpretable and fair comparison of link prediction or entity alignment methods with adjusted mean rank. arXiv preprint arXiv:2002.06914 (2020)
5. Cai, H., Zheng, V.W., Chang, K.C.C.: Active learning for graph embedding. arXiv preprint arXiv:1705.05085 (2017)
6. Cao, Y., Liu, Z., Li, C., Liu, Z., Li, J., Chua, T.: Multi-channel graph neural network for entity alignment. In: ACL, vol. 1, pp. 1452–1461. ACL (2019)
7. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In: IJCAI, pp. 1511–1517. ijcai.org (2017)
8. Cortés, X., Serratosa, F.: Active-learning query strategies applied to select a graph node given a graph labelling. In: Kropatsch, W.G., Artner, N.M., Haxhimusa, Y., Jiang, X. (eds.) GbRPR 2013. LNCS, vol. 7877, pp. 61–70. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38221-5_7
9. Cortés, X., Serratosa, F., Solé-Ribalta, A.: Active graph matching based on pairwise probabilities between nodes. In: Gimel'farb, G., et al. (eds.) SSPR /SPR 2012. LNCS, vol. 7626, pp. 98–106. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34166-3_11
10. Das, K., Samanta, S., Pal, M.: Study on centrality measures in social networks: a survey. Soc. Netw. Anal. Min. **8**(1), 1–11 (2018). https://doi.org/10.1007/s13278-018-0493-2
11. Dietz, L., Kotov, A., Meij, E.: Utilizing knowledge graphs for text-centric information retrieval. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, pp. 1387–1390. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3209978.3210187
12. Faerman, E., Borutta, F., Fountoulakis, K., Mahoney, M.W.: LASAGNE: locality and structure aware graph node embedding. In: WI, pp. 246–253. IEEE Computer Society (2018)
13. Faerman, E., Voggenreiter, O., Borutta, F., Emrich, T., Berrendorf, M., Schubert, M.: Graph alignment networks with node matching scores. In: Graph Representation Learning NeurIPS 2019 Workshop (2019)
14. Gal, Y.: Uncertainty in deep learning. Ph.D. thesis, University of Cambridge (2016)
15. Gal, Y., Ghahramani, Z.: Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In: ICML JMLR Workshop and Conference Proceedings, vol. 48, pp. 1050–1059. JMLR.org (2016)

16. Gal, Y., Islam, R., Ghahramani, Z.: Deep Bayesian active learning with image data. In: Proceedings of Machine Learning Research (ICML), vol. 70, pp. 1183–1192. PMLR (2017)
17. Gao, L., Yang, H., Zhou, C., Wu, J., Pan, S., Hu, Y.: Active discriminative network representation learning. In: IJCAI, pp. 2142–2148. ijcai.org (2018)
18. Geifman, Y., El-Yaniv, R.: Deep active learning over the long tail. arXiv preprint arXiv:1711.00941 (2017)
19. Guo, L., Sun, Z., Hu, W.: Learning to exploit long-term relational dependencies in knowledge graphs. In: Proceedings of Machine Learning Research (ICML), vol. 97, pp. 2505–2514. PMLR (2019)
20. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
21. Houlsby, N., Huszár, F., Ghahramani, Z., Lengyel, M.: Bayesian active learning for classification and preference learning. arXiv preprint arXiv:1112.5745 (2011)
22. Lewis, D.D., Catlett, J.: Heterogeneous uncertainty sampling for supervised learning. In: ICML, pp. 148–156. Morgan Kaufmann (1994)
23. Li, C., Cao, Y., Hou, L., Shi, J., Li, J., Chua, T.: Semi-supervised entity alignment via joint knowledge embedding model and cross-graph model. In: EMNLP/IJCNLP, vol. 1, pp. 2723–2732. ACL (2019)
24. Li, Y., Gu, C., Dullien, T., Vinyals, O., Kohli, P.: Graph matching networks for learning the similarity of graph structured objects. In: Proceedings of Machine Learning Research (ICML), vol. 97, pp. 3835–3845. PMLR (2019)
25. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: a knowledge base from multilingual wikipedias. In: CIDR (2015). www.cidrdb.org
26. Malmi, E., Gionis, A., Terzi, E.: Active network alignment: a matching-based approach. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1687–1696 (2017)
27. Ostapuk, N., Yang, J., Cudré-Mauroux, P.: ActiveLink: deep active learning for link prediction in knowledge graphs. In: WWW, pp. 1398–1408. ACM (2019)
28. Pei, S., Yu, L., Hoehndorf, R., Zhang, X.: Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In: WWW, pp. 3130–3136. ACM (2019)
29. Puthal, D., Nepal, S., Paris, C., Ranjan, R., Chen, J.: Efficient algorithms for social network coverage and reach. In: BigData Congress, pp. 467–474. IEEE Computer Society (2015)
30. Sener, O., Savarese, S.: Active learning for convolutional neural networks: a core-set approach. In: ICLR (Poster). OpenReview.net (2018)
31. Settles, B.: Active learning literature survey. University of Wisconsin-Madison Department of Computer Sciences, Technical report (2009)
32. Shen, Y., Yun, H., Lipton, Z.C., Kronrod, Y., Anandkumar, A.: Deep active learning for named entity recognition. In: ICLR (Poster). OpenReview.net (2018)
33. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: an open multilingual graph of general knowledge. In: AAAI, pp. 4444–4451. AAAI Press (2017)
34. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10587, pp. 628–644. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68288-4_37
35. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: IJCAI, pp. 4396–4402 (2018)
36. Sun, Z., et al.: Knowledge graph alignment network with gated multi-hop neighborhood aggregation. arXiv preprint arXiv:1911.08936 (2019)

37. Trisedya, B.D., Qi, J., Zhang, R.: Entity alignment between knowledge graphs using attribute embeddings. In: AAAI, pp. 297–304. AAAI Press (2019)
38. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
39. Wang, K., Zhang, D., Li, Y., Zhang, R., Lin, L.: Cost-effective active learning for deep image classification. IEEE Trans. Circ. Syst. Video Techn. **27**(12), 2591–2600 (2017)
40. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: EMNLP, pp. 349–357. ACL (2018)
41. Wu, Y., Xu, Y., Singh, A., Yang, Y., Dubrawski, A.: Active learning for graph neural networks via node feature propagation. arXiv preprint arXiv:1910.07567 (2019)
42. Xu, K., et al.: Cross-lingual knowledge graph alignment via graph matching neural network. In: ACL, vol. 1, pp. 3156–3161. ACL (2019)
43. Yang, L., Zhang, Y., Chen, J., Zhang, S., Chen, D.Z.: Suggestive annotation: a deep active learning framework for biomedical image segmentation. In: Descoteaux, M., et al. (eds.) MICCAI 2017. LNCS, vol. 10435, pp. 399–407. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66179-7_46
44. Zhang, Q., Sun, Z., Hu, W., Chen, M., Guo, L., Qu, Y.: Multi-view knowledge graph embedding for entity alignment. In: IJCAI, pp. 5429–5435. ijcai.org (2019)
45. Zhang, Y., Lease, M., Wallace, B.C.: Active discriminative text representation learning. In: AAAI, pp. 3386–3392. AAAI Press (2017)
46. Zhu, Q., Zhou, X., Wu, J., Tan, J., Guo, L.: Neighborhood-aware attentional representation for multilingual knowledge graphs. In: IJCAI, pp. 1943–1949. ijcai.org (2019)

Exploring Classic and Neural Lexical Translation Models for Information Retrieval: Interpretability, Effectiveness, and Efficiency Benefits

Leonid Boytsov[1](✉) and Zico Kolter[1,2]

[1] Bosch Center for Artificial Intelligence, Pittsburgh, USA
leonid.boytsov@us.bosch.com
[2] Carnegie Mellon University, Pittsburgh, USA
zkolter@cs.cmu.edu

Abstract. We study the utility of the lexical translation model (IBM Model 1) for English text retrieval, in particular, its neural variants that are trained end-to-end. We use the neural Model1 as an aggregator layer applied to context-free or contextualized query/document embeddings. This new approach to design a neural ranking system has benefits for effectiveness, efficiency, and interpretability. Specifically, we show that adding an *interpretable* neural Model 1 layer on top of BERT-based contextualized embeddings (1) does not decrease accuracy and/or efficiency; and (2) may overcome the limitation on the maximum sequence length of existing BERT models. The context-free neural Model 1 is less effective than a BERT-based ranking model, but it can run efficiently on a CPU (without expensive index-time precomputation or query-time operations on large tensors). Using Model 1 we produced best neural *and* non-neural runs on the MS MARCO document ranking leaderboard in late 2020.

1 Introduction

A typical text retrieval system relies on simple term-matching techniques to generate an initial list of candidates, which can be further re-ranked using a learned model [10,13]. Thus, retrieval performance is adversely affected by a mismatch between query and document terms, which is known as a vocabulary gap problem [18,74]. Two decades ago Berger and Lafferty [4] proposed to reduce the vocabulary gap and, thus, to improve retrieval effectiveness with a help of a lexical translation model called IBM Model 1 (henceforth, simply Model 1). Model 1 has strong performance when applied to finding answers in English question-answer (QA) archives using questions as queries [35,57,65,71] as well as to cross-lingual retrieval [38,73]. Yet, little is known about its effectiveness on *realistic monolingual* English queries, partly, because training Model 1 requires large query sets, which previously were not publicly available.

Research Question 1. In the past, Model 1 was trained on question-document pairs of similar lengths which simplifies the task of finding useful associations

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 63–78, 2021.
https://doi.org/10.1007/978-3-030-72113-8_5

between query terms and terms in relevant documents. It is not clear if Model 1 can be successfully trained if queries are substantially, e.g., two orders of magnitude, shorter than corresponding relevant documents.

Research Question 2. Furthermore, Model 1 was trained in a *translation* task using an expectation-maximization (EM) algorithm [9,16] that produces a sparse matrix of conditional translation probabilities, i.e., a *non-parametric* model. Can we do better by *parameterizing* conditional translation probabilities with a neural network and learning the model *end-to-end* in a *ranking*—rather than a translation—task?

To answer these research questions we experiment with lexical translation models on two recent MS MARCO collections, which have hundreds of thousands of real user queries [12,49]. Specifically, we consider a novel class of ranking models where an *interpretable* neural Model 1 layer *aggregates* an output of a token-embedding neural network. The resulting composite network (including token embeddings) is learned end-to-end using a ranking objective. We consider two scenarios: context-independent token embeddings [11,22] and contextualized token embeddings generated by BERT [17]. Note that our approach is *generic* and can be applied to other embedding networks as well.

The neural Model 1 layer produces all pairwise similarities $T(q|d)$ for all query and documents BERT word pieces, which are combined via a straightforward product-of-sum formula without any learned weights:

$$P(Q|D) = \prod_{q \in Q} \sum_{d \in D} T(q|d) P(d|D), \tag{1}$$

where $P(d|D)$ is a maximum-likelihood estimate of the occurrence of d in D. Indeed, a query-document score is a product of scores for individual query word pieces, which makes it easy to pinpoint word pieces with largest contributions. Likewise, for every query word piece we can easily identify document word pieces with highest contributions to its score. This makes our model *more interpretable* compared to prior work.

Our contributions can be summarized as follows:

1. Adding an *interpretable* neural Model 1 layer on top of BERT entails virtually no loss in accuracy *and* efficiency compared to the vanilla BERT ranker, which is not readily interpretable.
2. In fact, for long documents the BERT-based Model 1 may outperform baseline models applied to truncated documents, thus, overcoming the limitation on the maximum sequence length of existing pretrained Transformer [67] models. However, evidence was somewhat inconclusive and we found it was also not conclusive for previously proposed CEDR [44] models that too incorporate an aggregator layer (though a *non*-interpretable one);
3. A fusion of the non-parametric Model 1 with BM25 scores can outperform the baseline models, though the gain is modest (≈3%). In contrast, the fusion with the context-free neural Model 1 can be substantially (≈10%) more effective than the fusion with its non-parametric variant. We show that the neural

Model 1 can be sparsified and executed on a CPU more than 10^3 times faster than a BERT-based ranker on a GPU. We can, thus, improve the first retrieval stage *without expensive* index-time precomputation approaches.

2 Related Work

Translation Models for Text Retrieval. This line of work begins with an influential paper by Berger and Lafferty [4] who first applied Model 1 to text retrieval [4]. It was later proved to be useful for finding answers in *monolingual* QA archives [35,57,65,71] as well as for cross-lingual document retrieval [38,73]. Model 1 is a *non-parametric* and *lexical* translation model that learns *context-independent* translation probabilities of lexemes (or tokens) from a set of paired documents called a *parallel corpus* or *bitext*. The learning method is a variant of the expectation-maximization (EM) algorithm [9,16].

A generic approach to improve performance of non-parametric statistical learning models consists in parameterizing respective probabilities using neural networks. An early successful implementation of this idea in language processing were the hybrid HMM-DNN/RNN systems for speech recognition [5,26]. More concretely, our proposal to use the neural Model 1 as a last network layer was inspired by the LSTM-CRF [32] and CEDR [44] architectures.

There is prior history of applying the neural Model 1 to retrieval, however, without training the model on a ranking task. Zuccon et al. [75] computed translation probabilities using the cosine similarity between word embeddings (normalized over the sum of similarities for top-k closest words). They achieved modest 3–7% gains on four *small-scale* TREC collections. Ganguly et al. [19] used a nearly identical approach (on similar TREC collections) and reported slightly better (6–12%) gains. Neither Zuccon et al. [75] nor Ganguly et al. [19] attempted to learn translation probabilities from a large set of real user queries.

Zbib et al. [73] employed a context-*dependent* lexical neural translation model for *cross-lingual* retrieval. They first learn context-dependent translation probabilities from a bilingual parallel corpus in a lexical *translation* task. Given a document, highest translation probabilities together with respective tokens are precomputed in advance and stored in the index. Zbib et al. [73] trained their model on aligned sentences of similar lengths. In the case of *monolingual retrieval*, however, we do not have such fine-grained training data as queries are paired only with much longer relevant documents. To our knowledge, there is no *reliable* way to obtain sentence-level relevance labels from this data.

Neural Ranking models have been a popular topic in recent years [24], but the success of early approaches—which predate BERT—was controversial [40]. This changed with adoption of large pretrained models [55], especially after the introduction of the Transformer models [17] and release of BERT [17]. Nogueira and Cho were first to apply BERT to ranking of text documents [50]. In the TREC 2019 deep learning track [12] as well as on the MS MARCO leaderboard [1], BERT-based models outperformed all other approaches by a large margin.

The Transformer model [67] uses an attention mechanism [3] where each sequence position can attend to all the positions in the previous layer.

Because self-attention complexity is quadratic with respect to a sequence length, Transformer models (BERT including) support only limited-length inputs. A number of proposals—see Tay et al. [66] for a survey—aim to mitigate this constraint, which is complementary to our work.

To process longer documents with existing pretrained models, one has to split documents into several chunks, process each chunk separately, and aggregate results, e.g., by computing a maximum or a weighted prediction score [15,72]. Such models cannot be trained end-to-end on full documents. Furthermore, a training procedure has to assume that each chunk in a relevant document is relevant as well, which is not quite accurate. To improve upon simple aggregation approaches, MacAvaney et al. [44] combined output of several document chunks using three simpler models: KNRM [70], PACRR [33], and DRMM [23]. A more recent PARADE architectures use even simpler aggregation approaches [39]. However, none of the mentioned aggregator models is interpretable and we propose to replace them with our neural Model 1 layer.

Interpretability and Explainability of statistical models has become a busy area of research. However, a vast majority of approaches rely on training a separate explanation model or exploiting saliency/attention maps [41,59]. This is problematic, because explanations provided by extraneous models cannot be verified and, thus, trusted [59]. Moreover, saliency/attention maps reveal which data parts are being processed by a model, but not *how* the model processes them [34,59,62]. Instead of producing unreliable post hoc explanations, Rudin [59] advocates for networks whose computation is transparent *by design*. If full transparency is not feasible, there is still a benefit of *last-layer* interpretability.

In text retrieval we know only two implementations of this idea. Hofstätter et al. [29] use a kernel-based formula by Xiong et al. [70] to compute soft-match counts over contextualized embeddings. Because each pair of query-document tokens produces several soft-match values corresponding to different thresholds, it is problematic to aggregate these values in an explainable way. Though this approach does offer insights into model decisions, the aggregation formula is a relatively complicated two-layer neural network with a non-linear (logarithm) activation function after the first layer [29]. ColBERT in the re-ranking mode can be seen as an interpretable interaction layer, however, unlike the neural Model 1 its use entails a 3% degradation in accuracy [37].

Efficiency. It is possible to speed-up ranking by deferring some computation to index time. They can be divided into two groups. First, it is possible to precompute separate query and document representations, which can be quickly combined at query-time in a non-linear fashion [20,37]. This method entails little to no performance degradation. Second, one can generate (or enhance) independent query and document representations to compare them via the inner-product computation. Representations—either dense or sparse—were shown to improve the first-stage retrieval albeit at the cost of expensive indexing processing and some loss in effectiveness. In particular, Khattab et al. [36] show that *dense* representations are inferior to the vanilla BERT ranker [52] in a QA task.

In the case of sparse representations, one can rely on Transformer [67] models to generate importance weights for document or query terms [14], augment documents with most likely query terms [51,52], or use a combination of these methods [43]. Due to sparsity of data generated by term expansion and re-weighting models, it can be stored in a traditional inverted file to improve performance of the first retrieval stage. However, these models are less effective than the vanilla BERT ranker [52] and they require *costly* index-time processing.

3 Methods

Token Embeddings and Transformers. We assume that an input text is split into small chunks of texts called *tokens*. A token can be a complete English word, a word piece, or a lexeme (a lemma). The length of a document d—denoted as $|d|$—is measured in the number of tokens. Because neural networks cannot operate directly on text, a sequence of tokens $t_1 t_2 \ldots t_n$ is first converted to a sequences of d-dimensional embedding vectors $w_1 w_2 \ldots w_n$ by an *embedding* network. Initially, embedding networks were context independent, i.e., each token was always mapped to the same vector [11,22,46]. Peters et al. [55] demonstrated superiority of *contextualized*, i.e., context-dependent, embeddings produced a multi-layer bi-directional LSTM [21,27,61] pretrained on a large corpus in a *self-supervised* manner. These were later outstripped by large pretrained Transformers [17,56].

In our work we use two types of embeddings: vanilla context-free embeddings (see [22] for an excellent introduction) and BERT-based contextualized embeddings [17]. Due to space constraints, we do not discuss BERT architecture in detail (see [17,60] instead). It is crucial, however, to know the following:

- Contextualized token embeddings are vectors of the last-layer hidden state;
- BERT operates on word pieces [69] rather than complete words;
- The vocabulary has close to 30K tokens and includes two special tokens: `[CLS]` (an aggregator) and `[SEP]`(a separator);
- `[CLS]` is always prepended to every token sequence and its embedding is used as a sequence representation for classification and ranking tasks.

The "vanilla" BERT ranker uses a single fully-connected layer as a prediction head, which converts the `[CLS]` vector into a scalar. It makes a prediction based on the following sequence of tokens: `[CLS]` q `[SEP]` d `[SEP]`, where q is a query and $d = t_1 t_2 \ldots t_n$ is a document. Long documents and queries need to be truncated so that the overall number of tokens does not exceed 512. To overcome this limitation, MacAvaney et al. [44] proposed an approach that:

- splits longer documents d into m chunks: $d = d_1 d_2 \ldots d_m$;
- generates m token sequences `[CLS]` q `[SEP]` d_i `[SEP]`;
- processes each sequence with BERT to generate contextualized embeddings for regular tokens as well as for `[CLS]`.

The outcome of this procedure is m `[CLS]`-vectors cls_i and n contextualized vectors $w_1 w_2 \ldots w_n$: one for *each* document token t_i. MacAvaney et al. [44] explore several approaches to combine these contextualized vectors.

First, they extend the vanilla BERT ranker by making prediction on the average `[CLS]` token: $\frac{1}{m}\sum_{i=1}^{m} cls_i$. Second, they use contextualized embeddings as a direct replacement of context-free embeddings in the following neural architectures: KNRM [70], PACRR [33], and DRMM [23]. Third, they introduced a CEDR architecture where the `[CLS]` embedding is *additionally* incorporated into KNRM, PACCR, and DRMM in a model-specific way, which further boosts performance.

Non-parametric Model 1. Let $P(D|Q)$ denote a probability that a document D is relevant to the query Q. Using the Bayes rule, $P(D|Q)$ is convenient to re-write as $P(D|Q) \propto P(Q|D)P(D)$. Assuming a uniform prior for the document occurrence probability $p(D)$, one concludes that the relevance probability is proportional to $P(Q|D)$. Berger and Lafferty proposed to estimate this probability with a *term-independent* and *context-free* model known as Model 1 [4].

Let $T(q|d)$ be a probability that a query token q is a translation of a document token d and $P(d|D)$ is a probability that a token d is "generated" by a document D. Then, a probability that query Q is a translation of document D can be computed as a product of individual query term likelihoods as follows:

$$\begin{aligned} P(Q|D) &= \prod_{q \in Q} P(q|D) \\ P(q|D) &= \sum_{d \in D} T(q|d)P(d|D) \end{aligned} \tag{2}$$

The summation in Eq. 3 is over *unique* document tokens. The in-document term probability $P(d|D)$ is a maximum-likelihood estimate. Making the non-parametric Model 1 effective requires quite a few tricks. First, $P(q|D)$—a likelihood of a query term q—is linearly combined with the collection probability $P(q|C)$ using a parameter λ [65,71].[1]

$$P(q|D) = (1-\lambda)\left[\sum_{d \in D} T(q|d)P(d|D)\right] + \lambda P(q|C). \tag{3}$$

We take several additional measures to improve Model 1 effectiveness:

- We propose to create a parallel corpus by splitting documents and passages into small contiguous chunks whose length is comparable to query lengths;
- $T(q|d)$ are learned from a symmetrized corpus as proposed by Jeon et al. [35];
- We discard all translation probabilities $T(q|d)$ below an empirically found threshold of about 10^{-3} and keep at most 10^6 most frequent tokens;
- We make self-translation probabilities $T(t|t)$ to be equal to an empirically found positive value and rescale $T(t'|t)$ so that $\sum_{t'} T(t'|t) = 1$ as in [35,65];

Our Neural Model 1. Let us rewrite Eq. 2 so that the inner summation is carried out over all document tokens rather than over the set of unique ones. This is particularly relevant for contextualized embeddings where embeddings of identical

[1] $P(q|C)$ is a maximum-likelihood estimate. For an out-of-vocabulary term q, $P(q|C)$ is set to a small number (e.g., 10^{-9}).

tokens are not guaranteed to be the same (and typically they are not):

$$P(Q|D) = \prod_{q \in Q} \sum_{i=1}^{|D|} \frac{T(q|d_i)}{|D|}. \tag{4}$$

We further propose to compute $T(q|d)$ in Eq. 4 by a simple and efficient neural network. Networks "consumes" context-free or contextualized embeddings of tokens q and d and produces a value in the range $[0, 1]$. To incorporate a self translation probability—crucial for good convergence of the context-free model—we set $T(t|t) = p_{self}$ and multiply all other probabilities by $1 - p_{self}$. However, it was not practical to scale conditional probabilities to ensure that $\forall t_2 \ \sum_{t_1} T(t_1|t_2) = 1$. Thus, $T(t_1|t_2)$ is a similarity function, but not a true probability distribution. Note that—unlike CEDR [43]—we do not use the embedding of the `[CLS]` token.

We explored several approaches to neural parametrization of $T(t_1|t_2)$. Let $\text{embed}_q(t_1)$ and $\text{embed}_d(t_2)$ denote embeddings of query and document tokens, respectively. One of the simplest approaches is to learn separate embedding networks for queries and documents and use the scaled cosine similarity:

$$T(t_1|t_2) = 0.5\{\cos(\text{embed}_q(t_1), \text{embed}_d(t_2)) + 1\}.$$

However, this neural network is not sufficiently expressive and the resulting context-free Model 1 is inferior to the non-parametric Model 1 learned via EM. We then found that a key performance ingredient was a concatenation of embeddings with their Hadamard product, which we think helps the following layers discover better interaction features. We pass this combination through one or more fully-connected linear layer with RELUs [25] followed by a sigmoid:

$$\begin{aligned} T(q|d) &= \sigma(F_3(\text{relu}(F_2(\text{relu}(F_1([x_q, x_d, x_q \circ x_d])))))) \\ x_q &= P_q(\tanh(\text{layer-norm}(\text{embed}_q(q)))) \\ x_d &= P_d(\tanh(\text{layer-norm}(\text{embed}_d(d)))), \end{aligned}$$

where P_q, P_d, and F_i are fully-connected linear layers; $[x, y]$ is vector concatenation; layer-norm is layer normalization [2]; $x \circ y$ is the Hadamard product.

Neural Model 1 Sparsification/Export to Non-Parametric Format. We can precompute $T(t_1|t_2)$ for all pairs of vocabulary tokens, discard small values (below a threshold), and store the result as a sparse matrix. This format permits an extremely efficient execution on CPU (see results in Sect. 4.2).

4 Experiments

4.1 Setup

Data Sets. We experiment with MS MARCO collections, which include data for passage and document retrieval tasks [12,49]. Each MS MARCO collection has a large number of real user queries (see Table 1). To our knowledge, there are no other collections comparable to MS MARCO in this respect. The large set of queries is sampled from the log file of the search engine Bing. In that, data set creators ensured that all queries can be answered using a short text snippet. These queries are only sparsely judged (about one relevant passage per query). Sparse judgments are binary: Relevant documents have grade one and all other documents have grade zero.

In addition to large query sets with sparse judgments, we use two evaluation sets from TREC 2019/2020 deep learning tracks [12]. These query sets are quite small, but they have been thoroughly judged by NIST assessors *separately* for a document and a passage retrieval task. TREC NIST judgements range from zero (not-relevant) to three (perfectly relevant).

Table 1. MS MARCO data set details

	Documents	Passages
# of documents	3.2M	8.8M
Avg. # of doc. lemmas	476.7	30.6
Avg. # of query lemmas	3.2	3.5
	# of queries	
Train/fusion	10K	20K
Train/modeling	357K	788.7K
Development	2500	20K
Test	2693	3000
TREC 2019	100	100
TREC 2020	100	100

We randomly split publicly available training and validation sets into the following subsets: a small training set to train a linear fusion model (`train/fusion`), a large set to train neural models and non-parametric Model 1 (`train/modeling`), a development set (`development`), and a test set (`MS MARCO test`) containing at most 3K queries. Detailed data set statistics is summarized in Table 1. Note that the training subsets were obtained from the original training set, whereas the new development and test sets were obtained from the original development set. The leaderboard validation set is not publicly available.

We processed collections using Spacy 2.2.3 [30] to extract tokens (text words) and lemmas (lexemes) from text. The frequently occurring words and lemmas were filtered out using Indri's list of stopwords [64], which was expanded to include a few contractions such as "n't" and "'ll". Lemmas were indexed using Lucene 7.6. We also generated sub-word tokens, namely BERT word pieces [17, 69], using a HuggingFace Transformers library (version 0.6.2) [68]. We did *not* apply the stopword list to BERT word pieces.

Basic Setup. We experimented on a Linux server equipped with a six-core (12 threads) i7-6800K 3.4 Ghz CPU, 125 GB of memory, and four GeForce GTX 1080 TI GPUs. We used the text retrieval framework `FlexNeuART` [8], which is implemented in Java. It employs Lucene 7.6 with a BM25 scorer [58] to generate an initial list of candidates, which can be further re-ranked using either traditional or neural re-rankers. The traditional re-rankers, including the non-parametric Model 1, are implemented in Java as well. They run in a *multi-threaded mode* (12 threads) and *fully* utilize the CPU. The neural rankers are implemented using PyTorch 1.4 [54] and Apache Thrift.[2] A neural ranker operates as a standalone *single-threaded* server. Our software is available online [8].[3]

[2] https://thrift.apache.org/.

[3] https://github.com/oaqa/FlexNeuART.

Ranking speed is measured as the overall CPU/GPU *throughput*—rather than latency—per one *thousand* of documents/passages. Ranking accuracy is measured using the standard utility `trec_eval` provided by TREC organizers.[4]. Statistical significance is computed using a two-sided t-test with threshold 0.05.

All ranking models are applied to the candidate list generated by a tuned BM25 scorer [58]. BERT-based models re-rank 100 entries with highest BM25 scores: using a larger pool of candidates hurts both efficiency and accuracy. All other models, including the neural context-free Model 1 re-rank 1000 entries: Further increasing the number of candidates does not improve accuracy.

Training Models. Neural models are trained using a pairwise margin loss.[5] Training pairs are obtained by combining known relevant documents with 20 negative examples selected from a set of top-500 candidates returned by Lucene. In each epoch, we randomly sample one positive and one negative example per query. BERT-based models first undergo a target-corpus pretraining [31] using a masked language modeling and next-sentence prediction objective [17]. Then, we train them for one epoch in a ranking task. We use batch size 16 simulated via gradient accumulation. Context-free Model 1 is trained from scratch for 32 epochs using batch size 32. The non-parametric Model 1 is trained for five epochs with MGIZA [53].[6] Further increasing the number of epochs does not substantially improve results. MGIZA computes probabilities of spurious insertions (i.e., a translation from an empty word), but we discard them as in prior work [65].

We use a small weight decay (10^{-7}) and a warm-up schedule where the learning rate grows linearly from zero for 10–20% of the steps until it reaches the base learning rate [48,63]. The optimizer is AdamW [42]. For BERT-based models we use different base rates for the fully-connected prediction head ($2 \cdot 10^{-4}$) and for the main Transformer layers ($2 \cdot 10^{-5}$). For the context-free Model 1 the base rate is $3 \cdot 10^{-3}$, which is decayed by 0.9 after each epoch. The learning rate is the same for all parameters.

The trained *neural* Model 1 is "exported" to a non-parametric format by precomputing all pairwise translation probabilities and discarding probabilities smaller than 10^{-4}. This sparsification/export procedure takes three minutes and the exported model is executed using the same Java code as the non-parametric Model 1. Each neural model and the sparsified Model 1 is trained and evaluated for five seeds. To this end, we compute the value for each query and seed and average query-specific values (over five seeds). All hyper-parameters are tuned on a development set.

Because context-free Model 1 rankers are not strong on their own, we evaluate them in a *fusion* mode. First, Model 1 is trained on `train/modeling`. Then we linearly combine a model score with the BM25 score [58]. Optimal weights are computed on a `train/fusion` subset using the coordinate ascent algorithm [45] from RankLib.[7] To improve effectiveness of this linear fusion, we use Model 1

[4] https://github.com/usnistgov/trec_eval.

[5] We use the loss reduction type `sum`.

[6] https://github.com/moses-smt/mgiza/.

[7] https://sourceforge.net/p/lemur/wiki/RankLib/.

Table 2. Evaluation results: `bwps` denotes BERT word pieces, `lemm` denotes text lemmas, and `word` denotes original words. `NN-Model1` and `NN-Model1-exp` are the context-free neural Model 1 models: They use only `bwps`. `NN-Model1` runs on GPU whereas `NN-Model1-exp` runs on CPU. Ranking speed is throughput and not latency! Statistical significance is denoted by * and $^\#$. Hypotheses are explained in the main text.

	Documents				Passages			
	MS MARCO test	TREC 2019	TREC 2020	Rank. speed	MS MARCO test	TREC 2019	TREC 2020	Rank. speed
	MRR	NDCG@10		per 1K	MRR	NDCG@10		per 1K
	Baselines							
BM25 (lemm)	0.270	0.544	0.524	0.8 ms	0.256	0.522	0.516	0.5 ms
BM25 (lemm)+BM25 (word)	0.274	0.544	0.523	2.5 ms	0.265	0.517	0.521	0.7 ms
BM25 (lemm)+BM25 (bwps)	0.283	0.528	0.537	2.2 ms	0.270	0.518	0.525	0.9 ms
BERT-vanilla (short)	0.387	0.655	0.623	39 s	**0.426**	0.686	0.684	15 s
BERT-vanilla (full)	0.376$^\#$	0.667	0.631	82 s				
BERT-CEDR-KRNM	0.387	0.665	0.649*	88 s	0.421*	0.682	0.675	16 ms
BERT-CEDR-DRMM	0.377*	0.667	0.636	120 s	0.425	0.688	**0.685**	30 s
BERT-CEDR-PACRR	**0.392**	**0.670**	**0.652***	81 s	0.425	**0.690**	0.684	16 s
	Our methods							
BM25 (lemm)+Model1 (word)	0.283*	0.548	0.535	13 ms	0.274*	0.522	0.567*	1.2 ms
BM25 (lemm)+Model1 (bwps)	0.284	0.557	0.525	33 ms	0.271	0.517	0.509	2.7 ms
BM25 (lemm)+NN-Model1-exp	0.307*	0.568	0.545	16 ms	0.298*	0.541*	0.581*	2.4 ms
BM25 (lemm)+NN-Model1	0.311*	0.566	0.541	3 s	0.300*	0.549*	0.587*	0.32 s
BERT-Model1 (short)	0.384	0.657	0.631	36 s	**0.426**	0.685	0.682	16 s
BERT-Model1 (full)	0.391$^\#$	0.666	0.637*	80 s				

log-scores *normalized* by the number of query words. In turn, BM25 scores are normalized by the sum of query-term IDF values (see [58] for the description of BM25 and IDF). As one of the baselines, we use a fusion of BM25 scores for different tokenization approaches (basically a multi-field BM25). Fusion weights are obtained via RankLib on `train/fusion`.

4.2 Results

Model Overview. We compare several models (see Table 2). First, we use BM25 scores [58] computed for the lemmatized text, henceforth, `BM25 (lemm)`. Second, we evaluate several variants of the context-free Model 1. The non-parametric Model 1 was trained for both original words and BERT word pieces: Respective models are denoted as `Model1 (word)` and `Model1 (bwps)`. The neural context-free Model 1—denoted as `NN-Model1`—was used only with BERT word pieces. This model was sparsified and exported to a non-parametric format (see Sect. 3), which runs efficiently on a CPU. We denote it as `NN-Model1-exp`. Note that context-free Model 1 rankers are not strong on their own, thus, we evaluate them in a *fusion* mode by combining their scores with `BM25 (lemm)`.

Crucially, all context-free models incorporate exact term-matching signal via either the self-translation probability or via explicit smoothing with a word collection probability (see Eq. 3). Thus, these models should be compared not only with BM25, but also with the fusion model incorporating BM25 scores for original words or BERT word pieces. We denote these baselines as `BM25 (lemm)+ BM25 (word)` and `BM25 (lemm)+ BM25 (bwps)`, respectively.

As we describe in Sect. 3, our contextualized Model 1 applies the neural Model 1 layer to the contextualized embeddings produced by BERT. We denote this model as BERT-Model1. Due to the limitation of existing pretrained Transformer models, long documents need to be split into chunks each of which is processed, i.e., contextualized, separately. This is done in BERT-Model1 (full), BERT-vanilla (full), and BERT-CEDR [44] models. These models operate on (mostly) complete documents: For efficiency reasons we nevertheless use only the first 1431 tokens (three BERT chunks). Another approach is to make predictions on much shorter (one BERT chunk) fragments [15]. This is done in BERT-Model1 (short) and BERT-vanilla (short). In the passage retrieval task, all passages are short and no truncation or chunking is needed. Note that we use a *base*, i.e., a 12-layer Transformer [67] model, since it is more practical then a 24-layer BERT-large and performs at par with BERT-large on MS MARCO data [29].

We tested several hypotheses using a two-sided t-test:

- BM25 (lemm)+ Model1 (word) is the same as BM25 (lemm)+ BM25 (word);
- BM25 (lemm)+ Model1 (bwps) is the same as BM25 (lemm)+ BM25 (bwps);
- BERT-Model1 (full) is the same as BERT-vanilla (short);
- For each BERT-CEDR model, we test if it is the same as BERT-vanilla (short);
- BERT-vanilla (full) is the same as BERT-vanilla (short);
- BERT-Model1 (full) is the same as BERT-Model1 (short);

The main purpose of these tests is to assess if special aggregation layers (including the neural Model 1) can be more accurate compared to models that run on truncated documents. In Table 2 statistical significance is indicated by a special symbol: the last two hypotheses use #; all other hypotheses use $\star$.

Discussion of Results. The results are summarized in Table 2. First note that there is less consistency in results on TREC 2019/2020 sets compared to MS MARCO test sets. In that, some statistically significant differences (on MS MARCO test) "disappear" on TREC 2019/2020. TREC 2019/2020 query sets are quite small and its more likely (compared to MS MARCO test) to obtain spurious results. Furthermore, the fusion model BM25 (lemm)+ Model1 (bwps) is either worse than the baseline model BM25 (lemm)+ BM25 (bwps) or the difference is not significant. BM25 (lemm)+ Model1 (word) is mostly better than the respective baseline, but the gain is quite small. In contrast, the fusion of the neural Model 1 with BM25 scores for BERT word pieces is more accurate on all the query sets. On the MS MARCO test sets it is 15–17% better than BM25 (lemm). These differences are significant on both MS MARCO test sets as well as on TREC 2019/2020 tests sets for the passage retrieval task. Sparsification of the neural Model 1 leads only to a small (0.6–1.3%) loss in accuracy. In that, the sparsified model—executed on a CPU—is more than 10^3 times faster than BERT-based rankers, which run on a GPU. It is $5 \times 10^3 \times$ faster in the case of passage retrieval. In contrast, on a GPU, the fastest neural model KNRM is only 500 times faster than vanilla BERT [28] (also for passage retrieval). For large candidate sets computation of Model 1 scores can be further sped up (Sect. 3.1.2.1 [6]). Thus, BM25 (lemm)+NN-Model1-exp can be useful at the candidate generation stage.

We also compared BERT-based neural Model 1 with BERT-CEDR and BERT-vanilla models on the MS MARCO test set for the document retrieval task. By comparing `BERT-vanilla (short)`, `BERT-Model1 (short)`, and `BERT-Model1 (full)` we can see that the neural Model 1 layer entails virtually no efficiency or accuracy loss. In fact, `BERT-Model1 (full)` is 1.8% and 1% better than `BERT-Model1 (short)` and `BERT-vanilla (short)`, respectively. Yet, only the former difference is statistically significant.

Furthermore, the same holds for `BERT-CEDR-PACRR`, which was shown to outperform `BERT-vanilla` by MacAvaney et al. [44]. In our experiments it is 1% better than `BERT-vanilla (short)`, but the difference is neither substantial nor statistical significant. This does not invalidate results of MacAvaney et al. [44]: They compared `BERT-CEDR-PACRR` only with `BERT-vanilla (full)`, which makes predictions on the averaged `[CLS]` embeddings. However, in our experiments, this model is noticeably worse (by 4.2%) than `BERT-vanilla (short)` and the difference is statistically significant. We think that obtaining more conclusive evidence about the effectiveness of aggregation layers requires a different data set where relevance is harder to predict from a truncated document.

Leaderboard Submissions. We combined `BERT-Model1` with the strong first-stage pipeline, which uses Lucene to index documents expanded with doc2query [51, 52] and re-ranks them using a mix of traditional and `NN-Model1-exp` scores (our exported neural Model 1). This first-stage pipeline is about as effective as the Conformer-Kernel model [47]. The combination model achieved the top place on a well-known leaderboard in November and December 2020. Furthermore, using the non-parametric Model 1, we produced the best traditional run in December 2020, which outperformed several neural baselines [7].

5 Conclusion

We study a neural Model 1 combined with a context-free or contextualized embedding network and show that such a combination has benefits to efficiency, effectiveness, and interpretability. To our knowledge, the context-free neural Model 1 is the only neural model that can be sparsified to run efficiently on a CPU (up to $5 \times 10^3\times$ faster than BERT on a GPU) without expensive index-time precomputation or query-time operations on large tensors. We hope that effectiveness of this approach can be further improved, e.g., by designing a better parametrization of conditional translation probabilities.

References

1. MS MARCO leaderboard. https://microsoft.github.io/msmarco/
2. Ba, L.J., Kiros, J.R., Hinton, G.E.: Layer normalization, CoRR abs/1607.06450 (2016)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015 (2015)
4. Berger, A., Lafferty, J.: Information retrieval as statistical translation. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 222–229 (1999)

5. Bourlard, H., Bourlard, H.A., Morgan, N.: Connectionist Speech Recognition. A Hybrid Approach, vol. 247. Springer, New York (1994). https://doi.org/10.1007/978-1-4615-3210-1
6. Boytsov, L.: Efficient and Accurate Non-Metric k-NN Search with Applications to Text Matching. Ph.D. thesis, Carnegie Mellon University (2018)
7. Boytsov, L.: Traditional IR rivals neural models on the MS MARCO document ranking leaderboard (2020)
8. Boytsov, L., Nyberg, E.: Flexible retrieval with NMSLIB and FlexNeuART. In: Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS), pp. 32–43 (2020)
9. Brown, P.F., Pietra, S.D., Pietra, V.J.D., Mercer, R.L.: The mathematics of statistical machine translation: parameter estimation. Comput. Linguist. **19**(2), 263–311 (1993)
10. Büttcher, S., Clarke, C.L., Cormack, G.V.: Information Retrieval: Implementing and Evaluating Search Engines. MIT Press, Cambridge (2016)
11. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011)
12. Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Voorhees, E.M.: Overview of the TREC 2019 deep learning track, CoRR abs/2003.07820 (2020)
13. Croft, W.B., Metzler, D., Strohman, T.: Search Engines: Information Retrieval in Practice. Addison-Wesley, Reading (2010)
14. Dai, Z., Callan, J.: Context-aware sentence/passage term importance estimation for first stage retrieval, CoRR abs/1910.10687 (2019)
15. Dai, Z., Callan, J.: Deeper text understanding for IR with contextual neural language modeling. In: SIGIR, pp. 985–988. ACM (2019)
16. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc. Ser. B (Methodol.) **39**(1), 1–22 (1977)
17. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding, pp. 4171–4186 (2019)
18. Furnas, G.W., Landauer, T.K., Gomez, L.M., Dumais, S.T.: The vocabulary problem in human-system communication. Commun. ACM **30**(11), 964–971 (1987)
19. Ganguly, D., Roy, D., Mitra, M., Jones, G.J.: Word embedding based generalized language model for information retrieval. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 795–798 (2015)
20. Gao, L., Dai, Z., Callan, J.: EARL: speedup transformer-based rankers with pre-computed representation, CoRR abs/2004.13313 (2020)
21. Gers, F.A., Schmidhuber, J., Cummins, F.A.: Learning to forget: continual prediction with LSTM. Neural Comput. **12**(10), 2451–2471 (2000)
22. Goldberg, Y.: A primer on neural network models for natural language processing. J. Artif. Intell. Res. **57**, 345–420 (2016)
23. Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep relevance matching model for ad-hoc retrieval. In: CIKM, pp. 55–64. ACM (2016)
24. Guo, J., et al.: A deep look into neural ranking models for information retrieval. Inf. Process. Manage. **57**, 102067 (2019)
25. Hahnloser, R.H.R.: On the piecewise analysis of networks of linear threshold neurons. Neural Netw. **11**(4), 691–697 (1998)

26. Hinton, G., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process. Mag. **29**(6), 82–97 (2012)
27. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
28. Hofstätter, S., Hanbury, A.: Let's measure run time! extending the IR replicability infrastructure to include performance aspects. In: OSIRRC@SIGIR. CEUR Workshop Proceedings, vol. 2409, pp. 12–16. CEUR-WS.org (2019)
29. Hofstätter, S., Zlabinger, M., Hanbury, A.: Interpretable & time-budget-constrained contextualization for re-ranking. In: Frontiers in Artificial Intelligence and Applications (ECAI), vol. 325, pp. 513–520. IOS Press (2020)
30. Honnibal, M., Montani, I.: spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing (2017, to appear)
31. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. In: ACL, vol. 1, pp. 328–339. Association for Computational Linguistics (2018)
32. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging, CoRR abs/1508.01991 (2015)
33. Hui, K., Yates, A., Berberich, K., de Melo, G.: Co-PACRR: a context-aware neural IR model for ad-hoc retrieval. In: WSDM, pp. 279–287. ACM (2018)
34. Jain, S., Wallace, B.C.: Attention is not explanation. In: NAACL-HLT, vol. 1, pp. 3543–3556. Association for Computational Linguistics (2019)
35. Jeon, J., Croft, W.B., Lee, J.H.: Finding similar questions in large question and answer archives. In: CIKM, pp. 84–90. ACM (2005)
36. Khattab, O., Potts, C., Zaharia, M.: Relevance-guided supervision for OpenQA with ColBERT, CoRR abs/2007.00814 (2020)
37. Khattab, O., Zaharia, M.: ColBERT: efficient and effective passage search via contextualized late interaction over BERT. In: SIGIR, pp. 39–48. ACM (2020)
38. Lavrenko, V., Choquette, M., Croft, W.B.: Cross-lingual relevance models. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 175–182 (2002)
39. Li, C., Yates, A., MacAvaney, S., He, B., Sun, Y.: PARADE: passage representation aggregation for document reranking, CoRR abs/2008.09093 (2020)
40. Lin, J.: The neural hype and comparisons against weak baselines. In: ACM SIGIR Forum, vol. 52, pp. 40–51. ACM, New York (2019)
41. Lipton, Z.C.: The mythos of model interpretability. Commun. ACM **61**(10), 36–43 (2018)
42. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
43. MacAvaney, S., Nardini, F.M., Perego, R., Tonellotto, N., Goharian, N., Frieder, O.: Expansion via prediction of importance with contextualization. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1573–1576. ACM (2020)
44. MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: CEDR: contextualized embeddings for document ranking. In: SIGIR, pp. 1101–1104. ACM (2019)
45. Metzler, D., Croft, W.B.: Linear feature-based models for information retrieval. Inf. Retr. **10**(3), 257–274 (2007). https://doi.org/10.1007/s10791-006-9019-z
46. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)

47. Mitra, B., Hofstätter, S., Zamani, H., Craswell, N.: Conformer-kernel with query term independence for document retrieval, CoRR abs/2007.10434 (2020)
48. Mosbach, M., Andriushchenko, M., Klakow, D.: On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines, CoRR abs/2006.04884 (2020)
49. Nguyen, T., et al.: MS MARCO: a human generated MAchine Reading COmprehension dataset (November 2016)
50. Nogueira, R., Cho, K.: Passage re-ranking with BERT, CoRR abs/1901.04085 (2019)
51. Nogueira, R., Lin, J.: From doc2query to docTTTTTquery. An MS MARCO passage retrieval task [1] μpublication (2019)
52. Nogueira, R., Yang, W., Lin, J., Cho, K.: Document expansion by query prediction, CoRR abs/1904.08375 (2019)
53. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Comput. Linguist. **29**(1), 19–51 (2003)
54. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, pp. 8026–8037 (2019)
55. Peters, M.E., et al.: Deep contextualized word representations. In: Proceedings of NAACL-HLT, pp. 2227–2237 (2018)
56. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding with unsupervised learning. Technical report, OpenAI (2018)
57. Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V.O., Liu, Y.: Statistical machine translation for query expansion in answer retrieval. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, ACL 2007 (2007)
58. Robertson, S.: Understanding inverse document frequency: on theoretical arguments for IDF. J. Doc. **60**(5), 503–520 (2004)
59. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)
60. Rush, A.M.: The annotated transformer. In: Proceedings of Workshop for NLP Open Source Software (NLP-OSS), pp. 52–60 (2018)
61. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Sig. Process. **45**(11), 2673–2681 (1997)
62. Serrano, S., Smith, N.A.: Is attention interpretable? In: ACL, vol. 1, pp. 2931–2951. Association for Computational Linguistics (2019)
63. Smith, L.N.: Cyclical learning rates for training neural networks. In: WACV, pp. 464–472. IEEE Computer Society (2017)
64. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: a language-model based search engine for complex queries (2005). http://ciir.cs.umass.edu/pubfiles/ir-407.pdf. Accessed April 2017
65. Surdeanu, M., Ciaramita, M., Zaragoza, H.: Learning to rank answers to non-factoid questions from web collections. Comput. Linguist. **37**(2), 351–383 (2011)
66. Tay, Y., Dehghani, M., Bahri, D., Metzler, D.: Efficient transformers: a survey, CoRR abs/2009.06732 (2020)
67. Vaswani, A., et al.: Attention is all you need. In: NIPS, pp. 5998–6008 (2017)
68. Wolf, T., et al.: Huggingface's transformers: state-of-the-art natural language processing. arXiv abs/1910.03771 (2019)
69. Wu, Y., et al.: Google's neural machine translation system: bridging the gap between human and machine translation. CoRR abs/1609.08144 (2016)

70. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling. In: SIGIR, pp. 55–64. ACM (2017)
71. Xue, X., Jeon, J., Croft, W.B.: Retrieval models for question and answer archives. In: SIGIR, pp. 475–482 (2008)
72. Yilmaz, Z.A., Wang, S., Yang, W., Zhang, H., Lin, J.: Applying BERT to document retrieval with birch. In: EMNLP/IJCNLP, vol. 3, pp. 19–24. Association for Computational Linguistics (2019)
73. Zbib, R., et al.: Neural-network lexical translation for cross-lingual IR from text and speech. In: SIGIR, pp. 645–654. ACM (2019)
74. Zhao, L., Callan, J.: Term necessity prediction. In: Huang, J., Koudas, N., Jones, G.J.F., Wu, X., Collins-Thompson, K., An, A. (eds.) Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, pp. 259–268. ACM (2010)
75. Zuccon, G., Koopman, B., Bruza, P., Azzopardi, L.: Integrating and evaluating neural word embeddings in information retrieval. In: Proceedings of the 20th Australasian Document Computing Symposium, pp. 1–8 (2015)

Coreference Resolution in Research Papers from Multiple Domains

Arthur Brack[1](✉), Daniel Uwe Müller[2](✉), Anett Hoppe[1](✉), and Ralph Ewerth[1,2](✉)

[1] Research Group Visual Analytics, TIB – Leibniz Information Centre for Science and Technology, Hannover, Germany
{arthur.brack,anett.hoppe,ralph.ewerth}@tib.eu

[2] L3S Research Center, Leibniz University, Hannover, Germany

Abstract. Coreference resolution is essential for automatic text understanding to facilitate high-level information retrieval tasks such as text summarisation or question answering. Previous work indicates that the performance of state-of-the-art approaches (e.g. based on BERT) noticeably declines when applied to scientific papers. In this paper, we investigate the task of coreference resolution in research papers and subsequent knowledge graph population. We present the following contributions: (1) We annotate a corpus for coreference resolution that comprises 10 different scientific disciplines from Science, Technology, and Medicine (STM); (2) We propose transfer learning for automatic coreference resolution in research papers; (3) We analyse the impact of coreference resolution on knowledge graph (KG) population; (4) We release a research KG that is automatically populated from 55,485 papers in 10 STM domains. Comprehensive experiments show the usefulness of the proposed approach. Our transfer learning approach considerably outperforms state-of-the-art baselines on our corpus with an F1 score of 61.4 (+11.0), while the evaluation against a gold standard KG shows that coreference resolution improves the quality of the populated KG significantly with an F1 score of 63.5 (+21.8).

Keywords: Coreference resolution · Information extraction · Knowledge graph population · Scholarly communication

1 Introduction

Current research papers are generally published in form of PDF files. This makes them hard to handle for retrieval systems, since their content is hidden in human- but not machine-interpretable text. In consequence, current academic search engines are not able to adequately support researchers in their day-to-day tasks. This is further aggravated by the exploding number of published articles [5].

Approaches to automatically structure research papers are thus an active area of research. *Coreference resolution* is the task of identifying mentions in a text

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 79–97, 2021.
https://doi.org/10.1007/978-3-030-72113-8_6

which refer to the same entity or concept. It is an essential step for automatic text understanding and facilitates down-stream tasks such as text summarisation or question answering. For instance, the text *'Coreference resolution is... It is used for question answering...'*, has two coreferent mentions *'Coreference resolution'* and *'It'*. This allows us to extract the fact <coreference resolution, used_for, question answering>.

Current methods for coreference resolution based on deep learning achieve quite impressive results (e.g. an F1 score of 79.6 for the OntoNotes 5.0 dataset [21]) in the general domain, that is data from phone conversations, news, magazines, etc. But results of previous work indicate [11,23,34,44] that general coreference resolution systems perform poorly on scientific text. This is presumably caused by the specific terminology and phrasing used in a scientific domain. Some other studies state that annotating scientific text is costly since it demands certain expertise in the article's domain [2,6,20]. Most corpora for research papers cover only a single domain (e.g. biomedicine [11], artificial intelligence [27]) and are thus limited to these domains. As a result, the annotated corpora are relatively small and overall only a few domains are covered. Datasets for the general domain are usually much larger, but they have not been exploited yet by approaches for coreference resolution in research papers.

Coreference resolution is also one of the main steps in the KG population pipeline [28,39]. However, to date it is not clear, to which extent (a) coreference resolution can help to reduce the number of scientific concepts in the populated KG, and (b) how coreference resolution influences the quality of the populated KG. Besides, a KG comprising multiple scientific domains has not been populated yet.

In this paper, we address the task of coreference resolution in research papers and subsequent knowledge graph population. Our contributions can be summarised as follows: (1) First, we annotate a corpus for coreference resolution that consists of 110 abstracts from 10 domains from Science, Technology, and Medicine. The systematic annotation resulted in a substantial inter-coder agreement (0.68 κ). We provide and compare baseline results for this dataset by evaluating five different state-of-the-art approaches. Our experimental results confirm that state-of-the-art coreference approaches do not perform well on research papers. (2) Consequently, we propose sequential transfer learning for coreference resolution in research papers. This approach utilises our corpus by fine-tuning a model that is pre-trained on a large corpus from the general domain [37]. Experimental results show that our approach significantly outperforms the best state-of-the-art baseline (F1 score of 61.4, i.e. +11.0). (3) We investigate the impact of coreference resolution on automatic KG population. To evaluate the quality of various KG population strategies, we (i) compile a gold standard KG from our annotated corpus that contains scientific concepts referenced by mentions from text, and (ii) present a procedure to evaluate the clustering results of mentions. (4) We release (i) an automatically populated KG from 55,485 abstracts of the 10 STM domains and (ii) a gold KG (Test-STM-KG) from the annotated STM-corpus. Experimental results show that coreference resolution has only a small impact on the number of concepts in a populated KG, but it helps to improve the quality of the KG significantly: the population with coreference resolution yields an F1 score of 63.5 evaluated against the gold KG (+21.8 F1). We release the data corpora and the source code to facilitate further research: https://github.com/arthurbra/stm-coref.

The remainder of the paper is organised as follows: Sect. 2 summarises related work on coreference resolution. Section 3 describes the annotation procedure and the characteristics of the corpus, and our proposed approaches for coreference resolution, KG population, and KG evaluation. The experimental setup and results are reported in Sect. 4 and 5, while Sect. 6 concludes the paper and outlines areas of future work.

2 Related Work

2.1 Approaches for Coreference Resolution

For a given document, the task of coreference resolution is (a) to extract mentions of scientific concepts, and (b) to cluster those mentions that refer to the same concept. Recent approaches mostly rely on supervised learning and can be categorised into three groups [32]: (1) Mention-pair models [33,45] are binary classifiers that determine whether two mentions are coreferent or not. (2) Entity-mention models [9,41] determine whether a mention is coreferent to a preceding *cluster*. A cluster has more expressive features compared to a mention in mention-pair models. (3) Ranking-based models [12,25,31] simultaneously rank all candidate antecedents (i.e. preceding mention candidates). This enables the model to identify the most probable antecedent.

Lee et al. [25,26] propose an end-to-end neural coreference resolution model. It is a ranking-based model that jointly recognises mentions and clusters. Therefore, the model considers all spans in the text as possible mentions and learns distributions over possible antecedents for each mention. For computational efficiency, candidate spans and antecedents are pruned during training and inference. Joshi et al. [22] enhance Lee et al.'s model with BERT-based word embeddings [14], while Ma et al. [30] improve the model with better attention mechanisms and loss functions.

Furthermore, several approaches propose multi-task learning, such that related tasks may benefit from knowledge in other tasks to achieve better prediction accuracy: Luan et al. [27,49] train a model on three tasks (coreference resolution, entity and relation extraction) using one dataset of research papers. Sanh et al. [43] introduce a multi-task model that is trained on four tasks (mention detection, coreference resolution, entity and relation extraction) using two different datasets in the general domain.

Results of some previous studies [11,23,34,44] revealed that general coreference systems do not work well in the biomedical domain due to the lack of domain knowledge. For instance, on Colorado Richly Annotated Full Text (CRAFT) corpus [11] a coreference resolution system for the news domain achieves only 14.0 F1 (−32.0).

To the best of our knowledge, a transfer learning approach from the general to the scientific domain has not been proposed for coreference resolution yet.

2.2 Corpora for Coreference Resolution in Research Papers

For the general domain, multiple datasets exist for coreference resolution, e.g. Message Understanding Conference (MUC-7) [1], Automatic Content Extraction (ACE05) [15], or OntoNotes 5.0 [37]. The OntoNotes 5.0 dataset [37] is the largest one and is used in many benchmark experiments for coreference resolution systems [22,25,30].

Various annotated datasets for coreference resolution exist also for research papers: CRAFT corpus [11] covers 97 papers from biomedicine. The corpus of Schäfer et al. [44] contains 266 papers from computational linguistics and language technology. Chaimongkol et al. [7] annotated a corpus of 284 papers from four subdisciplines in computer science. The SciERC corpus [27] comprises 500 abstracts from the artificial intelligence domain and features annotations for scientific concepts and relations. It was used to generate an artificial intelligence (AI) knowledge graph [13]. Furthermore, several datasets exist for scientific concept extraction [2,6,27,40] and relation extraction [2,20,27] that cover various scientific domains.

To the best of our knowledge, a corpus for coreference resolution that comprises a broad range of scientific domains is not available yet.

3 Coreference Resolution in Research Papers

As the discussion of related work reveals, existing corpora for coreference resolution in scientific papers normally cover only a single domain, and coreference resolution approaches do not perform well on scholarly texts. To address these issues, we systematically annotate a corpus with coreferences in abstracts from 10 different science domains. Current approaches for coreference resolution in research papers do not exploit existing annotated datasets from the general domain, which are usually much larger than in the scientific domain. We propose a sequential transfer learning approach that takes advantage from large, annotated datasets. Finally, to the best of our knowledge, the impact of (a) coreference resolution and (b) cross-domain collapsing of mentions to scientific concepts on KG population with multiple science domains has not been investigated yet. Consequently, we present an evaluation procedure for the clustering aspect in the KG population pipeline.

In the sequel, we describe our annotated corpus, our transfer learning approach for coreference resolution, and an evaluation procedure for clustering in KG population.

3.1 Corpus for Coreference Resolution in 10 STM Domains

In this section, we describe the STM corpus [6], which we used as the basis for the annotation, our annotation process, and the characteristics of the resulting corpus.

STM Corpus: The STM corpus [6] comprises 110 articles from 10 domains in Science, Technology and Medicine, namely Agriculture (Agr), Astronomy (Ast), Biology (Bio), Chemistry (Che), Computer Science (CS), Earth Science (ES), Engineering (Eng), Materials Science (MS), Mathematics (Mat), and Medicine (Med). It contains annotated mentions of scientific concepts in abstracts with four domain-independent concept types, namely *Process*, *Method*, *Material*, and *Data*. These concept mentions were later linked to entities in Wikipedia and Wikidata [16]. The 110 articles (11 per domain) were taken from the OA-STM corpus [17] of Elsevier Labs.

We build upon related work and extend the STM corpus with coreference annotations. In particular, we (1) annotate coreference links between existing scientific concept mentions in abstracts using the BRAT annotation tool [46], and (2) annotate further mentions, i.e. pronouns and noun phrases consisting of multiple consecutive mentions.

Annotation Process: Other studies have shown that non-expert annotations are viable for the scientific domain [6,8,19,44,47], and they are less costly than domain-expert annotations. Therefore, we also annotate the corpus with non-domain experts, i.e. by two students in computer science. Furthermore, we follow mostly the annotation procedure of the STM corpus [6], which consists of the following three phases:

Table 1. Per-domain and overall inter-annotator agreement (Cohen's κ and MUC) for coreference resolution annotation in our STM corpus.

	Mat	*Med*	*Ast*	*CS*	*Bio*	*Agr*	*ES*	*Eng*	*Che*	*MS*	*Overall*
κ	0.84	0.80	0.78	0.72	0.70	0.66	0.61	0.58	0.56	0.52	0.68
MUC	0.83	0.69	0.78	0.73	0.70	0.72	0.61	0.66	0.56	0.63	0.69

Table 2. Characteristics of the annotated STM corpus with 110 abstracts per concept type in terms of number of scientific concept mentions, number of coreferent mentions, number of coreference clusters and singleton clusters, and the number of overall clusters. MIXED denotes clusters consisting of mentions with different concept types, NONE denotes coreference mentions and clusters without a scientific concept mention.

	Data	Material	Method	Process	MIXED	NONE	Total
# mentions	1,658	2,099	258	2,112	0	0	6,127
# coreferent mentions	351	910	101	510	0	705	2,577
# coreference clusters	153	339	30	198	50	138	908
# singleton clusters	1,307	1,189	157	1,602	0	0	4,255
# overall clusters	1,460	1,528	187	1,800	50	138	5,163

Table 3. Characteristics of the STM corpus per domain (11 abstracts per domain).

	Agr	Ast	Bio	Che	CS	ES	Eng	MS	Mat	Med	Total
# mentions	741	791	649	553	483	698	741	574	297	600	6,127
# coreferent mentions	276	365	275	282	181	241	318	256	124	259	2,577
# coreference clusters	106	120	98	90	67	93	117	87	48	82	908
# singleton clusters	520	549	443	384	339	525	503	371	210	411	4,255
# clusters	626	669	541	474	406	618	620	458	258	493	5,163

1. *Pre-annotation:* This phase aims at developing annotation guidelines through trial annotations. We adapted the comprehensive annotation guidelines of the OntoNotes 5.0 dataset [38], which were developed for the general domain, to research papers. In particular, we provide briefer and simpler descriptions with examples from the scientific domain. Within three iterations both annotators labelled independently 10, 9 and 7 abstracts (i.e. 26 abstracts), respectively. After each iteration the annotators discussed the outcome and refined the annotation guidelines.
2. *Independent Annotation:* After the annotation guidelines were finalised, both annotators independently re-annotated the previously annotated abstracts and 24 additional abstracts. The final inter-coder agreement was measured on the 50 abstracts (5 per domain) using Cohen's κ [10,24] and MUC [48]. As shown in Table 1, we achieve a substantial agreement with 0.68 κ and 0.69 MUC.
3. *Consolidation:* Finally, the remaining 60 abstracts were annotated by one annotator and the annotation results of this author were used as the gold standard corpus.

Corpus Characterstics: Table 2 shows the characteristics of the resulting corpus broken down per concept type, while they are listed per domain in Table 3. The original corpus has in total 6,127 mentions. 2,577 mentions were annotated as coreferent resulting in 908 coreference clusters. Thus, each coreference cluster contains on average 2.84 mentions, while *Method* clusters contain the most (3.4 mentions) and *Data* clusters the least (2.3 mentions). Furthermore, 705 mentions were annotated additionally (referred to as NONE) since they represent pronouns (422 mentions) or noun phrases consisting of multiple consecutive original mentions (283 mentions) such as *'... [[A], [B], and [C] [treatments]]... [These treatments]...'*. Fifty clusters (5%) contain mentions with different concept types (referred to as MIXED) due to disagreements between the annotators of the original concept mentions, and the annotators of coreferences. For instance, non-coreferent mentions were annotated as coreferent, or coreferent mentions have different concept types. Finally, 138 clusters (15%) do not have a concept type (NONE) since they form clusters which are not coreferent with the original concept mentions.

3.2 Transfer Learning for Coreference Resolution

We suggest sequential transfer learning [42] for coreference resolution in research papers. Therefore, we fine-tune a model pre-trained on a large (source) dataset to our (target) dataset. As the source dataset, we use the English portion of the OntoNotes 5.0 dataset [37], since it is a broad corpus that consists of 3,493 documents with telephone conversations, magazine and news articles, web data, broadcast conversations, and the New Testament. Besides, our annotation guidelines were adapted from OntoNotes 5.0.

For the model, we utilise *BERT for Coreference Resolution (BFCR)* [22] with *SpanBERT* [21] word embeddings. This model achieves state-of-the-art results on the OntoNotes dataset [21]. Another advantage is the availability of the pre-trained model and the source code. The BFCR model improves Lee et al.'s approach [26] by replacing the LSTM encoder with the SpanBERT transformer-encoder. SpanBERT [21] has different training objectives than BERT [14] to better represent spans of text.

3.3 Cross-Domain Research Knowledge Graph Population

Let $d \in D$ be an abstract, $M(d) = \{m_1, ..., m_h\}$ the mentions of scientific concepts in d, and $c_d(m_i) \subseteq M(d)$ the corresponding coreference cluster for mention m_i in d. If mention m_s is not coreferent with other mentions in d, then $c_d(m_s) = \{m_s\}$ is a singleton cluster. The set of all clusters is denoted by C. An equivalence relation *collapsable* $\subseteq C \times C$ defines if two clusters can be collapsed, i.e. if the clusters refer to the same scientific concept. To create the set of all concepts E, we build the quotient set for the set of clusters C with respect to the relation *collapsable*:

$$C := \{c_d(m) | d \in D, m \in M(d)\} \tag{1}$$

$$[c] := \{x \in C | collapsable(c, x)\} \tag{2}$$

$$E := \{[c] | c \in C\} \tag{3}$$

Now, we can construct the KG: for each paper $d \in D$ and for each scientific concept $e \in E$ we create a node in the KG. The scientific concept type of e is the most frequent concept type of all mentions in e. Then, for each mention $m \in M(d)$ we create a 'mentions' link between the paper and the corresponding scientific concept $[m] \in E$.

Cross-Domain vs. In-Domain Collapsing: One commonly used approach to define the *collapsable* relation is to treat two clusters as equivalent, if and only if the 'label' of the clusters is the same. The label of a cluster is the longest mention in the cluster normalised by (a) lower-casing, (b) removing articles, possessives and demonstratives, (c) resolving acronyms, and (d) lemmatisation using WordNet [18] to transform plural forms to singular. Other studies [13, 27] used a similar label function for KG population.

However, a research KG that comprises multiple scientific disciplines has not been populated yet. Thus, it is not clear whether it is feasible to collapse clusters across domains. Usually, terms within a scientific domain are unambiguous, but some terms can have different meanings across scientific disciplines (e.g. "neural network" in *CS* and *Med*). Thus, we investigate both cross-domain and in-domain collapsing strategies.

Knowledge Graph Population Approach: We populate a research KG with research papers from multiple scientific domains, i.e. 55,485 abstracts of Elsevier with CC-BY licence from the 10 investigated domains. First, we extract (a) concept mentions from the abstracts using the scientific concept extractor of the STM-corpus [6], and (b) clusters within the abstracts with our transfer learning coreference model. Then, those mention clusters, which contain solely mentions recognised by the coreference resolution model and not by the scientific concept extraction model, are dropped, since the coreference resolution model does not recognise the concept type of the mentions. Finally, the remaining clusters serve for the population of the KG as described above.

3.4 Evaluation Procedure of Clustering in KG Population

One common approach to evaluate the quality of a populated KG is to annotate a (random) subset of statements by humans as true or false and to calculate precision and recall [13,50]. To evaluate recall, small collections of ground-truth capturing *all* knowledge is necessary, that are usually difficult to obtain [50]. To the best of our knowledge, a common approach to evaluate the clustering aspect of the KG population pipeline does not exist yet. Thus, in the following, we present (1) an annotated test KG, and (2) metrics to evaluate clustering of mentions to concepts in KG population.

Test KG: To enable evaluation of KG population strategies, we compile a test KG, referred to as *Test-STM-KG*. For this purpose, we reuse the STEM-ECR corpus [16], in which 1,221 mentions of the STM corpus are linked to Wikipedia entities. First, we extract all annotated clusters of the STM corpus in which all mentions of the cluster uniquely refer to the same Wikipedia entity. Then, we collapse all clusters which refer to the same Wikipedia entity to concepts. Formally, the Test-STM-KG is a partition of mentions, where each part denotes a concept, i.e. a disjoint set of mentions. A mention is uniquely represented by the tuple (start offset, end offset, concept type, doc id).

Table 4 shows the characteristics of the compiled Test-STM-KG. It consists of 920 clusters, of which 711 are singleton clusters. These clusters were collapsed to 762 concepts, of which 31 concepts are used across multiple domains (referred to as MIX).

Evaluation Procedure: To evaluate the clustering result of a KG population strategy, we use the metrics of coreference resolution. The three popular metrics for coreference resolution are MUC [48], B^3 [3] and $CEAFe_{\phi 4}$ [29]. Each of them

Table 4. Characteristics of the *Test-STM-KG*: number of concepts per concept type and per domain. MIX denotes the number of cross-domain concepts.

	Agr	Ast	Bio	CS	Che	ES	Eng	MS	Mat	Med	MIX	Total
Data	5	18	3	20	4	9	28	13	37	8	9	154
Material	27	35	30	20	26	52	32	30	9	40	7	308
Method	1	1	1	21	6	2	4	10	3	8	7	64
Process	17	12	21	34	13	33	20	25	15	38	8	236
Total	50	66	55	95	49	96	84	78	64	94	31	762

represents different evaluation aspects (see [36] for more details). To calculate these metrics, we treat the gold concepts (i.e. a partition of mentions) of the Test-STM-KG as the 'key' and the predicted concepts as the 'response'. We report also the *CoNLL P/R/F1* scores, that is the averages of MUC's, B^3's and $CEAFe_{\phi 4}$'s respective precision (P), recall (R) and F1 scores. The CoNLL metrics were proposed for the conference on Computational Natural Language Learning (CoNLL) shared tasks on coreference resolution [36].

4 Experimental Setup

Here we describe our experimental setup for coreference resolution and KG population.

4.1 Automatic Coreference Resolution

We evaluate three different state-of-the-art architectures on the STM dataset: (I) *BERT for Coreference Resolution (BFCR)* [22] with *SpanBERT* [21] word embeddings (referred to as *BFCR_Span*), (II) BFCR with *SciBERT* [4] word embeddings (referred to as *BFCR_Sci*), and (III) *Scientific Information Extractor (SCIIE)* [27] with ELMo [35] word embeddings (referred to as *SCIIE*). The three architectures are evaluated in the following six approaches (#1–#6):

- *Pre-Trained Models:* We evaluate already pre-trained models on the test sets of the STM corpus, i.e. #1 *BFCR_Span* trained on the English portion of the OntoNotes dataset [38], and #2 *SCIIE* trained on SciERC [27] from the AI domain.
- *Supervised Learning:* We train a model from scratch with the three architectures using the training data of the STM corpus and evaluate their performance with the test sets of STM: #3 *BFCR_Span*, #4 *BFCR_Sci*, and #5 *SCIIE*.
- *Transfer Learning:* This is our proposed approach #6. We fine-tune all parameters of a pre-trained model on the English portion of the OntoNotes dataset [21] with the training data of our STM corpus. For that, we use the *BFCR_Span* architecture.

Table 5. Performance of the baseline approaches #1–#5 and our proposed transfer learning approach #6 on the test sets of the STM corpus across five-fold cross validation.

			MUC			B^3			$CEAFe_{\phi 4}$			$CoNLL$		
		Training data	P	R	F1	P	R	F1	P	R	F1	P	R	F1
#1	BFCR_Span	OntoNotes	57.1	31.1	40.2	55.9	25.7	35.2	50.2	28.1	36.0	54.4	28.3	37.1
#2	SCIIE	SciERC	13.4	4.5	6.8	13.1	4.3	6.5	18.1	6.0	9.0	14.9	4.9	7.4
#3	BFCR_Span	STM	61.6	45.6	52.3	59.8	41.5	48.8	57.9	44.4	50.0	59.8	43.8	50.4
#4	BFCR_Sci	STM	61.9	40.2	48.6	59.7	36.1	44.9	61.7	36.9	46.0	61.1	37.7	46.5
#5	SCIIE	STM	60.3	45.2	51.6	57.6	41.7	48.3	56.6	43.6	49.1	58.1	43.5	49.7
#6	**BFCR_Span**	**Onto→STM**	**64.5**	**63.5**	**63.9**	**61.0**	**60.0**	**60.4**	**60.5**	**59.6**	**60.0**	**62.0**	**61.0**	**61.4**

Table 6. Per domain and overall CoNLL F1 results of the best baseline #3 and our transfer learning approach #6 on the STM corpus across five-fold cross validation.

		Training data	Agr	Ast	Bio	Che	CS	ES	Eng	MS	Mat	Med	Overall
#3	BFCR_Span	STM	48.0	50.5	52.2	49.0	59.1	39.6	52.8	47.6	42.5	51.0	50.4
#6	**BFCR_Span**	**Onto→STM**	**62.8**	**61.1**	**57.5**	**56.3**	**74.9**	**57.5**	**59.8**	**52.1**	**55.7**	**62.1**	**61.4**

Evaluation: We use the metrics MUC [48], B^3 [3], $CEAFe_{\phi 4}$ [29] and $CoNLL$ [36] in compliance with other studies on coreference resolution [22,25,30]. To obtain robust results, we apply five-fold cross-validation, according to the data splits given by Brack et al. [6], and report averaged results. For each fold, the dataset is split into train/validation/test sets with 8/1/2 abstracts per domain, respectively, i.e. 80/10/20 abstracts. We reuse the original implementations and default hyperparameters of the above architectures. Hyperparameter-tuning of the best baseline approach #3 according to [22] confirmed that the default hyperparameters of *BFCR_Span* perform best on our corpus.

4.2 Evaluation of KG Population Strategies

We compare four KG population strategies: (1) cross-domain and (2) in-domain collapsing, as well as (3) cross-domain and (4) in-domain collapsing without coreference resolution. To evaluate cross-domain and in-domain collapsing, we take the gold clusters (i.e. mention clusters within the abstracts) of the Test-STM-KG and collapse them to concepts according to the respective strategy. When leaving out the coreference resolution step, we treat all mentions in the Test-STM-KG as singleton clusters and collapse them to concepts according to the respective strategy. Finally, we calculate the metrics as described in Sect. 3.4.

5 Results and Discussion

In this section, we discuss the experimental results for automatic coreference resolution and KG population.

5.1 Automatic Coreference Resolution

Table 5 shows the overall results of the six evaluated approaches and Table 6 the results per domain of the best baseline #3 and our approach #6. Our transfer learning approach #6 *BFCR_Span* from OntoNotes (Onto) [37] to STM significantly outperforms the best baseline approach #3 with an overall CoNLL F1 of 61.4 (+10.0) and a low standard deviation ±1.5 across the five folds.

Table 7. CoNLL scores on the tests sets of the SciERC corpus [27] across 3 random restarts of the approaches: current state of the art of Luan et al., the best baseline approach (#3), and our transfer learning approach (#6). We report results using the whole and using only $\frac{1}{5}$th of the training data of SciERC (referred to as $\frac{1}{5}$SciERC).

		Training data	P	R	F1
Luan et al. [27]		SciERC	52.0	44.9	48.2
#3	BFCR_Span	SciERC	63.3	55.7	59.3
#6	**BFCR_Span**	**OntoNotes→SciERC**	**63.9**	**57.1**	**60.1**
#3	BFCR_Span	$\frac{1}{5}$SciERC	63.1	39.1	47.1
#6	**BFCR_Span**	**OntoNotes→ $\frac{1}{5}$SciERC**	**52.8**	**56.7**	**54.2**

The approaches #1 *BFCR_Span* pre-trained on OntoNotes [37], and #2 *SCIIE* pre-trained on SciERC [27] achieve a CoNLL F1 score of 37.1 and 7.4, respectively. These scores are quite low compared to the approaches #3–#6 that use training data of the STM corpus. This indicates that models pre-trained on existing datasets do not generalise sufficiently well for coreference resolution in research papers. Models trained only on the STM corpus (i.e. #3–#5) achieve better results. However, they have quite low recall scores indicating that the size of the training data might not be sufficient to enable the model to generalise well. SciBERT #4, although pre-trained on scientific texts, performs worse than SpanBERT #3. Presumably the reason is that SpanBERT has approximately 3 times more parameters than SciBERT. Our transfer learning approach #6 achieves the best results with quite balanced precision and recall scores.

Furthermore, to evaluate the effectiveness of our transfer learning approach, we compare the best baseline #3 and our transfer learning approach #6 also with the SciERC corpus [27]. The SciERC corpus comprises 500 abstracts from the AI domain. Since SciERC has around 5 times more training data than STM, we compare the approaches #3 and #6 also using only $\frac{1}{5}$th of the training data in SciERC while keeping the original validation and test sets. It can be seen in Table 7 that our transfer learning approach #6 improves slightly the baseline result using the whole training data with 60.1 F1 (+0.8). When using only $\frac{1}{5}$th of the training data, our transfer learning approach noticeably outperforms the baseline with 54.2 F1 (+7.1). Thus, our transfer learning approach can help significantly to improve the performance of coreference resolution in research papers with few labelled data.

5.2 Cross-Domain Research KG

In this subsection, we describe the characteristics of our populated KG and discuss the evaluation results of various KG population strategies.

Characteristics of the Research KG: Table 8 shows the characteristics of the populated KGs per domain. The resulting KGs with cross-domain and in-domain collapsing have more than 994,000 and 1.1 Mio. scientific concepts, respectively, obtained from 55,485 abstracts with more than 2,1 Mio. concept mentions and 726,000 coreferent mentions. *Ast* and *Bio* are the most represented domains, while *CS* and *Mat* are the most underrepresented.

Table 8. Characteristics of the populated research KGs per domain: (1) number of abstracts, number of extracted scientific concept mentions and coreferent mentions, (2) the number of scientific concepts for the KG with cross-domain collapsing, (3) in-domain collapsing, (4) cross-domain collapsing but without coreference resolution, and (5) in-domain collapsing but without coreference resolution. Reduction denotes the percentual reduction of mentions to scientific concepts and MIX the cross-domain concepts.

	Agr	Ast	Bio	CS	Che	ES	Eng	MS	Mat	Med	MIX	Total
# abstracts	7,731	15,053	11,109	1,216	1,234	2,352	3,049	2,258	665	10,818	–	55,485
# mentions	332,983	370,311	423,315	45,388	46,203	129,288	127,985	86,490	20,466	586,019	–	2,168,448
# coref. men.	108,579	120,942	143,292	17,674	14,059	40,974	42,654	25,820	8,510	203,884	–	726,388
Cross-domain collapsing												
KG concepts	138,342	173,027	177,043	20,474	21,298	62,674	55,494	39,211	9,275	227,690	70,044	994,572
- Data	27,132	64,537	32,946	5,380	5,124	19,542	17,053	10,629	2,982	66,473	19,715	271,513
- Material	69,534	45,296	83,627	6,242	10,154	24,322	19,689	17,276	2,406	68,141	20,812	367,499
- Method	2,992	8,819	6,135	2,001	1,055	1,776	2,953	1,605	685	9,363	1,627	39,011
- Process	38,684	54,375	54,335	6,851	4,965	17,034	15,799	9,701	3,202	83,713	27,890	316,549
Reduction	58%	53%	58%	55%	54%	52%	57%	55%	55%	61%	–	54%
In-domain collapsing												
KG concepts	180,135	197,605	229,201	30,736	32,191	81,584	78,417	55,358	14,567	278,686	–	1,178,480
Reduction	46%	47%	46%	32%	30%	37%	39%	36%	29%	52%	–	46%
Cross-domain collapsing without coreference resolution												
KG concepts	146,894	182,479	187,557	21,950	22,555	66,600	59,689	41,776	9,939	242,797	77,493	1,059,729
Reduction	56%	51%	56%	52%	51%	48%	53%	52%	51%	59%	–	51%
In-domain collapsing without coreference resolution												
KG concepts	184,218	199,894	234,399	31,525	32,937	83,445	80,476	56,690	14,911	284,547	–	1,203,042
Reduction	45%	46%	45%	31%	29%	35%	37%	34%	27%	51%	–	45%

Evaluation of KG Population Strategies: Next, we discuss the different KG population strategies. For each strategy, Table 8 reports the number of concepts in the populated KG and the percentage reduction of mentions to concepts, and in Table 9 the evaluation results of KGs against the Test-STM-KG.

Cross-Domain vs. In-Domain Collapsing: Cross-domain collapsing achieves a higher CoNLL F1 score of 64.8 than in-domain collapsing with a score of 63.5

(see Table 9). However, in-domain collapsing yields (as expected) a higher precision (CoNLL P 85.5), since some terms have different meanings across domains (e.g. *Measure_(mathematics)* vs. *Measurement* in https://en.wikipedia.org). Furthermore, the Test-STM-KG has only 31 cross-domain concepts due to its small size. Thus, we expect that cross-domain collapsing would yield worse results on a larger test set.

Furthermore, as shown in Table 8, cross-domain collapsing yields less concepts than in-domain collapsing (more than 994,000 versus 1.1 Mio. concepts). We can also observe that only 70,044 (7%) of the concepts are used across multiple domains. This indicates that every scientific domain mostly uses its own terminology. However, the concepts used across domains can have different meanings. Thus, when precision is more important than recall in downstream tasks, in-domain collapsing should be the preferred choice.

Effect of Coreference Resolution: Coreference resolution has only a small impact on the number of resulting concepts in a populated KG (see Table 8). However, as shown in Table 9, leaving out the coreference resolution step during KG population yields only low CoNLL F1 scores, i.e. 41.7 (−21.8) F1 and 43.5 (−21.3) F1. Thus, coreference resolution significantly improves the quality of a populated KG .

Table 9. Performance of the collapsing strategies evaluated against the *Test-STM-KG*: in-domain and cross-domain collapsing with and without coreference resolution.

	#concepts	*MUC*			B^3			$CEAFe_{\phi 4}$			*CoNLL*		
	in KG	P	R	F1	P	R	F	P	R	F1	P	R	F1
In-domain collapsing	859	**86.3**	70.6	77.7	**86.0**	69.0	76.6	84.1	23.1	36.2	**85.5**	54.2	63.5
- Without coreferences	900	75.5	38.8	51.2	75.2	37.9	50.4	71.1	14.0	23.4	73.9	30.2	41.7
Cross-domain collapsing	837	85.0	**73.0**	**78.5**	84.5	**72.1**	**77.8**	**84.7**	**24.6**	**38.1**	84.7	**56.6**	**64.8**
- Without coreferences	876	73.5	41.0	52.6	72.2	15.5	25.5	72.2	15.5	25.5	73.0	32.4	43.5

Qualitative Analysis: We also inspected the top-five frequent domain-specific concepts in the populated KG (a list of these concepts can be found in our public repository). As far as we can judge with our computer science background, we consider the extracted top frequent concepts to be reasonable and useful for the domains. For instance, in *Ast*, the method ‘standard model’ is frequently mentioned, while in *CS* the process ‘cyber attack’ appears most often. The frequency of the top concepts differs significantly between the domains: In *Med*, *Ast*, *Eng*, *ES* and *Agr*, a top frequent concept is referenced 10.8, 10.2, 4.9, 3.8, and 3.1 times per 1000 abstracts, respectively. In *Che*, *MS*, *Mat*, *Bio*, and *CS*, a top frequent concept is referenced only by few abstracts (0.3, 0.4, 1.0, 1.4, and 2.3, respectively, per 1000 abstracts).

6 Conclusions

In this paper, we have investigated the task of coreference resolution in research papers across 10 different scientific disciplines. We have annotated a corpus that comprises 110 abstracts with coreferences with a substantial inter-coder agreement. Our baseline results with current state-of-the-art approaches for coreference resolution demonstrate that current approaches perform poorly on our corpus. The proposed approach, which uses sequential transfer learning and exploits annotated datasets from the general domain, outperforms noticeably the state-of-the-art baselines. Thus, our transfer learning approach can help to reduce annotation costs for scientific papers, while obtaining high-quality results at the same time.

Furthermore, we have investigated the impact of coreference resolution on KG population. For this purpose, we have compiled a gold KG from our annotated corpus and propose an evaluation procedure for KG population strategies. We have demonstrated that coreference resolution has a small impact on the number of resulting concepts in the KG, but improved significantly its quality. Finally, we have generated a research KG from 55,485 abstracts of the 10 investigated domains. We show that every domain mostly uses its own terminology and that the populated KG contains useful concepts.

In future work, we plan to evaluate multi-task learning approaches, and to populate and evaluate a much larger research KG to get more insights in scientific language use.

References

1. Seventh Message Understanding Conference: Proceedings of a Conference Held in Fairfax, Virginia, USA, MUC 1998, 29 April–1 May 1998. ACL (1998). https://www.aclweb.org/anthology/volumes/M98-1/
2. Augenstein, I., Das, M., Riedel, S., Vikraman, L., McCallum, A.: Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications. In: Bethard, S., Carpuat, M., Apidianaki, M., Mohammad, S.M., Cer, D.M., Jurgens, D. (eds.) Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, 3–4 Aug 2017, pp. 546–555. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/S17-2091
3. Bagga, A., Baldwin, B.: Algorithms for scoring coreference chains. In: The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference, pp. 563–566 (1998)
4. Beltagy, I., Lo, K., Cohan, A.: SciBERT: a pretrained language model for scientific text. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 Nov 2019, pp. 3613–3618. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1371
5. Bornmann, L., Mutz, R.: Growth rates of modern science: a bibliometric analysis based on the number of publications and cited references. J. Assoc. Inf. Sci. Technol. **66**(11), 2215–2222 (2015). https://doi.org/10.1002/asi.23329
6. Brack, A., D'Souza, J., Hoppe, A., Auer, S., Ewerth, R.: Domain-independent extraction of scientific concepts from research articles. In: Jose, J.M., et al. (eds.) ECIR 2020, Part I. LNCS, vol. 12035, pp. 251–266. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45439-5_17

7. Chaimongkol, P., Aizawa, A., Tateisi, Y.: Corpus for coreference resolution on scientific papers. In: Calzolari, N., et al. (eds.) Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, 26–31 May 2014, pp. 3187–3190. European Language Resources Association (ELRA) (2014). http://www.lrec-conf.org/proceedings/lrec2014/summaries/286.html
8. Chambers, A.: Statistical Models for Text Classification and Clustering: Applications and Analysis. Ph.D. thesis, University of California, Irvine (2013)
9. Clark, K., Manning, C.D.: Entity-centric coreference resolution with model stacking. In: ACL (1), pp. 1405–1415. The Association for Computer Linguistics (2015)
10. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Measur. **20**(1), 37–46 (1960)
11. Cohen, K.B., et al.: Coreference annotation and resolution in the colorado richly annotated full text (CRAFT) corpus of biomedical journal articles. BMC Bioinform. **18**(1), 372:1–372:14 (2017). https://doi.org/10.1186/s12859-017-1775-9
12. Denis, P., Baldridge, J.: Specialized models and ranking for coreference resolution. In: 2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25–27 Oct 2008, Honolulu, Hawaii, USA. A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 660–669. ACL (2008). https://www.aclweb.org/anthology/D08-1069/
13. Dessì, D., Osborne, F., Reforgiato Recupero, D., Buscaldi, D., Motta, E., Sack, H.: AI-KG: an automatically generated knowledge graph of artificial intelligence. In: Pan, J.Z., et al. (eds.) ISWC 2020, Part II. LNCS, vol. 12507, pp. 127–143. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62466-8_9
14. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, vol. 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/n19-1423
15. Doddington, G.R., Mitchell, A., Przybocki, M.A., Ramshaw, L.A., Strassel, S.M., Weischedel, R.M.: The automatic content extraction (ACE) program - tasks, data, and evaluation. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, 26–28 May 2004, Lisbon, Portugal. European Language Resources Association (2004). http://www.lrec-conf.org/proceedings/lrec2004/summaries/5.htm
16. D'Souza, J., Hoppe, A., Brack, A., Jaradeh, M.Y., Auer, S., Ewerth, R.: The STEM-ECR dataset: grounding scientific entity references in STEM scholarly content to authoritative encyclopedic and lexicographic sources. In: Calzolari, N., et al. (eds.) Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, 11–16 May 2020, pp. 2192–2203. European Language Resources Association (2020), https://www.aclweb.org/anthology/2020.lrec-1.268/
17. Elsevier Labs: Elsevier OA STM corpus. https://github.com/elsevierlabs/OA-STM-Corpus (2017). Accessed 15 July 2020
18. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. Language, Speech, and Communication. MIT Press, Cambridge (1998)

19. Fisas, B., Saggion, H., Ronzano, F.: On the discoursive structure of computer graphics research papers. In: Meyers, A., Rehbein, I., Zinsmeister, H. (eds.) Proceedings of The 9th Linguistic Annotation Workshop, LAW@NAACL-HLT 2015, 5 June 2015, Denver, Colorado, USA, pp. 42–51. The Association for Computer Linguistics (2015). https://doi.org/10.3115/v1/w15-1605
20. Gábor, K., Buscaldi, D., Schumann, A., QasemiZadeh, B., Zargayouna, H., Charnois, T.: Semeval-2018 task 7: semantic relation extraction and classification in scientific papers. In: Apidianaki, M., Mohammad, S.M., May, J., Shutova, E., Bethard, S., Carpuat, M. (eds.) Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2018, New Orleans, Louisiana, USA, 5–6 June 2018, pp. 679–688. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/s18-1111
21. Joshi, M., Chen, D., Liu, Y., Weld, D.S., Zettlemoyer, L., Levy, O.: SpanBERT: improving pre-training by representing and predicting spans. Trans. Assoc. Comput. Linguistics **8**, 64–77 (2020). https://transacl.org/ojs/index.php/tacl/article/view/1853
22. Joshi, M., Levy, O., Zettlemoyer, L., Weld, D.S.: BERT for coreference resolution: Baselines and analysis. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 Nov 2019, pp. 5802–5807. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1588
23. Kim, J., Nguyen, N.L.T., Wang, Y., Tsujii, J., Takagi, T., Yonezawa, A.: The genia event and protein coreference tasks of the BioNLP shared task 2011. BMC Bioinform. **13**(S–11), S1 (2012). https://doi.org/10.1186/1471-2105-13-S11-S1
24. Kopeć, M., Ogrodniczuk, M.: Inter-annotator agreement in coreference annotation of polish. In: Sobecki, J., Boonjing, V., Chittayasothorn, S. (eds.) Advanced Approaches to Intelligent Information and Database Systems. SCI, vol. 551, pp. 149–158. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-05503-9_15
25. Lee, K., He, L., Lewis, M., Zettlemoyer, L.: End-to-end neural coreference resolution. In: Palmer, M., Hwa, R., Riedel, S. (eds.) Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 Sept 2017, pp. 188–197. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/d17-1018
26. Lee, K., He, L., Zettlemoyer, L.: Higher-order coreference resolution with coarse-to-fine inference. In: Walker, M.A., Ji, H., Stent, A. (eds.) Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, 1–6 June 2018, vol. 2 (Short Papers), pp. 687–692. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/n18-2108
27. Luan, Y., He, L., Ostendorf, M., Hajishirzi, H.: Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 Oct – 4 Nov 2018, pp. 3219–3232. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1360
28. Lubani, M., Noah, S.A.M., Mahmud, R.: Ontology population: approaches and design aspects. J. Inf. Sci. **45**(4), 502–515 (2019). https://doi.org/10.1177/0165551518801819

29. Luo, X.: On coreference resolution performance metrics. In: HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6–8 Oct 2005, Vancouver, British Columbia, Canada, pp. 25–32. The Association for Computational Linguistics (2005). https://www.aclweb.org/anthology/H05-1004/
30. Ma, J., et al.: Jointly optimized neural coreference resolution with mutual attention. In: Caverlee, J., Hu, X.B., Lalmas, M., Wang, W. (eds.) WSDM 2020: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, 3–7 Feb 2020, pp. 402–410. ACM (2020). https://doi.org/10.1145/3336191.
31. Marasovic, A., Born, L., Opitz, J., Frank, A.: A mention-ranking model for abstract anaphora resolution. In: Palmer, M., Hwa, R., Riedel, S. (eds.) Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 Sept 2017, pp. 221–232. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/d17-1021
32. Ng, V.: Machine learning for entity coreference resolution: a retrospective look at two decades of research. In: Singh, S.P., Markovitch, S. (eds.) Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 4–9 Feb 2017, San Francisco, California, USA, pp. 4877–4884. AAAI Press (2017). http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14995
33. Ng, V., Cardie, C.: Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In: 19th International Conference on Computational Linguistics, COLING 2002, Howard International House and Academia Sinica, Taipei, Taiwan, 24 Aug – 1 Sept 2002 (2002). https://www.aclweb.org/anthology/C02-1139/
34. Nguyen, N.L.T., Kim, J., Miwa, M., Matsuzaki, T., Tsujii, J.: Improving protein coreference resolution by simple semantic classification. BMC Bioinform. **13**, 304 (2012). https://doi.org/10.1186/1471-2105-13-304
35. Peters, M.E., et al.: Deep contextualized word representations. In: Walker, M.A., Ji, H., Stent, A. (eds.) Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, 1–6 June 2018, vol. 1 (Long Papers), pp. 2227–2237. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/n18-1202
36. Pradhan, S., Luo, X., Recasens, M., Hovy, E.H., Ng, V., Strube, M.: Scoring coreference partitions of predicted mentions: a reference implementation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, 22–27 June 2014, Baltimore, MD, USA, vol. 2: Short Papers, pp. 30–35. The Association for Computer Linguistics (2014). https://doi.org/10.3115/v1/p14-2006
37. Pradhan, S., et al.: Towards robust linguistic analysis using ontonotes. In: Hockenmaier, J., Riedel, S. (eds.) Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, 8–9 Aug 2013, pp. 143–152. ACL (2013). https://www.aclweb.org/anthology/W13-3516/

38. Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., Zhang, Y.: Conll-2012 shared task: modeling multilingual unrestricted coreference in OntoNotes. In: Pradhan, S., Moschitti, A., Xue, N. (eds.) Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning - Proceedings of the Shared Task: Modeling Multilingual Unrestricted Coreference in OntoNotes, EMNLP-CoNLL 2012, 13 July 2012, Jeju Island, Korea, pp. 1–40. ACL (2012). https://www.aclweb.org/anthology/W12-4501/
39. Pujara, J., Singh, S.: Mining knowledge graphs from text. In: Chang, Y., Zhai, C., Liu, Y., Maarek, Y. (eds.) Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, 5–9 Feb 2018, pp. 789–790. ACM (2018). https://doi.org/10.1145/3159652.3162011
40. Q. Zadeh, B., Handschuh, S.: The ACL RD-TEC: a dataset for benchmarking terminology extraction and classification in computational linguistics. In: Proceedings of the 4th International Workshop on Computational Terminology (Computerm), pp. 52–63. Association for Computational Linguistics and Dublin City University, Dublin, Ireland (Aug 2014). https://doi.org/10.3115/v1/W14-4807
41. ur Rahman, M.A., Ng, V.: Supervised models for coreference resolution. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6–7 Aug 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 968–977. ACL (2009). https://www.aclweb.org/anthology/D09-1101/
42. Ruder, S.: Neural Transfer Learning for Natural Language Processing. Ph.D. thesis, National University of Ireland, Galway (2019)
43. Sanh, V., Wolf, T., Ruder, S.: A hierarchical multi-task approach for learning embeddings from semantic tasks. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, 27 Jan – 1 Feb 2019, pp. 6949–6956. AAAI Press (2019). https://doi.org/10.1609/aaai.v33i01.33016949
44. Schäfer, U., Spurk, C., Steffen, J.: A fully coreference-annotated corpus of scholarly papers from the ACL anthology. In: Kay, M., Boitet, C. (eds.) COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Posters, 8–15 Dec 2012, Mumbai, India, pp. 1059–1070. Indian Institute of Technology Bombay (2012). https://www.aclweb.org/anthology/C12-2103/
45. Soon, W.M., Ng, H.T., Lim, C.Y.: A machine learning approach to coreference resolution of noun phrases. Comput. Linguist. **27**(4), 521–544 (2001). https://doi.org/10.1162/089120101753342653
46. Stenetorp, P., et al.: BRAT: a web-based tool for NLP-assisted text annotation. In: Daelemans, W., Lapata, M., Màrquez, L. (eds.) EACL 2012, 13th Conference of the European Chapter of the Association for Computational Linguistics, Avignon, France, 23–27 Apr 2012, pp. 102–107. The Association for Computer Linguistics (2012). https://www.aclweb.org/anthology/E12-2021/
47. Teufel, S., Siddharthan, A., Batchelor, C.R.: Towards domain-independent argumentative zoning: evidence from chemistry and computational linguistics. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6–7 Aug 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 1493–1502. ACL (2009). https://www.aclweb.org/anthology/D09-1155/

48. Vilain, M.B., Burger, J.D., Aberdeen, J.S., Connolly, D., Hirschman, L.: A model-theoretic coreference scoring scheme. In: Proceedings of the 6th Conference on Message Understanding, MUC 1995, Columbia, Maryland, USA, 6–8 Nov 1995, pp. 45–52. ACL (1995). https://doi.org/10.3115/1072399.1072405
49. Wadden, D., Wennberg, U., Luan, Y., Hajishirzi, H.: Entity, relation, and event extraction with contextualized span representations. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 Nov 2019, pp. 5783–5788. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1585
50. Weikum, G., Dong, L., Razniewski, S., Suchanek, F.M.: Machine knowledge: Creation and curation of comprehensive knowledge bases. CoRR abs/2009.11564 (2020). https://arxiv.org/abs/2009.11564

How Do Simple Transformations of Text and Image Features Impact Cosine-Based Semantic Match?

Guillem Collell(✉) and Marie-Francine Moens

Department of Computer Science, KU Leuven, 3001 Heverlee, Belgium
{gcollell,sien.moens}@kuleuven.be2

Abstract. Practitioners often resort to off-the-shelf feature extractors such as language models (e.g., BERT or Glove) for text or pre-trained CNNs for images. These features are often used without further supervision in tasks such as text or image retrieval and semantic similarity with cosine-based semantic match. Although cosine similarity is sensitive to centering and other feature transforms, their impact on task performance has not been systematically studied. Prior studies are limited to a single domain (e.g., bilingual embeddings) and one data modality (text). Here, we systematically study the effect of simple feature transforms (e.g., standardizing) in 25 datasets with 6 tasks covering semantic similarity and text and image retrieval. We further back up our claims in *ad-hoc* laboratory experiments. We include 15 (8 image + 7 text) embeddings, covering the state-of-the-art models. Our second goal is to determine whether the common practice of defaulting to the cosine similarity is empirically supported. Our findings reveal that: (i) some feature transforms provide solid improvements, suggesting their default adoption; (ii) cosine similarity fares better than Euclidean similarity, thus backing up standard practices. Ultimately, our takeaways provide actionable advice for practitioners.

Keywords: Feature transform · Cosine similarity · Image retrieval · Text retrieval · Semantic similarity · Text embeddings · Image embeddings

1 Introduction

Extraction of image and text features with pre-trained off-the-shelf models enjoys widespread adoption among practitioners (e.g., BERT-as-a-service [58]). These features are often used in tasks such as multimedia retrieval [50,55], semantic similarity [12,13,26,40,52], word analogies [37,40] or zero-shot image recognition [46,61], to name a few. Not infrequently, features are used directly without further supervised training, typically via cosine-based semantic match. As noted [1,24], the cosine similarity is sensitive to centering, cross-dimension correlations and scale variations (Fig. 1). However, the extent to which this impacts task performance has not yet been systematically studied. Studies assessing the effect

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 98–114, 2021.
https://doi.org/10.1007/978-3-030-72113-8_7

of feature transforms (e.g., normalizing or PCA) typically restrict to a single domain and task (e.g., bilingual word embeddings [1,59]) and a single modality (text). This prompts our first research question (**RQ1**): *Can we improve the features with simple transforms in a variety of text and image tasks?* In particular, quantifying the (hypothesized) negative impact of vector *uncenteredness* on cosine-based performance (Fig. 1) is among our foremost hypothesis to test.

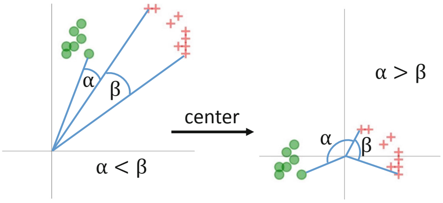


Fig. 1. Illustration of uncentered vectors hindering cosine similarity performance. Since cosine similarity computes the angle (α, β) from the origin $\vec{0}$, in this example where all vectors are dimension-wise positive, the *cosine* judges two points from different classes as more similar than two points of the same class. *Centering* helps obtaining more meaningful similarity estimates.

The cosine is generally chosen as default similarity measure in retrieval [15,50] and semantic similarity tasks [12,26,27,31,40,52,53]. This choice may eventually be informed in a (labelled) validation set or even the metric itself can be learned [14,56] if a labelled training set exists. However, because often none of these are available [12,26,31,40,46,52], our study assumes a scenario without either set. This motivates our second research question (**RQ2**): *Is the default choice of cosine similarity (versus Euclidean) empirically supported?*

To answer **RQ1** and **RQ2**, we perform an extensive empirical study in *real-world* tasks with both image and text data. We provide further insight and back up our claims in laboratory experiments. Our tests include 25 datasets with 6 different tasks covering text and image retrieval, word-, sentence- and visual-similarity, and paraphrase detection. We include 15 types of image (8) and text (7) embeddings, covering state-of-the-art models. Simple feature transforms are also compared with manifold learning methods.

Our findings reveal that: (i) *Centering* and *standardizing* are remarkably effective across *real-world* tasks (**RQ1**); (ii) the *cosine* significantly outperforms the *Euclidean* similarity across 74 conditions (embedding × task), hence supporting the default choice (**RQ2**). Ultimately, our findings provide actionable advice to practitioners and warning about the negative impact of using cosine similarity along with uncentered features.

This paper is organized as follows. In Sect. 2, we discuss related work. We present our methods in Sect. 3 and our tasks in Sect. 4. In Sect. 5, we describe our embeddings and setup. In Sect. 6, we discuss our empirical results. Section 7 concludes the paper.

2 Related Work

Feature transforms: [25] study the optimality of five different whitening transformations from the viewpoint of the properties of their covariance matrices.

In contrast to this study, [25] do not include empirical evaluations in text or image problems.

Additionally, [11] studied the effect of transforming features with an untrained neural network (i.e., random projections), finding that the performance of transformed vectors does not drop in word-similarity tasks. The impact on performance of different feature transforms on *classification* problems such as of biomedical data is also studied [4].

The closest works to ours are [32], [59] and [1], all of whom study the effect of feature transforms in the context of text problems. [32] study the effect of hyperparameters and normalization of word embeddings, revealing that the impact of design decisions and hyperparameters on performance is more important than the choice of the embedding algorithms themselves. [59] finds that constraining word embeddings to the unit hyper-sphere (i.e., normalizing them) improves performance in mono-lingual word similarity and bi-lingual word translation. [1] investigate several transformations including PCA, mean centering, normalization and whitening in the context of multi-lingual word embeddings. In contrast with ours, these studies restrict to a single domain and to text data (no images), and do not discuss *standardizing* – which we find to be a top performer.

Similarity measures: [24] analytically study the behavior and properties of similarity measures such as cosine similarity and the inner product from a geometric viewpoint, focusing on iso-similarity contours. Also analytically, [41] studies similarity measures in the retrieval context. In contrast to them, we carry out extensive empirical tests.

Metric learning: algorithms such as the ITML [14] or LMNN [56] learn a metric distance which can be seen as a form of learning a suitable transformation to the input vectors. However, this metric is learned in a supervised fashion, typically to be used in conjunction with a nearest-neighbor classifier, which falls out of the *unsupervised* scope of our study. It is worth mentioning that unsupervised metric learning algorithms also exist [9,23], yet they do not witness widespread adoption among data practitioners.

Manifold learning: methods, such as Isomap [49], Locally Linear Embedding (LLE) [42], diffusion maps [10], multi dimensional scaling (MDS) [29] or t-SNE [34], try to discover the underlying data manifold, which enables disentangling the vectors in a lower-dimensional space. Such methods are widely used for data visualization, yet they are not popular as feature transforms for predictive models – perhaps due to their limited success for such purpose. Although the inclusion of manifold learning methods in our study obeys mainly completeness reasons – given that our focus are *simple* feature transforms – an empirical comparison of *simple* transforms and manifold learning methods across multiple tasks has not been performed yet and we believe that is of practical interest.

3 Method

Let us first lay down our general **framework**. Let $S = \{s_i\}_{i=1}^{N}$ be a set of N data points (sentences, words or images). One extracts corresponding feature vectors

$V = \{v_i\}_{i=1}^N$ with a text or image encoder $E()$ (e.g., BERT or a CNN model), where $v_i = E(s_i)$ and $v_i \in \mathbb{R}^d$. The parameters θ of a feature transform T_θ are learned using the vectors V (e.g., in *centering*, θ are the dimension-wise means). A new vector v can then be transformed with $T_\theta(v)$ (Sect. 3.1), where v may belong or not to the set V used for learning T_θ.

3.1 Feature Transforms

In the following, we describe the feature transforms included in our experiments.

- **Original (orig):** denotes the original vectors $V = \{v_i\}_{i=1}^N$ without any transformation.
- **Centering (ctr):** $\text{ctr}(v) = v - \overline{V}$; subtracts the centroid vector $\overline{V} = \frac{1}{N}\sum_{i=1}^N v_i = \frac{1}{N}\sum_{i=1}^N (v_i^1, \cdots, v_i^d)$ to a vector v.
- **Standardizing (stz):** $\text{stz}(v) = (v - \overline{V})/\text{sd(V)}$; where sd(V) are the component-wise standard deviations $\text{sd(V)} = (\text{sd}(V^1), \cdots, \text{sd}(V^d))$ with $V^k = \{v_i^k\}_{i=1}^N$; and sd() is the standard deviation. *Stz* zero-means the data V and sets variances equal to 1.
- **Whitening (wht):** We use the *Zero Components Analysis* (ZCA) whitening as described in [28]. ZCA de-correlates the data dimensions and makes the variances equal to 1.
- **Normalizing (Nrm):** $\text{nrm}(v) = v/||v||$; moves any vector v to the unit hypersphere. Unlike the rest, this transform depends only on the same vector v, and not on the whole set $V = \{v_i\}_{i=1}^N$. Normalizing has no effect when the *cosine* similarity is used.
- **Isomap (Iso):** [49] and **Locally Linear Embedding (LLE)** [42] are used analogously by first learning the parameters θ in the training set of vectors V, and applying the learned transformation T_θ to a new vector $v \in \mathbb{R}^d$, with $T_\theta(v) \in \mathbb{R}^m$ with $m \leq d$[1].
- **Principal Component Analysis (PCA):** is a classical dimensionality reduction method that finds orthogonal directions that best fit the data in the least-squares sense. We keep a number of components (dimensions) such that 80% of the variance is explained.[2] Our implementation of PCA [39] centers but does not scale the data (for each feature) before applying the SVD decomposition.

Unlike simple transforms (e.g., *center*) the more complex PCA, Isomap and LLE have hyperparameters (e.g., output dimensionality) that impact their performance. Thus a validation set is often necessary, which is a shortcoming in our *unsupervised* setting.

[1] For both *Isomap* and *LLE* we set $m = 100$ in the *real-world*, and $m = 2$ for *synthetic* tasks. The number of nearest neighbors is set to 10 in all tasks (as default in sklearn [39]).

[2] The choice of 80% of the variance is discussed and compared to other values in the Supplement.

3.2 Complementary Experiments

- **Additive bias:** As a complement to *centering*, we study the effect of "uncenteredness" on the *cosine* similarity (as the Euclidean is shift invariant) by uncentering $V = \{v_i\}_{i=1}^N$ with a dimension-wise bias $b > 0$, namely $(v_i^1 + b, \cdots, v_i^d + b)\ \forall\ i = 1, \cdots, N$. This equates shifting all vectors to the positive quadrant (i.e., $v_i^k > 0\ \forall\ k = 1, \cdots, d$), if b large enough, or moving them further up in case they already are (see Sect. 6 for a discussion).
- **Multiplicative bias:** to study the effect of *non-homogeneity* of *scale* and *variances* across dimensions, we multiply each dimension with a bias $b>0$ randomly drawn from a uniform $b \sim \mathcal{U}[0.001, 10]^d$, i.e., $v_i = (b_1 v_i^1, \cdots, b_d v_i^d)\ \forall\ i = 1, \cdots, N$. This study complements the *standardizing* method.

4 Tasks and Data

In this section, we first describe the procedure of two grand groups of tasks (Sect. 4.1), and then we introduce the datasets used in each individual task (Sect. 4.2). Our dataset selection criteria included: (i) Feasibility of implementing an *unsupervised* prediction approach (i.e., simply cosine-based); (ii) medium-sized datasets; (iii) rather popular and already clean data (thus little preprocessing required); (iv) diversity.

4.1 Task Descriptions

Grouping tasks: It is convenient to group our tasks in two functionally different categories, as they exhibit identical prediction-evaluation pipelines: (1) **Retrieval tasks:** (i) *text retrieval* and (ii) *image retrieval*; (2) **Similarity tasks:** (iii) *word similarity*, (iv) *sentence similarity*, (v) *visual similarity* and (vi) *synthetic data*. Furthermore, we refer throughout to **real-world** tasks being all tasks except the **synthetic** ones.

In *all* tasks, we consider two **similarity measures** to compute the *predicted similarity* $\text{sim}(s_1, s_2)$ between any two inputs s_1, s_2 (words, sentences or images) encoded with their respective features $v_i, v_j \in \mathbb{R}^d$:

- **Cosine** similarity: $\cos(v_i, v_j) = \frac{v_i v_j}{\|v_i\| \|v_j\|}$.
- **Euclidean** similarity $\text{Eucl}(v_i, v_j) = \frac{1}{1+\|v_i - v_j\|}$.

In the interest of the practitioner, we focus on *simple* and widely adopted transforms, *cosine* and Euclidean similarity, rather than aiming for an exhaustive comparison of all existing similarity measures and feature transforms. After having obtained the vectors $V = \{v_i\}_{i=1}^N$ and learned $T_\theta(v)$ as described in Sect. 3, we consider the task-specific procedures below.

■ <u>Similarity tasks:</u> All word-, sentence- and image-similarity datasets consist of a list of word, sentence or image pairs (s_i, s_j), e.g., ('car', 'truck') along with a human (ground-truth) rating of their similarity or relatedness $y_{i,j} \in [1, 10]$. The system needs to predict a similarity score $\hat{y}_{i,j} \in [1, 10]$ for each pair (s_i, s_j). Model predictions are computed via $\cos(v_i, v_j)$ or $\mathrm{Eucl}(v_i, v_j)$, where $(v_i, v_j) = E(s_i, s_j)$.

- **Evaluation:** Following [12,26,40], we use the **Spearman correlation** $\rho(\hat{y}, y)$ between the predicted $\hat{y} \in \mathbb{R}^N_+$ and the ground-truth similarity scores $y \in \mathbb{R}^N_+$ as the standard measure to evaluate the quality of semantic similarity predictions.

■ Retrieval tasks: We split the given test set V^{ts} into two disjoint sets: a *query set* $\mathcal{Q}$ and a *test collection* $\mathcal{T}$. Given a query $s_i \in \mathcal{Q}$, the goal of the task is to rank the relevant items from $\mathcal{T}$ higher than the non-relevant ones. The similarity between each item $s_i \in \mathcal{Q}$ in the query set $\mathcal{Q}$ is computed against *every* item $s_j \in \mathcal{T}$ in the test collection $\mathcal{T}$ via $\cos(v_i, v_j)$ or $\mathrm{Eucl}(v_i, v_j)$ similarity, where $(v_i, v_j) = E(s_i, s_j)$.

- **Evaluation:** Performance is evaluated with the TREC standard **mean average precision (mAP)**, as described in [35]. Following [50,54,55], a test-collection item $s_i \in \mathcal{T}$ is considered relevant to a query $s_j \in \mathcal{Q}$ if they both belong to the same class.

4.2 Datasets

■ Text retrieval: **AG-news**[3] is text classification and retrieval benchmark [60] consisting in (120,000 train; 7,600 test) sentences, each belonging to exactly one of the 4 classes (sports, world, business, sci/tech). E.g.,"Economic growth in Japan slows down as the country experiences a drop in domestic and corporate spending" (class = business).

■ Image retrieval:

- **Caltech-256** [20] is a benchmark widely used in image retrieval [15] and classification. The data consists of 30,607 images, each of which belongs to exactly one of the 256 categories (e.g., sushi, swan, tripod, etc.).
- **CorelDB** database [51]: consists of 10,800 images, each of which belongs to exactly one of the 80 classes (ship, waterfall, lion, etc.).

■ Word similarity tasks are typically used to evaluate the quality of word embedding models [2,26,31,40,52]. Following [12,52,53], we use five *word similarity* benchmarks, which include three types of similarity ratings: *(i) Semantic similarity*: **SemSim** [44], **Simlex999** [22] and **SimVerb-3500** [19]; *(ii) Relatedness*: **MEN** [3] and **WordSim-353** [18]; *(iii) Visual similarity*: **VisSim** [44] which contains the same data as SemSim, yet word pairs are rated for visual similarity instead of semantic similarity.

[3] https://www.kaggle.com/amananandrai/ag-news-classification-dataset.

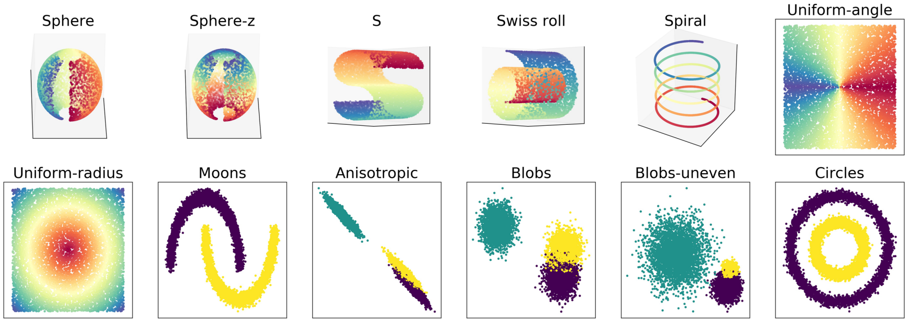


Fig. 2. Synthetic datasets of our laboratory experiment. Color indicates the semantic value of each data point (either a class label or a continues value) – best seen in color. The first five datasets are 3D while the rest are in 2D. In the first seven datasets, the semantic value assigned to data points is continuous, while for the last five datasets the class labels are discrete. (Color figure online)

■ Sentence similarity: Our datasets are from the GLUE[4] and SentEval[5] collections.

- **STS** (Semantic Textual Similarity) [5] is a semantic relatedness benchmark consisting of sentence pairs with a crowd-annotated similarity score. E.g., ("A woman is eating something", "A woman is eating meat") has a score of 3 (out of 5). There are 5,749 train, 1,500 val and 1,379 test pairs.
- **SICK** (Sentences Involving Compositional Knowledge) [36] evaluates compositional distributional semantics. SICK contains sentence pairs along with their semantic relatedness score. E.g., ("Two men are boxing", "Two men are fighting") have a score of 4 (out of 5). SICK has 4,501 train, 501 val and 4,928 test sentence pairs.
- **MSRP** (Microsoft Research Paraphrase Corpus) [17] does not strictly evaluate sentence similarity but paraphrase detection, yet due to functional parallels with the former, we include MSRP in this group. It contains (4,077 train; 1,726 test) sentence pairs along with a label $\{1 = \textit{paraphrase}$ or $0 = \textit{not paraphrase}\}$. MSRP is always used with supervision, thus it may not be the most adequate test-bed for our setting.

■ Visual similarity: **Visual-STS (vis-STS)** [30] is a subset of STS where each textual caption is associated to an image. Here, we only use the images since (a larger super-set of) the sentences are already evaluated in STS. Vis-STS consists of 1,089 images and a single set of 829 image-image pairs along with their ground-truth similarity rating.

[4] https://gluebenchmark.com/tasks.

[5] In contrast to most papers using SICK, MSRP and STS [13,27] we do not use labels. E.g., while [27] learn a logistic regression model to predict the similarity between embedding pairs v_i, v_j, we output the similarity directly (Sect. 4.1).

■ Synthetic data: In contrast to *real-world* tasks, laboratory tasks offer a unique window to study the behavior of feature transforms by having full control of: (i) the (distribution of) feature vectors, (ii) the task itself, i.e., the *assignation of semantic value* to each data point. The majority of our **synthetic (laboratory) datasets** are from sklearn [39], except *sphere-z*, *unif-rad*, *unif-angle* and *spiral* (Fig. 2), which are built by ourselves.

We randomly generate 2,000 train and 200 test data points. Then, we build our *similarity* task by presenting all pairwise combinations of test points to the system, i.e., 40,0000 pairs (= 200 × 200). In the *discrete*-labelled datasets (e.g., *circles*, Fig. 2) where each data point s_i has a class label $l_i \in \{t_1, \cdots, t_C\}$ (where $C = \#$ classes) the ground-truth similarity $y_{i,j} \in \{0, 1\}$ between two points s_i, s_j is 1 if they belong to the same class, or 0 otherwise. In the *continuous*-labelled datasets (e.g., *sphere*), where the assignation of semantic value to each data point is a continuous value $l_i \in \mathbb{R}_+$, the ground truth similarity $y_{i,j}$ between s_i, s_j is the absolute difference: $y_{i,j} = |l_i - l_j| \in \mathbb{R}_+$.

5 Experimental Setup

5.1 Feature Vectors (Embeddings)

We group below our embeddings by the unit that they represent (a word, a sentence or an image). An overview of which embeddings apply to what task can be seen in Table 1.

■ Word-level features:

- **GloVe**[6] [40]: We use 300-*d* vectors pre-trained on the Common Crawl corpus with 840B tokens and a 2.2M-word vocabulary.
- **word2vec (w2v)** [37]: We use the skip-gram 300-*d* embeddings trained on Wikipedia.
- In *word-similarity*, we adopt the publicly available[7] **VGG-128** [6] and **ResNet** [21] visual features from [12]. Notice that unlike the image retrieval and visual-STS tasks, word-similarity datasets do not have any images and hence one needs to find a way to visually represent each word (e.g., 'cat' or 'table') by using external visual data. To this end, [12] used ImageNet [43], and for each image they extracted 128-*d* VGG-128 and 2,048-*d* ResNet features from the last layer (before the softmax) by using the forward pass of the CNN. The final representation for any given word is the average feature vector (centroid) of all available images for this word in ImageNet.

■ Sentence-level features:

- **BERT** [16]: The large uncased version of BERT[8] (24 layers, 1,024 units) is used as a *sentence* feature extractor. We obtain a 1,024-*d* vector from the last

[6] http://nlp.stanford.edu/projects/glove.

[7] http://liir.cs.kuleuven.be/software.html.

[8] Although we are aware that BERT is not meant to represent a single word as it is designed to account for context words, we include BERT in the *word-similarity* tasks for completeness.

layer (24th), before the model top, by average-pooling the output sequence of hidden state vectors, similar to BERT-as-a-service [58]. The model is pre-trained on masked language modeling and next sentence prediction in the Toronto Book Corpus and Wiki.

- **RoBERTa** [33]: We obtain 1,024-*d* features in an identical manner as in BERT above with the large-version of a case-sensitive RoBERTa model.
- **Skipthoughts** vectors [27] is a popular neural-based universal sentence encoder that learns sentence representations by predicting the surrounding sentences. We use the best-performing 4,800-*d* vectors (combine-skip) as recommended by the authors.
- **Vector averaging (bag of words):** In the sentence-level tasks (SICK, MSRP, STS and AG-news), we include the baseline sentence representation $v = \frac{1}{m}\sum_{i=1}^{m} v_i$ of averaging word vectors in a sentence $s = (s_1, \cdots, s_m)$, where $v_i = E(s_i)$ and m is the number of words. We add a subscript *avg* to the averaged vectors (e.g., GloVe$_{\text{avg}}$).

■ Image-level features. Vector dimensionality is in parenthesis: **NASNet** [62] ($d = 4{,}032$), **ResNet-50** [21] ($d = 2{,}048$), **ResNet-inception-v2** [47] ($d = 1{,}536$), **Inception-v3** [48] ($d = 2{,}048$), **VGG19** [45] ($d = 512$), **Xception** [8] ($d = 2{,}048$). In all these CNN networks, the feature vector $v_i = E(s_i)$ for a given image s_i is obtained as the forward pass average-pooled activations from the last layer before the output layer.

5.2 Training Setup and Implementation

- **Given training data:** In **all datasets except word-similarity** (Sect. 4.2), we obtain the training data $V^{tr} = E(S^{tr})$ given in the dataset (yet *without* using class labels). In the case of AG-news, STS, SICK and MSRP we use the provided train-test split (Sect. 4.2). Although CorelDB, Visual-STS and Caltech-256 do not have publicly available train-test set splits, we create the train-test splits ourselves via 3-fold cross-validation. I.e., we split the full data $S = \{v_i\}_{i=1}^{N}$ into 3 disjoint parts and we employ 2 parts for training (S^{tr}) and 1 part for testing (S^{ts}), repeating this 3 times and reporting the average. However, our setting does not require having an available training set. There are two main alternatives to using the given train split: (1) learning $T_\theta()$ in the *test* set; (2) generating S^{tr} ourselves. Although (1) is a legit option (as one does not use labels), it falls within a *transductive learning* setup and assumes a test set of a certain size to enable learning $T_\theta()$. Hence, this is not an option in the case of a singe-instance test set. We also evaluated learning $T_\theta()$ in the test set, and results are discussed in Sect. 6.1.
- **Built training data:** For **word-similarity**, where no training data are available, we use external data to generate V^{tr} (option (2) above). Following [12], we build V^{tr} in *word-similarity* by using features obtained from all words in ImageNet, i.e., visual features for CNNs (ResNet & VGG128) and word embeddings for text (GloVe & word2vec).

Implementation: We use diverse Python libraries, including: Keras [7] for the CNNs, Theano for skipthoughts, sklearn [39], Pytorch and Huggingface [57] for BERT & RoBERTa. We make our code publicly available[9] as well as a Supplement with further specific implementation and hyperparameter details and additional results.

6 Results

Unless otherwise specified, results below are discussed for the *cosine* similarity (Table 1). Performance measures in the tables are according to Sect. 4.1, and scaled × 100, for readability. Table 2 reports **statistical significance** of comparing a given method with the *original* vectors under *cosine* (i.e., the top left corner entry). Each comparison is a two-sided Wilcoxon signed-rank test across the 74 combinations of a *real-world* dataset with an embedding type (i.e., rows in Table 1). We report significance at $p < 0.01$ after a Bonferroni correction for 10 comparisons (7 methods in the first row + Eucl + add. bias + mult. bias)[10]. **Win-tie-loss** results (W, T, L) indicate the number of wins (W), ties (T) and losses (L) of the first method against the second one, across the 74 combinations.

6.1 Real-World Tasks

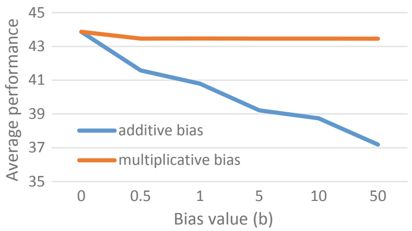


Fig. 3. Averaged results across datasets and features for different values of biases (Sect. 3.2) on *original* vectors. The b=0 point means no bias.

Centeredness: Performance of *original* with an **additive bias** (Sect. 3.2) drastically drops (Fig. 3 and Table 2). This confirms the inadequacy of using uncentered vectors along with the *cosine* similarity. Results of *PCA*, *ctr* and *stz* are unaffected.

- **Centering:** Consistently with the results above, *centering* significantly improved ($p < 10^{-4}$) the *original* features by an absolute 2.5% on average (Table 2), with a win-tie-loss of (W = 52, T = 1, L = 21) (Table 1), hence proving the effectiveness of this method (**RQ 1**).
- **Centeredness of original vectors:** All our CNN vectors (ResNet, etc.) are positive (thus uncentered), and simple statistical inspection reveals that our text vectors are also uncentered. This implies that *centering* has an effect on all our features.

(Non-)homogeneity of variances and scale: In contrast with the large hindering effect of the *additive bias*, performance with the **multiplicative bias**

[9] https://github.com/gcollell/transforms-cosine.

[10] We did not test all pair-wise conditions as our interest is on a specific set of hypotheses.

(Sect. 3.2) barely drops (Fig. 3 and Table 2). This suggests that *centeredness* may have a larger impact on the *cosine* similarity than scale and variance differences across dimensions.

• **Standardizing** is the overall winner in *real-world* tasks (**RQ 1**). It improved significantly ($p < 10^{-6}$) the *orig* features by an absolute 3.3% on average (Table 2) and their win-tie-loss is (W = 60, T = 0, L = 14) (Table 1). Notice that *stz* also centers the vectors.

Cosine versus Euclidean: *Cosine* similarity significantly outperformed ($p < 10^{-6}$) the Euclidean similarity (**RQ 2**) by an average absolute 5.1% (Table 2) and (W = 54, T = 7, L = 13), for the *original* vectors – yet the trend is similar for all transforms. This supports the common practice of defaulting to *cosine* similarity, yet we strongly recommend considering the remarks about *centering* above, to avoid sub-optimal performance. Further, if a labeled validation set is available (e.g., in SICK, STS, or AG-news), one may use it in order to make a more educated choice between *cosine* and *Euclidean* similarity.

Learning times: Remarkably, manifold learning methods are over ×1,000 times slower than *standardizing* (Table 2), and perform markedly worse.

Learning in test set: Notably, *center* and *standardize* can be further improved by learning them in test data (Table 2) – provided the test set is large enough.

Manifold learning methods generally underperform the simple transforms in *real-world* tasks. We emphasize that we do not claim that we fairly portray the full potential of manifold learning methods (and PCA), as we did not tune their hyperparameters (e.g., dimensionality) with a validation set for the sake of comparability with the simple transforms – as our setting does not assume a validation set.

PCA improved *orig* features by 1.8% on average (Table 2) and (W = 53, T = 0, L = 21).

Failure cases: Notably, VGG19 was not improved by any method in any dataset (Table 1), and all methods fared poorly in MSRP. However, the performance loss by *standardizing* or *centering* is small in MSRP, which suggests that, in the absence of a validation set for making more informed decisions, the large upside of defaulting to standardizing may offset its eventual and rather small potential performance downside.

Consistency: Some methods that perform poorly on average such as *Iso* or *wht* (Table 2) eventually hit the most spectacular gains (and losses) (Table 2). This contrasts with *stz* and *ctr* which tend to have less “volatility” and exhibit more consistent gains.

Table 1. Results with *cosine* similarity on **real-world** tasks. Since performance trends are similar, the *word-similarity* table (left) includes only the visual subsets, i.e., word-pairs for which images are available for both words – number of instances is in parenthesis. Results in all sets are in the Supplement. Best-performing method per row is boldfaced.

		orig	ctr	stz	wth	iso	LLE	PCA
wordsim (63)	GloVe	63.2	61.3	67.6	**69.6**	43.9	50.1	59.4
	w2v	**66.9**	64.9	65.5	62	58.8	25.3	63
	BERT	20.8	30.1	28	**47.4**	16.7	26.3	29.9
	RoBERTa	23.9	23.5	26.7	**44.9**	23.1	11.2	22.4
	ResNet	42.3	48.9	48.1	28.7	**51.7**	43.7	48.6
	VGG128	44.8	49.2	49.1	50	**55.4**	54	46.7
men (795)	GloVe	80.1	80	**82.7**	79.9	76.6	67	80.2
	w2v	78.7	81	**81.1**	74.3	70.8	53.5	80.9
	BERT	32.7	31.9	35.9	**49.5**	25.3	24.1	31.3
	RoBERTa	20.9	29.5	32.2	**48**	25.1	20.7	27.6
	ResNet	56.7	59	60.7	36.8	**60.9**	42.3	59.2
	VGG128	59.3	58.9	59.8	54	**60**	42.9	58.8
semSim (5,238)	GloVe	76.8	74.6	**78**	62	77.3	61.2	75.1
	w2v	74.2	77.3	77.3	54.1	71.2	50	**77.6**
	BERT	23.4	22.6	25.2	**28.3**	17.3	22.7	22.8
	RoBERTa	20	28.5	**30.3**	26.1	26.2	22.8	28.2
	ResNet	53.4	67.6	67.6	11.7	**70.3**	39.1	67.8
	VGG128	53.4	65.8	65.1	37.9	**69**	36	66.1
visSim (5,238)	GloVe	60.6	60.6	**62.9**	53.7	61.5	47.7	61
	w2v	57.6	60.8	60.8	47.7	54.8	37.9	**61**
	BERT	16.2	16.7	18.4	**23.7**	12	14.8	16.7
	RoBERTa	15.7	21.1	**22.4**	22.2	18.6	15.4	20.7
	ResNet	54.3	60.6	**61.7**	14.5	57.9	37	60.8
	VGG128	56	60.7	**61.2**	42.9	60	35.2	60.7
simlex (261)	GloVe	37.1	36.1	42	45.1	35.6	**45.3**	35.7
	w2v	43.5	44.3	**44.9**	41.7	41.9	35.2	43.8
	BERT	24.3	21.6	23.7	**36.6**	18.5	16	20.9
	RoBERTa	−7.6	−6.6	−4.5	**19.3**	−11	−3.8	−7.7
	ResNet	40.9	45	45.6	36	**47.3**	38.8	45.5
	VGG128	40.6	42.6	42.2	40.3	43	34	**43.3**
SimVerb (41)	GloVe	32	29.8	**34.3**	22.8	10.5	−6.6	34.1
	w2v	30.8	19.7	21.3	**31.3**	−4.9	−3.1	12.7
	BERT	−7.2	−8.2	−6.6	13.6	−11.6	**21.5**	−7.1
	RoBERTa	4.7	1.5	2	−6.1	−9.8	1.5	**5.7**
	ResNet	21.1	21.2	**27.7**	22	23.7	4.5	19.1
	VGG128	23.5	23.4	20.6	**52**	20.8	14.6	22

		orig	ctr	stz	wht	iso	LLE	PCA
STS	GloVe$_{avg}$	50	57.6	58.9	**68.1**	37.7	42.5	51.2
	w2v$_{avg}$	55.1	59.6	60.1	**66**	38.2	42.8	56.6
	skipthoug	34.5	38.8	44.9	**59.3**	17.2	30.9	35.3
	BERT	47.7	53.5	55.1	**64.5**	34.4	42.3	51.1
	RoBERTa	44.9	58.6	62.7	**69.1**	38.6	42.8	55.1
SICK	GloVe$_{avg}$	56	58.5	**59.3**	58	51.9	45.6	55.2
	w2v$_{avg}$	58.5	60	**60.3**	56.3	51.6	46.1	57.9
	skipthoug	56.5	**58.5**	56.1	43.3	48.4	44	57.1
	BERT	53.8	57.8	58.2	**58.4**	50.7	44.5	55.8
	RoBERTa	58.1	61.9	**63.8**	61.5	54.4	46.4	59.9
MSRP	GloVe$_{avg}$	**39.9**	37.2	38	39.5	16.3	16.1	34.4
	w2v$_{avg}$	**38.8**	35.6	36.2	37.8	14	21.6	33.7
	skipthoug	15.3	19.6	21.1	**32.9**	7.6	8.8	16.4
	BERT	31.5	30	30.8	**38.2**	11.6	15.3	26.3
	RoBERTa	**43.7**	39.5	39.5	42.3	10.9	15.8	36.6
AG-news	GloVe$_{avg}$	48.5	55.6	52.6	30.6	**62.9**	59.4	55.7
	w2v$_{avg}$	54.9	60.9	60	31.3	**66.5**	56.7	61.2
	skipthoug	36.6	**39.3**	38.5	27.1	**39.3**	37.8	**39.3**
	BERT	49.9	57.6	56.9	29.6	**66.2**	50.5	57.6
	RoBERTa	47.3	55	54.8	29.6	**67.8**	54.1	55.1
vis-STS	Incptn-v3	44.9	**55.3**	54.6	10.3	53.4	37.8	54.2
	ResNet-in	45.2	56.1	55.5	9.5	**57.9**	37.2	54.7
	ResNet	61.7	**64.6**	60.9	10.8	59.2	43.2	63.5
	xception	56.2	**59.2**	59.1	14.3	47.9	38	58.2
	VGG19	**63.4**	56.5	53.2	5.8	53.2	32.4	54.2
	NASNet	49.9	**58.7**	57.4	15.5	53.7	34.7	57.2
Caltech	Incptn-v3	45.8	47.9	**48.5**	27.4	42	45.5	48.3
	ResNet-in	54.9	55	55	38.7	55.2	52.8	**55.4**
	ResNet	40.6	41.1	**42.7**	22.5	36.2	38.9	41.3
	xception	46.8	50	49.9	27.8	41.7	46.5	**51**
	VGG19	**35.1**	34.3	35.2	25.8	26.6	26.7	33.8
	NASNet	60.4	60.6	60.2	29.9	58.3	57.2	**61.6**
CorelDB	Incptn-v3	43.8	46.9	47.3	11.6	42.6	**50.2**	47.9
	ResNet-in	52.3	53.5	53.2	18.6	**55.4**	54.5	55.1
	ResNet	45.7	47.2	46.1	12	**48.5**	52.4	48.1
	xception	47.4	50.4	49.7	13.6	47	**52.8**	51.9
	VGG19	36.9	38.2	37.7	18	34.8	**38.8**	38.5
	NASNet	57.9	59.2	57.4	2.2	56.6	57.7	**60.9**

6.2 Synthetic Data

Unlike *real-world* data (Sect. 6.1) where vectors and semantic value assignment (i.e., the task) cannot be visualized, synthetic data enable intuitively grasping and visualizing the effect that transforming vectors (**RQ 1**) have on the similarity measures (**RQ 2**).

Table 2. Averaged results across *real-world* datasets and features. Rows include (in order) results of: (i) *cosine* similarity (i.e., averaged results of Tab. 1); (ii) *Euclidean* similarity; (iii) *additive* and (iv) *multiplicative* bias (Sect. 3.2) ($b = 10$); (v) learning T_θ in the test set, and (vi) *training times* (in seconds). Despite omitting datasets, this table portrays a representative summary of the performance landscape. SDs are omitted for being uninformative, as they reflect inter-dataset variance. For individual results, see Table 1, the Supplement and win-tie-loss mentions in the text. Asterisks (*) indicate statistically different performance ($p < 0.01$) from *orig* × cos (two-sided).

	orig	ctr	stz	wht	nrm	iso	LLE	PCA
cos	43.9	46.4*	47.2*	36.2*	43.9	40.9	35.7*	45.7*
Eucl	38.8*	38.8	38.1	21.6	43.9	35.7	23.1	39.8
add. bias	38.7*	46.4	47.2	35.1	38.7	40.9	35.5	45.7
mult. bias	43.5	45.8	47.2	36.1	43.5	40.6	35.8	44.9
learn in test	43.9	47.3	47.9	35.8	43.9	44.5	38.1	46.8
train time	0	0.01	0.3	7.9	2.1e-05	1174.5	965.7	2.4

Centeredness: Crucially, *original* vectors in synthetic tasks are generally centered while in real-world tasks features are uncentered (Sect. 6.1). It is reasonable to not expect that features will be natively centered at $\vec{0}$, unless explicitly imposed. Thus, using uncentered vectors *orig (add)* as a reference point in Table 3 may be more "realistic" than *orig*.

- Applying an **additive bias** (*orig (add)*) generally hinders the *original* vectors (with *cosine*) (Table 3), yet one can find a pathological case in *circles*, where having centered vectors (e.g., *orig* or *ctr*) is detrimental. The reason being that, with centered vectors, the $\vec{0}$ point falls inside the circles (Fig. 2), hence the angle (or *cosine* similarity) which stems from $\vec{0}$, is utterly unhelpful to tell apart the inner from the outer circle. Although it is important to gain insight on these cases with synthetic data, real-world feature vectors (and tasks) are unlikely to exhibit this onion-like structure unless explicitly imposed [38, 59]. Thus, there is no substitute for a systematic study in *real world* tasks (Table 1).

Task versus vectors: A key question that this paper answers is whether it suffices to look at (the statistics of) the vectors alone in order to tell when a transform will perform well. *Unif-radius* and *unif-angle* illustrate a negative answer. *All* methods fail at *unif-radius* (radius matters) while they all do reasonably well in *unif-angle* (angle matters). The only difference is the assignment of a semantic value to data points, i.e., the *task* itself. Thus, vectors alone do not suffice to determine effectiveness of a transform but they must be considered along with the task. Many *real-world* instances support this conclusion, e.g., *stz* improving NASNet in vis-STS, yet not in Caltech nor in CorelDB.

Failure cases: *Circles* illustrates a task where *cosine* similarity is entirely unhelpful to tell both classes apart (and the Euclidean only barely useful) for the

Table 3. Results on **synthetic** datasets. The (add) and (mult) indicate that an additive or multiplicative bias, respectively, is added to the method (Sect. 3.2). SDs are left to the Supplement.

	orig		orig (add)		orig (mult)		ctr		stz		wth		nrm		iso		LLE		PCA	
	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl	Cos	Eucl
sphere	65.1	65.1	44	65.1	58.5	54.9	62.1	65.1	62.5	65.8	65.4	65.3	65.1	65.1	59.7	76.7	58.2	71.3	50.7	50.2
sphere-z	53.1	53.1	40.6	53.1	46.5	45.4	53.5	53.1	51.6	49.7	51.4	50.1	53.1	53.1	52.7	49.7	51.1	44	59.6	64.2
s	82.7	88.3	74.2	88.3	70.3	72.8	76.3	88.3	53.3	58.9	66.2	74.3	82.7	82.7	76.3	98.7	70.5	77.5	75.3	91.3
roll	20.2	23.9	19.6	23.9	17.9	21.7	22.1	23.9	21.2	22.1	24.5	28.9	20.2	20.2	75.3	96	60.8	71.1	23.8	31.8
spiral	40.3	100	89.4	100	36.9	94.3	83.2	100	41.5	55.5	13.7	21.4	40.3	40.3	82.8	97.2	71.5	76.1	79.1	100
unif-ang	62.1	50.8	28.2	50.8	55.7	44.2	62	50.8	62	50.8	62.1	50.8	62.1	62.1	62	50.9	61.8	49.8	62	50.8
unif-rad	0	5.2	3.5	5.2	−0.1	7.7	0	5.2	0	5.2	0	5.2	0	0	0	5.1	0	6	0	5.2
moons	43	41.2	49.7	41.2	43.3	42.6	38.4	41.2	47.6	52.4	47	51.6	43	43	0	16.9	32.4	53.7	38.4	41.2
Aniso	54.3	62.1	54.3	62.1	52.8	61.1	62.1	62.1	61.5	61.1	58.2	62	54.3	54.3	39.9	40.3	53.7	63.1	62.1	62.1
blobs	57.9	66.3	53.4	66.3	56	64.3	64.2	66.3	64.8	66.4	58.2	62	57.9	57.9	39	42.6	55.8	66.1	64.2	66.3
blobs-un	54.8	61.2	48.1	61.2	52.4	59.1	63.8	61.2	64	60.4	55	55.1	54.8	54.8	58.9	42.3	55.9	60.2	63.8	61.2
circles	−0.1	13.4	11.8	13.4	0	13.1	−0.1	13.4	−0.1	13.4	−0.1	13.4	−0.1	−0.1	40.4	28.4	35.2	53.6	−0.1	13.4

regular methods, yet manifold learning methods fare better (Table 3). Further notice the detrimental effect of *normalizing* with the Euclidean similarity in the same dataset, as normalizing collapses both circles into one. We also highlight the general failure of *all* methods in our own "stress test" task, *unif-rad*. Likely, polar coordinates would have done a better job.

7 Conclusions and Future Work

Limitations. The answer to whether any of our top-performing transforms is a universal recipe to improve (text or image) features, is a negative one. As usual, there is *no free lunch*. However, this study strives to include a representative and reasonable number of datasets and varied tasks to gain insight on the success rate and effect size of each transform. **Performance trends** showcase promising potential on defaulting to *center*, *PCA* or *standardize* the features in applications, as well as using *cosine*-based (instead of Euclidean) semantic match. That said, our task selection is not exhaustive and hence we encourage researchers to report results on new tasks and datasets.

A word of caution. In line with [33] and [32], an important contribution of this work is rising awareness about the potential source of improvements in some word and sentence embedding models, which are often tested in semantic-similarity tasks and default to *cosine* similarity. As shown, feature re-scaling can have a much greater impact on the overall performance than the embedding model itself. Hence, it is crucial to control for any possible feature re-scalings occurring in any step of the pipeline.

Acknowledgment. This research was supported by the ERC Advanced Grant CALCULUS (H2020-ERC-2017-ADG 788506).

References

1. Artetxe, M., Labaka, G., Agirre, E.: Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In: AAAI, pp. 5012–5019 (2018)
2. Baroni, M., Dinu, G., Kruszewski, G.: Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In: ACL, pp. 238–247 (2014)
3. Bruni, E., Tran, N.K., Baroni, M.: Multimodal distributional semantics. J. Artif. Intell. Res. **49**, 1–47 (2014)
4. Cao, X.H., Stojkovic, I., Obradovic, Z.: A robust data scaling algorithm to improve classification accuracies in biomedical data. BMC Bioinformatics **17**(1), 359 (2016)
5. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. arXiv preprint arXiv:1708.00055 (2017)
6. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: delving deep into convolutional nets. In: BMVC (2014)
7. Chollet, F., et al.: Keras (2015). https://github.com/keras-team/keras
8. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: CVPR, pp. 1251–1258 (2017)
9. Cinbis, R.G., Verbeek, J., Schmid, C.: Unsupervised metric learning for face identification in TV video. In: ICCV, pp. 1559–1566. IEEE (2011)
10. Coifman, R.R., Lafon, S.: Diffusion maps. Appl. Comput. Harmonic Anal. **21**(1), 5–30 (2006)
11. Collell, G., Moens, M.F.: Do neural network cross-modal mappings really bridge modalities? In: ACL, pp. 462–468 (2018)
12. Collell, G., Zhang, T., Moens, M.F.: Imagined visual representations as multimodal embeddings. In: AAAI, pp. 4378–4384. AAAI (2017)
13. Conneau, A., Kiela, D.: Senteval: An evaluation toolkit for universal sentence representations. arXiv preprint arXiv:1803.05449 (2018)
14. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: ICML, pp. 209–216. Corvalis, Oregon, USA (June 2007)
15. Deng, J., Berg, A.C., Fei-Fei, L.: Hierarchical semantic indexing for large scale image retrieval. In: CVPR, pp. 785–792. IEEE (2011)
16. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL, pp. 4171–4186 (2019)
17. Dolan, B., Quirk, C., Brockett, C.: Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In: COLING, pp. 350–356 (2004)
18. Finkelstein, L., et al.: Placing search in context: the concept revisited. In: WWW, pp. 406–414. ACM (2001)
19. Gerz, D., Vulić, I., Hill, F., Reichart, R., Korhonen, A.: Simverb-3500: a large-scale evaluation set of verb similarity. In: EMNLP, pp. 2173–2182 (2016)
20. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset (2007)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385 (2015)
22. Hill, F., Reichart, R., Korhonen, A.: Simlex-999: evaluating semantic models with (genuine) similarity estimation. Comput. Linguist. **41**(4), 665–695 (2015)

23. Jiang, J., Wang, B., Tu, Z.: Unsupervised metric learning by self-smoothing operator. In: ICCV, pp. 794–801. IEEE (2011)
24. Jones, W.P., Furnas, G.W.: Pictures of relevance: a geometric analysis of similarity measures. J. Am. Soc. Inform. Sci. **38**(6), 420–442 (1987)
25. Kessy, A., Lewin, A., Strimmer, K.: Optimal whitening and decorrelation. Am. Stat. **72**(4), 309–314 (2018)
26. Kiela, D., Bottou, L.: Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In: EMNLP, pp. 36–45 (2014)
27. Kiros, R., et al.: Skip-thought vectors. In: NIPS, pp. 3294–3302 (2015)
28. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
29. Kruskal, J.B.: Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika **29**(1), 1–27 (1964)
30. de Lacalle, O.L., Soroa, A., Agirre, E.: Evaluating multimodal representations on sentence similarity: vSTS, visual semantic textual similarity dataset. arXiv preprint arXiv:1809.03695 (2018)
31. Lazaridou, A., Baroni, M., et al.: Combining language and vision with a multimodal skip-gram model. In: NAACL, pp. 153–163 (2015)
32. Levy, O., Goldberg, Y., Dagan, I.: Improving distributional similarity with lessons learned from word embeddings. Trans. Assoc. Comput. Linguist. **3**, 211–225 (2015)
33. Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
34. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(Nov), 2579–2605 (2008)
35. Manning, C.D., Schütze, H., Raghavan, P.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
36. Marelli, M., Menini, S., Baroni, M., Bentivogli, L., Bernardi, R., Zamparelli, R., et al.: A sick cure for the evaluation of compositional distributional semantic models. In: LREC, pp. 216–223 (2014)
37. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
38. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: NIPS, pp. 6338–6347 (2017)
39. Pedregosa, F., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
40. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, pp. 1532–1543 (2014)
41. Raghavan, V.V., Wong, S.M.: A critical analysis of vector space model for information retrieval. J. Am. Soc. Inf. Sci. **37**(5), 279–287 (1986)
42. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
43. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vision **115**(3), 211–252 (2015)
44. Silberer, C., Lapata, M.: Learning grounded meaning representations with autoencoders. In: ACL, pp. 721–732 (2014)
45. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
46. Socher, R., Ganjoo, M., Manning, C.D., Ng, A.: Zero-shot learning through cross-modal transfer. In: NIPS, pp. 935–943 (2013)

47. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-ResNet and the impact of residual connections on learning. arXiv preprint arXiv:1602.07261 (2016)
48. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR, pp. 2818–2826 (2016)
49. Tenenbaum, J.B., De Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)
50. Wang, B., Yang, Y., Xu, X., Hanjalic, A., Shen, H.T.: Adversarial cross-modal retrieval. In: ACM Multimedia, pp. 154–162 (2017)
51. Wang, J.Z., Li, J., Wiederhold, G.: Simplicity: semantics-sensitive integrated matching for picture libraries. IEEE Trans. Pattern Anal. Mach. Intell. **23**(9), 947–963 (2001)
52. Wang, S., Zhang, J., Zong, C.: Associative multichannel autoencoder for multimodal word representation. In: EMNLP, pp. 115–124 (2018)
53. Wang, S., Zhang, J., Zong, C.: Learning multimodal word representation via dynamic fusion methods. In: AAAI (2018)
54. Wang, W., Ooi, B.C., Yang, X., Zhang, D., Zhuang, Y.: Effective multi-modal retrieval based on stacked auto-encoders. Proc. VLDB Endow. **7**(8), 649–660 (2014)
55. Wei, Y., et al.: Cross-modal retrieval with CNN visual features: a new baseline. IEEE Trans. Cybern. **47**(2), 449–460 (2016)
56. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. **10**(2), 207–244 (2009)
57. Wolf, T., et al.: Huggingface's transformers: State-of-the-art natural language processing. arXiv abs/1910.03771 (2019)
58. Xiao, H.: bert-as-service (2018). https://github.com/hanxiao/bert-as-service
59. Xing, C., Wang, D., Liu, C., Lin, Y.: Normalized word embedding and orthogonal transform for bilingual word translation. In: ACL, pp. 1006–1011 (2015)
60. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: NIPS, pp. 649–657 (2015)
61. Zhang, Y., Gong, B., Shah, M.: Fast zero-shot image tagging. In: CVPR, pp. 5985–5994. IEEE (2016)
62. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: CVPR, pp. 8697–8710 (2018)

An Enhanced Evaluation Framework for Query Performance Prediction

Guglielmo Faggioli[1](✉), Oleg Zendel[2](✉), J. Shane Culpepper[2](✉), Nicola Ferro[1](✉), and Falk Scholer[2](✉)

[1] University of Padova, Padova, Italy
guglielmo.faggioli@phd.unipd.it, ferro@dei.unipd.it
[2] RMIT University, Melbourne, Australia
{oleg.zendel,shane.culpepper,falk.scholer}@rmit.edu.au

Abstract. Query Performance Prediction (QPP) has been studied extensively in the IR community over the last two decades. A by-product of this research is a methodology to evaluate the effectiveness of QPP techniques. In this paper, we re-examine the existing evaluation methodology commonly used for QPP, and propose a new approach. Our key idea is to model QPP performance as a distribution instead of relying on point estimates. Our work demonstrates important statistical implications, and overcomes key limitations imposed by the currently used correlation-based point-estimate evaluation approaches. We also explore the potential benefits of using multiple query formulations and ANalysis Of VAriance (ANOVA) modeling in order to measure interactions between multiple factors. The resulting statistical analysis combined with a novel evaluation framework demonstrates the merits of modeling QPP performance as distributions, and enables detailed statistical ANOVA models for comparative analyses to be created.

1 Introduction

The Information Retrieval (IR) community has long recognized the importance of applying statistical tests to evaluation results. Although best practices continue to evolve, conference/journal guidelines and discussion papers [20,34] have led the community to appreciate the importance of a more theoretically grounded evaluation, and practitioners in IR have been urged over the years to include sound analyses using statistical tests of significance or confidence intervals in submitted manuscripts. While this has led to higher quality analytical comparisons in many IR-related fields, not all areas have adopted the practice. An example of a common IR problem that might benefit from alternative evaluation techniques is Query Performance Prediction (QPP).

The goal of QPP is to estimate the effectiveness of a retrieval system in response to a query when no relevance judgments are available [8]. The most widely-used method for evaluating QPP approaches is based on the strength of a relationship between per-topic prediction scores, and the actual per-topic system

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 115–129, 2021.
https://doi.org/10.1007/978-3-030-72113-8_8

effectiveness as measured using a standard IR effectiveness metric, usually Average Precision (AP). The association is measured using a correlation coefficient, with different papers reporting the Pearson (linear) correlation, Spearman's rank correlation, or Kendall's τ. A QPP approach that achieves a higher correlation value than another is taken to be the superior approach. This evaluation method compares QPP effectiveness at a very high level, with the performance of a QPP approach over a whole set of topics being summarized just by a correlation coefficient as a *point value*.

In order to statistically validate the results two alternatives are available. First, we can test whether or not the correlation between a predictor and the retrieval results is significantly different from zero [9,11,12,14,16,23,24,27,37,48–50]. However, this validation approach just tells us how reliable the conclusions are for a single QPP method, and does not allow two or more QPP approaches to be directly compared. Second, by relying on repeated randomized topic sampling, we can test whether or not the correlation coefficients for two different QPP methods are significantly different from each other. A statistically appropriate method to test the latter would rely on Fisher's z transformation of sample correlation coefficients. In fact, this approach was previously suggested by Hauff et al. [22] and again more recently by Roitman [32] to more reliably test significant differences in QPP model performance. However, this practice has not been adopted in published QPP work to date. Instead, a Student's t-test for the difference of means of the correlated correlation coefficients is currently the preferred approach [30,46,47]. However, it is important to note that both of these approaches are fundamentally different from the pair-wise significance test used for system retrieval effectiveness, which is now common practice in IR evaluation exercises.

Motivated by these observations, we re-examine how QPP efficacy can be analyzed using a more fine-grained approach – by modeling the performance of QPP techniques as *distributions*. This approach has also previously been applied successfully in system evaluation exercises. A distribution-based model can be constructed as follows. First, an estimate of the performance for each system-topic combination is computed using a traditional performance measure, such as AP. Then, all of the topics for a collection are used to model the performance distribution. Note that this is fundamentally different from a classical QPP evaluation approach. Indeed, even when various sampling techniques (e.g., randomization/bootstrap) are currently used in QPP, this is a re-sampling of topics, and leads to a new (aggregated) *point estimate*, e.g., Kendall's τ, for that sample. The different re-samples are then used to compute an expectation and a confidence interval for the point estimate. In contrast, when randomization/bootstrap techniques are used for the evaluation of retrieval effectiveness [40], it is topics that are re-sampled; for *each* topic a performance score such as AP is computed, and a *distribution* of performance for that sample is obtained. An aggregate of this distribution, e.g., a mean or a confidence interval, is then computed, and finally, the different re-samples are used to compute a further expectation and confidence interval for the aggregate.

In this work, we propose a methodology similar to the latter approach. Our evaluation approach has several appealing properties: it allows formal inferential statistics to be applied, which generalizes the results to the entire population of topics; it allows the behavior of a QPP approach to be more clearly isolated, for example through confidence intervals; and, it enables factor decomposition, which in turn allows us to measure the relative contributions to observed effectiveness systematically. We also incorporate recent work in retrieval effectiveness on query variation and reformulation of each topic [3,4,7,43,47] into our framework, which allows a more fine-grained sampling of retrieval performance, and to estimate interaction between systems, topics and query formulations, which is not possible using only a single point estimate.

Our work focuses on two closely related research questions:

- **RQ1**: How can detailed statistical analysis and testing be applied to QPP evaluation exercises?
- **RQ2**: What factors contribute to improving or reducing the performance of a QPP model?

The overall contribution of this paper is a new evaluation framework for QPP which models the performance of QPP methods as distributions of topics. Beside providing a statistically grounded evaluation procedure, our approach provides practitioners with new tools to carry out comprehensive analyses of QPP models.

2 Related Work

Retrieval performance can vary widely across different systems, even for a single query [8]. This has resulted in a large body of work on QPP, which is divided into two common approaches. *Pre-retrieval predictors* analyze query and corpus statistics prior to retrieval [12,23,24,27,36,48] and *post-retrieval predictors* that also analyze the retrieval results [1,2,9,14,16,31,38,46,49]. Predictors are typically evaluated by measuring the correlation coefficient between the AP values attained with relevance judgments and the values assigned by the predictor. Such evaluation methodologies are based on a *point estimate* and have been shown to be unreliable when comparing multiple systems, corpora and predictors [22,35]. Hauff et al. [22] demonstrate that higher correlation does not necessarily attest to better prediction, and used Root Mean Square Error (RMSE) in their evaluation. Hauff et al. applied methods from Meng et al. [26] to compare 2 or more correlation coefficients, and argued that to test the significance of differences in correlation between the predictors, Fisher's z transformation should be used and the Confidence Interval (CI) should be reported. When computing the CI for Pearson's linear correlation in the evaluation using multiple previously reported pre-retrieval predictors, they found that many of the predictors had overlapping CIs, and concluded that they were not significantly different from the best performing predictor. Hauff et al. focused on prediction of normalized scores that can be compared to AP using linear correlation as measured with a parametric statistic. In this work, we focus on ranking the queries based on the retrieval effectiveness, which is analogous to a rank-based correlation given by Kendall's τ as our reference for the existing evaluation framework, but many other alternatives are possible. We chose to use a rank-based correlation as it is a non-parametric statistical method, and hence makes no assumptions about the underlying distributions of the data.

Also of interest, recent work using query variations for QPP [43,47] has demonstrated that the relative prediction quality of predictors can vary with respect to the effectiveness of the queries used to represent the topics, and we explore such observation further using advanced statistical instrumentation. One principled approach that can be used in IR evaluation is ANOVA [25,33]. ANOVA is commonly used to assess the presence of statistically significant differences in mean performance observed when using different experimental conditions. This technique can be operationalized as a General Linear Mixed Model (GLMM), where a response variable, called *Data*, is linearly modeled into two parts: the experimental conditions (the $Model$) and the $Error$: $Data = Model + Error$. The $Error$ represents that part of the variance in the *Data* that the $Model$ cannot account for. The ANOVA approach is particularly useful in our work as it allows us to break down the variance observed in the data, assigning it to the factors that caused it [5,10,17,19,29,41,45]. The $Model$ often includes a subject component (which in IR evaluation often corresponds to the topic), one or more factors, which are the different experimental conditions (either the entire system, or its components - e.g., the stemmer, the stoplist and the QPP model), and possibly their interactions. If all the possible combinations of factors are applied to all subjects, this is a *Factorial/Crossed Design*, and its factors are called *Crossed Factors*. Specific factors might be *nested* inside others: in the following analyses, query formulations are a nested factor of the topic, since each formulation represents a single topic and cannot be used to represent others. To compare the *effect size* of different factors, which cannot be done by looking only at the F-statistic or p-value, the Strength of Association (SOA) is reported, measured as ω^2, and is the factor significance, bounded between [0, 1]. The larger ω^2 is, the greater the impact is for factor levels to the response variable.

3 Experimental Analysis

3.1 Experimental Setup

In our analyses, we use the TREC Robust 2004 (Robust04) Ad Hoc [44] collection. The Robust04 ad hoc track consists of approximately 528K documents from TREC disks 4 & 5, minus the Congressional Record from the TIPSTER corpus, and contains 249 topics with at least one relevant document in the QREL file. We enrich the set of queries for the corpus using publicly available human-curated query variants for each topic [6].[1] Our experiments use a Grid of Points (GoP) of runs as described by Ferro and Harman [18], using 4 different stoplists

[1] http://culpepper.io/publications/robust-uqv.txt.gz.

Table 1. A summary of QPP models used in this work.

QPP model	Description
	Pre-retrieval
SCQ [48]	Measures similarity based on *cf.idf* to the corpus, summed over the query terms
AvgSCQ [48]	SCQ normalized by the query length
MaxSCQ [48]	The query term with maximal SCQ score
SumVAR [48]	Measures the *cf.idf* variability of the query terms in the corpus
AvgVAR [48]	Variability normalized with the query length
MaxVAR [48]	The query term with maximal variability
AvgIDF [13]	The mean *idf* value of the query terms
MaxIDF [36]	The query term with maximal *idf* value
	Post-retrieval
Clarity [12]	Measures the divergence between the Language Model (LM) constructed over top documents in the result list to the LM of the entire corpus
NQC [39]	Measures the standard deviation of the top documents scores in the retrieval list
WIG [50]	Measures the difference between the mean retrieval score of the top retrieved documents and the score of the entire corpus
SMV [42]	Scores the queries based on a combination of the scores standard deviation and magnitude
UEF [37]	Prediction framework that is based on the similarity of the initial result list with the list re-ranked using a Relevance Model (RM), scaled by an estimator of the RM quality. In this work we scale the RM with the existing post-retrieval predictors: UEF(Clarity), UEF(NQC), UEF(WIG) and UEF(SMV)

(`atire`, `zettair`, `indri`, `lingpipe`), plus the `no stop` approach and 2 different stemmers, (`lovins`, `porter`) plus a nostem approach. All the runs were produced using the query-likelihood model [28], and repeated 15 times. We test 16 QPP models (12 + 4 UEF-based methods) for our analyses, which are summarized in Table 1. Our goal was to choose representative and well known system configurations and QPP models, and the evaluation framework is not limited to any specific configuration. So it can easily be extended by others for further experiments in the future. In total, 240 different predictor-system combinations were generated for the Robust04 collection. The pre-retrieval approaches are parameter-free and do not require tuning. For the parameters of the post-retrieval predictors we used fixed settings that have been demonstrated to be effective for the Robust04 collection previously [37,39,42]. We apply Average Precision (AP) to measure the effectiveness of the different retrieval pipelines, as our primary goal is to be consistent with previous evaluation exercises, as Average Precision (AP) was the most common effectiveness metric used in prior QPP work.

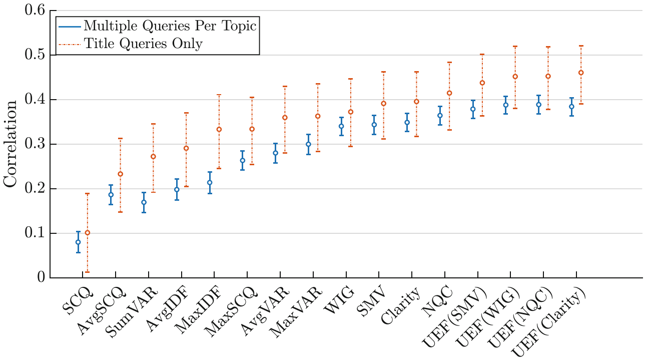


Fig. 1. Prediction quality of the selected QPP models on Robust04 (Confidence Intervals computed with Kendall's τ), using either title queries or all available formulations. (Color figure online)

3.2 Traditional QPP Evaluation Using Correlations

Prior work on QPP has relied primarily on a single evaluation paradigm. Given a set of topics (information needs), where each topic is represented by a single query, a single retrieval method, and a single document corpus, the prediction quality of the predictors is evaluated as follows:

1. Retrieval effectiveness of the queries is measured with a common IR metric, usually AP or possibly Normalized Discounted Cumulated Gain (nDCG) to induce a ranking of the queries. This ordering serves as the ground truth in the evaluation process.
2. The QPP method is applied to the queries, which generates a candidate list where the queries are ranked by their prediction values.
3. A correlation coefficient is computed between the ground truth list and the candidate list produced by the predictor.
4. The correlation coefficients of different predictors are then compared, with an underlying assumption that a higher correlation value attests to the superior quality of a predictor.

The correlation coefficient is often measured as Pearson's r for linear correlation, Kendall's τ, and/or Spearman's ρ for the monotonic rank correlation.

Figure 1 shows the performance of 16 different QPP models when using this common evaluation approach – Kendall's τ correlation in this case – with 95% confidence intervals shown as well. In this example, the results are generated for a specific retrieval pipeline, using the `indri` stoplist and `porter` stemmer. To compute the confidence intervals (at significance level $\alpha = 0.05$), we used a bias-corrected and accelerated bootstrap procedure with 10,000 samples. Observe that when using title queries only (orange bars), there is a large degree of overlap between the different QPP approaches. Similar results were observed when using all of the other pipelines described in this work. The pairwise comparison

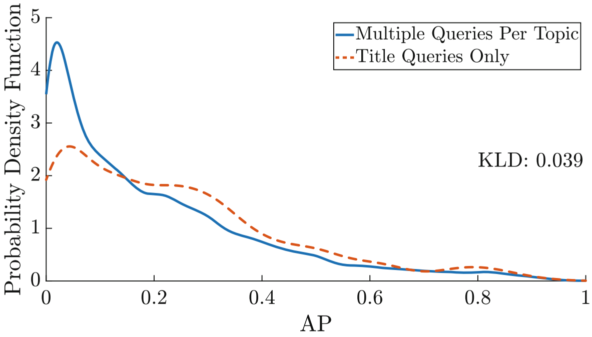


Fig. 2. Comparison between the AP score distributions of title-only queries and multi-query topic formulations. (Color figure online)

using the data from Fig. 1 (title queries only, p-values omitted due to space constraints), shows that 57 pairs of predictors are found to be statistically significantly different, out of 120 total pairs of QPP models (47.5%). In particular, among the best performing predictors, UEF(Clarity) is not statistically different from UEF(WIG), UEF(NQC), UEF(SMV), Clarity and NQC. This suggests that using confidence intervals does indeed make it difficult to decide which QPP system is the best performing one, as suggested by Hauff et al. [22].

In addition to using the traditional title queries, we also explore the scenario of using multiple formulations, which allows us to produce replicas for the same experimental conditions (i.e., the retrieval system or the QPP model used) on the same subject (i.e., the topic). While the performance is generally lower when using multiple topic formulations (the blue bars shown in Fig. 1), there is a high degree of similarity between the ordering of the QPP models for multiple query formulations to the ordering for title-only (Kendall's tau correlation between using title-only versus multiple queries per topic is 0.98, $p < 0.0001$). Overall, the statistically induced bootstrap intervals are substantially larger if a traditional title-only evaluation approach is used, which makes it less suitable for determining if any single system is a clear winner, while using multiple queries does induce smaller intervals and better discriminative power between the QPP approaches. Even if, as shown, using query variants does not dramatically impact the ranking of QPP models, it is nevertheless important to consider whether adding variants has an impact on the distribution of the raw AP scores. The Mean Average Precision (MAP) values are 0.211 and 0.254 for the set of all query formulations and title queries only, respectively, and thus are quite consistent. Figure 2 shows the Probability Density Function (PDF) for the AP scores for the two scenarios – title-only (red line) and multiple queries per topic (blue line). The Kullback-Leibler Divergence (KLD), a measure of the similarity between the two distributions, is 0.039. In summary, the distributions are similar and thus the introduction of the multiple formulations for each topic does not appear to skew the overall AP score distribution.

3.3 ANOVA Modeling and Analysis of QPP

To support a more detailed analysis of QPP methods and associated factors, we now explore the use of ANOVA, which can be achieved by modifying steps 3 and 4 of the traditional QPP evaluation process shown above. Instead of computing the correlations between the complete lists, we measure the difference, for each query, in the rank position assigned by a QPP method and the ground truth rank position assigned by AP. Ties in ranks are broken using the average of tie rank spans, as is the default in many statistical applications [21]. Other tie breaking rules were also considered but initial investigation led to consistent final results, so are not reported here. Observe that this transitions us from *point estimates* of a single correlation value for the two lists over a whole set of topics to a *distribution* of the rank differences between the two lists for each query in the set. In order to scale the scores to the range $[0, 1]$ we divide them by the number of samples. The error, labeled as AP induced scaled Absolute Rank Error (sARE$_{AP}$), for each query is:

$$\text{sARE}_{AP}(q_i) := \frac{|r_i^p - r_i^e|}{|Q|}, \tag{1}$$

where r_i^p and r_i^e are the ranks assigned by the predictor and the evaluation metric respectively for query i; Q is the set of queries. If we still require the single point estimate of the prediction quality for each predictor $\mathcal{P}$, we can calculate the AP induced scaled Mean Absolute Rank Error (sMARE$_{AP}$) as follows:

$$\text{sMARE}_{AP}(\mathcal{P}) := \frac{1}{|Q|} \sum_{q_i \in Q} \text{sARE}_{AP}(q_i). \tag{2}$$

Note that sMARE$_{AP}$ can be seen as a derivation of *Spearman's Footrule distance*, making it a metric for the full rankings instead of a correlation. Among the properties of Spearman's Footrule distance, Diaconis and Graham [15] list that it is bounded between $[0, \lfloor 0.5n^2 \rfloor]$, where n is the length of the ranking. Since both sARE$_{AP}$ and sMARE$_{AP}$ are normalized by the number of queries, sMARE$_{AP}$ is bounded between [0, 0.5]. To demonstrate the agreement between the proposed evaluation method with existing evaluation practices from a high-level (point estimate) perspective, we use the QPP methods over the Robust04 title queries. Figure 3 plots the ranking of the predictors based on the median of the point estimates for each predictor for all 15 system configurations which is simply the median of the Kendall's τ correlation for the traditional evaluation approach and the median of sMARE$_{AP}$ for our evaluation approach. Each predictor consists of 15 values that represent the prediction quality. Though the directionality of the two approaches is inverted, the ranking of the predictors clearly agrees on the overall rank ordering. The corresponding box-plots also demonstrate the similarity of the variance estimate. In order to validate the agreement we computed the Pearson's correlation coefficient over the point estimates for the predictors for each of the 15 system configurations. The resulting correlations coefficients were all -0.99 or higher ($p < 0.0001$ for each).

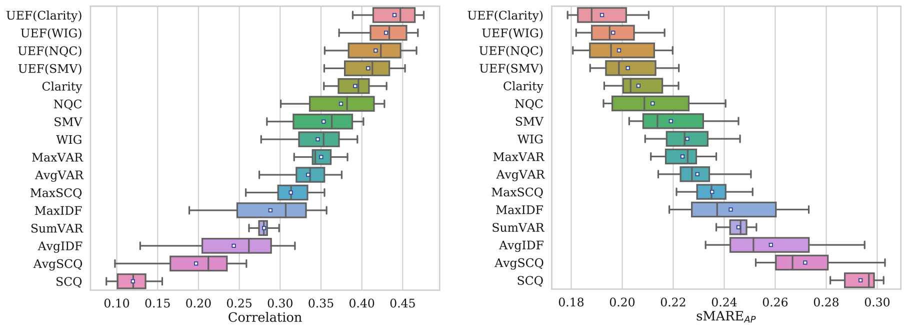


Fig. 3. Prediction quality when measuring correlation with Kendall's τ and sMARE$_{AP}$ for Robust04 title-only queries and 15 different system configurations. The line inside the interquartile range (IQR) is the median, and the white square is the mean.

Table 2. MD0$_{micro}$ ANOVA on the Robust04 collection. Topics are represented with the title queries. SS: Sum of Squares; DF: Degrees of Freedom; MS: Mean Square; F: F statistics.

Source	SS	DF	MS	F	p-value	$\hat{\omega}^2_{\langle fact\rangle}$
Topic	876.524	248	3.534	168.136	<0.001	0.410
Stoplist	1.185	4	0.296	14.095	<0.001	0.001
Stemmer	5.218	2	2.609	124.108	<0.001	0.004
QPP model	46.569	15	3.105	147.691	<0.001	0.036
Error	1250.538	59490	0.021			
Total	2180.034	59759				

We are in a position to introduce our first ANOVA model which will enable a more comprehensive experimental analysis of the results.

$$y_{iqrs} = \mu + \tau_i + \gamma_q + \delta_r + \zeta_s + \varepsilon_{iqrs} \qquad (\text{MD0}_{micro})$$

where: $y_{i...}$ is the performance (sARE$_{AP}$) on the i-th topic (using the specified QPP pipeline); μ is the *grand mean*; τ_i is the effect of the i-th topic (represented with the title query formulation); γ_q, δ_r, and ζ_s are the effect of the q-th stoplist, the r-th stemmer, and the s-th QPP model; ε_{iqrs} is the error component. Table 2 summarizes the ANOVA results of our first experiment. It can be seen that the stoplist, the stemmer, and the QPP model have a small size effect, while the topic effect is large (indicating that most of the performance of the QPP depends on the chosen topic). Based on the results of this analysis, we also ran a Tukey's Honestly Significant Difference (HSD) post-hoc analysis to test for pairwise differences. Figure 4 shows the Tukey's HSD confidence intervals for sMARE$_{AP}$ over the different QPP models.

When comparing Fig. 1 (orange bars) and Fig. 4, we can observe that there is less overlap between the CIs, in particular, we observe that, by computing the

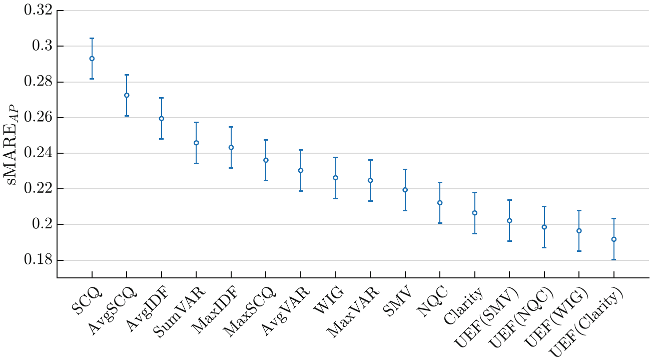


Fig. 4. Confidence Intervals of sMARE_{AP} from MD0_{micro} on the ROBUST04 title queries.

p-values for the pairwise comparisons, out of 120 pairs of predictors, 96 of them are significantly different (80.0%). Thus, compared to the results observed for the bootstrap-based approach, we are able to differentiate between 68.4% more pairs of predictors. In this case, the top performing cluster includes UEF(WIG), UEF(SMV), UEF(NQC), and UEF(Clarity).

The "Topic" factor, as Table 2 suggests, is responsible for the largest part of the variance; this is in line with results from IR effectiveness evaluation (see for example Tague-Sutcliffe and Blustein [41]). Thus, the estimation of the performance for a specific QPP model can vary significantly as it is dependent on properties of the underlying collection (performance differences in topics/queries). By removing the contribution of the topics from the global variance, ANOVA removes any volatility in the underlying experimental data allowing the relative performance of predictors to be compared more precisely. When using only correlations aggregated across all topics, such information is lost, while an ANOVA analysis facilitates more discriminative performance comparisons between systems by systematically accounting for each factor separately.

3.4 ANOVA Modeling of Multiple Queries and Interactions

One of the most interesting aspects of our framework is the capability to compute the effect sizes of interactions between factors. This is achieved using MD1_{micro}

$$\begin{aligned} y_{ijqrs} = \mu + \tau_i + \nu_{j(i)} + \gamma_q + \delta_r + \zeta_s + (\tau\gamma)_{iq} + (\tau\delta)_{ir} + (\tau\zeta)_{is} \\ + (\nu\gamma)_{j(i)q} + (\nu\delta)_{j(i)r} + (\nu\zeta)_{j(i)s} + (\gamma\delta)_{qr} + (\gamma\zeta)_{qs} + (\delta\zeta)_{rs} + \varepsilon_{ijqrs} \end{aligned} \tag{MD1_{micro}}$$

which extends MD0_{micro} to include $\nu_{j(i)}$ to represent the effect of the j-th query formulation for the i-th topic. Moreover, this model considers all of the possible two-way interactions which are now computable using the replicates provided by the multi-query topic formulations.

Table 3. MD1$_{micro}$ ANOVA applied on Robust04 collection. ω^2 for non-significant factors is ill-defined and thus not reported.

Source	SS	DF	MS	F	p-value	$\hat{\omega}^2_{\langle fact \rangle}$
Topic	1840.082	248	7.420	1293.936	<0.001	0.518
Formulation (Topic)	1746.213	996	1.753	305.749	<0.001	0.504
Stoplist	1.179	4	0.295	51.402	<0.001	0.001
Stemmer	10.622	2	5.311	926.188	<0.001	0.006
QPP model	305.796	15	20.386	3555.233	<0.001	0.151
Topic*Stoplist	40.224	992	0.041	7.071	<0.001	0.020
Topic*Stemmer	154.200	496	0.311	54.216	<0.001	0.081
Topic*QPP model	2051.688	3720	0.552	96.182	<0.001	0.542
Frm.*Stoplist	87.110	3984	0.022	3.813	<0.001	0.036
Frm.*Stemmer	312.955	1992	0.157	27.398	<0.001	0.150
Frm.*QPP model	3348.894	14940	0.224	39.091	<0.001	0.656
Stoplist*Stemmer	0.059	8	0.007	1.288	0.2444	–
Stoplist*QPP model	0.901	60	0.015	2.618	<0.001	<0.001
Stemmer*QPP model	4.850	30	0.162	28.195	<0.001	0.003
Error	1555.757	271312	0.006			
Total	11460.530	298799				

Table 3 presents the ANOVA summary statistics for Ex. MD1$_{micro}$. In this analysis we add the query formulations as a nested factor for each topic, in this case we randomly chose 5 for each topic.[2] The table empirically shows that the largest differences in QPP performance are due to the topics, and their formulations. While this is a well-known phenomenon, our model is able to explicitly quantify the magnitude of this effect. The effect for the QPP factor is medium-sized. It is important to note that the dimension of the effect is due to the wide variety of QPP models (and their performance) taken into account. For example, a practitioner wishing to evaluate new QPP models may observe a smaller ω^2 for the QPP model factor if the relative performance differences between the models being compared is less substantial.

We have also ran similar experiments using alternative models with fewer factors, but found that including all of the possible interactions is the most informative. For example, the effect size of stoplists and stemmers are both small but still significant. This suggests that stemmers and stoplists may affect overall prediction quality, and practitioners should consider all possible factors when comparing and contrasting QPP performance for a corpus.

We are now in a position to observe the interaction between topics (and their query formulations) and the predictors, which is large, indicating that important differences between QPP model performance exists within reformulations of a

[2] The topic with the minimal number of query formulations had 5 formulations.

single topic. Finding the QPP model where interactions are smallest is valuable in practice as this corresponds to be choosing a model that is most robust to query reformulation. Additionally, this enables a series of additional analyses, such as a failure analysis for topics with the largest interaction with a QPP model. There are many additional factors that could influence the performance of various QPP approaches, beyond the ones included in our model. For example, alternative ranking functions or evaluation metrics can also be used with sMARE, and may provide additional experimental evidence and insights into performance differences between various QPP models in the future.

4 Conclusion

We have presented a novel evaluation framework for QPP. The framework estimates the performance of QPP on every topic as the distance between its predicted rank - computed using the QPP – and the expected one – measured through AP (or any other traditional IR measure). This allows us to obtain a distribution of performance for the QPP over the different topics. Furthermore, our framework makes use of multiple query formulations for each topic to enhance the power of our analyses. Together, the use of multiple query formulations and the distributional representation of the performance enables carrying out more accurate studies. In particular, we showed that it is possible to rely on the statistical properties of ANOVA and corresponding post hoc procedures to better identify pairs of QPP approaches that are statistically significantly different. The newly proposed framework also enables the analysis of interaction effects for QPP models and topics, allowing failure analyses and a deeper understanding into how a QPP model works. Our framework can be extended and adapted to different investigation needs. For example, in an academic setting, you may add further factors to the model such as tokenizers, query expansion components, or ranking functions to deepen the investigation into the factors that influence QPP performance. In industrial deployment settings, comparisons between competing QPP techniques may require an ANOVA model consisting of only two factors: topics and QPP approaches. This simple two-way ANOVA is sufficient to determine if QPP models are significantly different, and has the added benefit of relying on a statistically-sound and easy to deploy framework. In future work, we plan to study additional components of the evaluation framework, such as the impact of the ranking methods which are used to establish “ground truth” performance; new factors that influence QPP systems such as the ranking approach used in the post-retrieval QPP; and the effects of using multiple corpora, in order to more comprehensively model and understand corpus and QPP interactions. In order to aid reproducibility of our results, the code for our proposed evaluation framework is publicly available.[3]

[3] https://github.com/Zendelo/QPP-EnhancedEval.

Acknowledgments. We thank the reviewers for their comments. The work is partially funded by the DAta BenchmarK for Keyword-based Access and Retrieval (DAKKAR) Starting Grants project sponsored by University of Padua and Fondazione Cassa di Risparmio di Padova e di Rovigo. This work was also partially supported by the Australian Research Council's Discovery Projects Scheme (DP190101113).

References

1. Amati, G., Carpineto, C., Romano, G.: Query difficulty, robustness, and selective application of query expansion. In: McDonald, S., Tait, J. (eds.) ECIR 2004. LNCS, vol. 2997, pp. 127–137. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24752-4_10
2. Aslam, J.A., Pavlu, V.: Query hardness estimation using Jensen-Shannon divergence among multiple scoring functions. In: Amati, G., Carpineto, C., Romano, G. (eds.) ECIR 2007. LNCS, vol. 4425, pp. 198–209. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71496-5_20
3. Bailey, P., Moffat, A., Scholer, F., Thomas, P.: UQV100: a test collection with query variability. In: Proceedings SIGIR, pp. 725–728 (2016)
4. Bailey, P., Moffat, A., Scholer, F., Thomas, P.: Retrieval consistency in the presence of query variations. In: Proceedings of the SIGIR, pp. 395–404 (2017)
5. Banks, D., Over, P., Zhang, N.F.: Blind men and elephants: six approaches to TREC data. Inf. Retrieval **1**(1–2), 7–34 (1999)
6. Benham, R., Culpepper, J.S.: Risk-reward trade-offs in rank fusion. In: Proceedings ADCS, pp. 1:1–1:8 (2017)
7. Benham, R., Mackenzie, J., Moffat, A., Culpepper, J.S.: Boosting search performance using query variations. ACM Trans. Inf. Syst. **37**(4), 41:1–41:25 (2019)
8. Carmel, D., Yom-Tov, E.: Estimating the Query Difficulty for Information Retrieval. Morgan & Claypool Publishers, San Rafael (2010)
9. Carmel, D., Yom-Tov, E., Darlow, A., Pelleg, D.: What makes a query difficult? In: Proceedings of the SIGIR, pp. 390–397 (2006)
10. Carterette, B.A.: Multiple testing in statistical analysis of systems-based information retrieval experiments. ACM Trans. Inf. Syst. **30**(1), 4:1–4:34 (2012)
11. Chifu, A.G., Laporte, L.é., Mothe, J., Ullah, M.Z.: Query performance prediction focused on summarized letor features. In: Proceedings of the SIGIR, pp. 1177–1180 (2018)
12. Cronen-Townsend, S., Zhou, Y., Croft, W.B.: Predicting query performance. In: Proceedings of the SIGIR, pp. 299–306 (2002)
13. Cronen-Townsend, S., Zhou, Y., Croft, W.B.: A language modeling framework for selective query expansion. Technical report, Center for Intelligent Information Retrieval, University of Massachusetts (2004)
14. Cummins, R.: Document score distribution models for query performance inference and prediction. ACM Trans. Inf. Syst. **32**(1), 2:1–2:28 (2014)
15. Diaconis, P., Graham, R.L.: Spearman's footrule as a measure of disarray. J. R. Stat. Soc. **39**(2), 262–268 (1977)
16. Diaz, F.: Performance prediction using spatial autocorrelation. In: Proceedings of the SIGIR, pp. 583–590 (2007)
17. Ferro, N., Fuhr, N., Maistro, M., Sakai, T., Soboroff, I.: CENTRE@CLEF 2019. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11438, pp. 283–290. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15719-7_38

18. Ferro, N., Harman, D.: CLEF 2009: Grid@CLEF pilot track overview. In: Peters, C., et al. (eds.) CLEF 2009. LNCS, vol. 6241, pp. 552–565. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15754-7_68
19. Ferro, N., Silvello, G.: A general linear mixed models approach to study system component effects. In: Proceedings of the SIGIR, pp. 25–34 (2016)
20. Fuhr, N.: Some common mistakes in IR evaluation, and how they can be avoided. SIGIR Forum **51**(3), 32–41 (2017)
21. Gibbons, J.D., Chakraborti, S.: Nonparametric Statistical Inference, 5th edn. Chapman & Hall/CRC, Taylor and Francis Group, Boca Raton (2011)
22. 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). https://doi.org/10.1007/978-3-642-00958-7_28
23. Hauff, C., Hiemstra, D., de Jong, F.: A survey of pre-retrieval query performance predictors. In: Proceedings of the CIKM, pp. 1419–1420 (2008)
24. He, B., Ounis, I.: Inferring query performance using pre-retrieval predictors. In: Apostolico, A., Melucci, M. (eds.) SPIRE 2004. LNCS, vol. 3246, pp. 43–54. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30213-1_5
25. Maxwell, S., Delaney, H.D.: Designing Experiments and Analyzing Data. A Model Comparison Perspective, 2nd edn. Lawrence Erlbaum Associates, Mahwah (2004)
26. Meng, X.L., Rosenthal, R., Rubin, D.B.: Comparing correlated correlation coefficients. Psychol. Bull. **111**(1), 172–175 (1992)
27. Mothe, J., Tanguy, L.: Linguistic features to predict query difficulty. In: Proceedings of the SIGIR, pp. 7–10 (2005)
28. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: Proceedings of the SIGIR, pp. 275–281 (1998)
29. Robertson, S.E., Kanoulas, E.: On per-topic variance in IR evaluation. In: Proceedings of the SIGIR, pp. 891–900 (2012)
30. Roitman, H.: An extended query performance prediction framework utilizing passage-level information. In: Proceedings of the SIGIR, pp. 35–42 (2018)
31. Roitman, H.: Query performance prediction using passage information. In: Proceedings of the SIGIR, pp. 893–896 (2018)
32. Roitman, H.: ICTIR tutorial: modern query performance prediction: theory and practice. In: Proceedings of the SIGIR, pp. 195–196 (2020)
33. Rutherford, A.: ANOVA and ANCOVA. A GLM Approach, 2nd edn. Wiley, New York (2011)
34. Sakai, T.: Topic set size design. Inf. Retrieval J. **19**(3), 256–283 (2016)
35. Scholer, F., Garcia, S.: A case for improved evaluation of query difficulty prediction. In: Proceedings of the SIGIR, pp. 640–641 (2009)
36. Scholer, F., Williams, H.E., Turpin, A.: Query association surrogates for web search. J. Assoc. Inf. Sci. Technol. **55**(7), 637–650 (2004)
37. Shtok, A., Kurland, O., Carmel, D.: Using statistical decision theory and relevance models for query-performance prediction. In: Proceedings of the SIGIR, pp. 259–266 (2010)
38. Shtok, A., Kurland, O., Carmel, D.: Query performance prediction using reference lists. ACM Trans. Inf. Syst. **34**(4), 19:1–19:34 (2016)
39. Shtok, A., Kurland, O., Carmel, D., Raiber, F., Markovits, G.: Predicting query performance by query-drift estimation. ACM Trans. Inf. Syst. **30**(2), 1–35 (2012)
40. Smucker, M.D., Allan, J., Carterette, B.A.: A comparison of statistical significance tests for information retrieval evaluation. In: Proceedings of the CIKM, pp. 623–632 (2007)

41. Tague-Sutcliffe, J.M., Blustein, J.: A statistical analysis of the TREC-3 data. In: Proceedings of the TREC, pp. 385–398 (1994)
42. Tao, Y., Wu, S.: Query performance prediction by considering score magnitude and variance together. In: Proceedings of the CIKM, pp. 1891–1894 (2014)
43. Thomas, P., Scholer, F., Bailey, P., Moffat, A.: Tasks, queries, and rankers in pre-retrieval performance prediction. In: Proceedings of the ADCS (2017)
44. Voorhees, E.M.: Overview of the TREC 2004 robust track. In: Proceedings of the TREC (2004)
45. Voorhees, E.M., Samarov, D., Soboroff, I.: Using replicates in information retrieval evaluation. ACM Trans. Inf. Syst. **36**(2), 12:1–12:21 (2017)
46. Zamani, H., Croft, W.B., Culpepper, J.S.: Neural query performance prediction using weak supervision from multiple signals. In: Proceedings of the SIGIR, pp. 105–114 (2018)
47. Zendel, O., Shtok, A., Raiber, F., Kurland, O., Culpepper, J.S.: Information needs, queries, and query performance prediction. In: Proceedings of the SIGIR, pp. 395–404 (2019)
48. Zhao, Y., Scholer, F., Tsegay, Y.: Effective pre-retrieval query performance prediction using similarity and variability evidence. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 52–64. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78646-7_8
49. Zhou, Y., Croft, W.B.: Ranking robustness: a novel framework to predict query performance. In: Proceedings of the CIKM, pp. 567–574 (2006)
50. Zhou, Y., Croft, W.B.: Query performance prediction in web search environments. In: Proceedings of the SIGIR, pp. 543–550 (2007)

Open-Domain Conversational Search Assistant with Transformers

Rafael Ferreira(✉), Mariana Leite(✉), David Semedo(✉), and Joao Magalhaes(✉)

NOVA LINCS, School of Science and Technology,
NOVA School of Science and Technology, Almada, Portugal
{rah.ferreira,me.leite}@campus.fct.unl.pt, {df.semedo,jmag}@fct.unl.pt

Abstract. Open-domain conversational search assistants aim at answering user questions about open topics in a conversational manner. In this paper we show how the Transformer architecture [30] achieves state-of-the-art results in key IR tasks, leveraging the creation of conversational assistants that engage in open-domain conversational search with single, yet informative, answers. In particular, we propose an open-domain abstractive conversational search agent pipeline to address two major challenges: first, conversation context-aware search and second, abstractive search-answers generation. To address the first challenge, the conversation context is modeled with a query rewriting method that unfolds the context of the conversation up to a specific moment to search for the correct answers. These answers are then passed to a Transformer-based re-ranker to further improve retrieval performance. The second challenge, is tackled with recent Abstractive Transformer architectures to generate a digest of the top most relevant passages. Experiments show that Transformers deliver a solid performance across all tasks in conversational search, outperforming the best TREC CAsT 2019 baseline.

Keywords: Conversational search · Transformers · Query rewriting · Re-ranking · Answer generation

1 Introduction

Conversational search systems are an emerging research topic, and the natural evolution of the traditional search paradigm, allowing for a more natural interaction between users and search systems. Building intelligent systems able to establish and develop meaningful conversations is one of the key goals of AI and the ultimate goal of natural language research [9]. The interactions between a user and conversational systems have been studied in [32], which showed that users are willing to utilise conversational assistants as long as their needs are met with success. However, conversational search assistants still put a considerable burden on users that have to go through a list of documents, or passages, to find the information they need.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 130–145, 2021.
https://doi.org/10.1007/978-3-030-72113-8_9

We depart from this document-based approach to conversational search, and propose an open-domain abstractive conversational assistant that is aware of the context of the conversation to generate a single and informative search-answer. We argue that by doing so, we can capture in one single and short answer the information contained on several relevant documents. Moreover, we show that Transformer architectures [30] outperform the state-of-the-art results across all the steps of the conversational system pipeline. Hence, the core contributions of this paper are twofold: first, we show that one can tightly integrate different Transformers to deliver an end-to-end conversational search pipeline with state-of-the-art results; second, abstractive answer generation can effectively compress the information of several retrieved passages into a short answer. These contributions are rooted in the groundbreaking architecture of the Transformer [30] that leverages attention mechanisms to model complex interactions between sequence data. In particular, we explore Transformer's advantages to: (a) capture complex relations between conversation turns to rewrite a query in the middle of a conversation; (b) to look into the interactions between words in a conversation query and a candidate passage; and (c) to compress multiple retrieved passages into one single, yet informative, search-answer. The final result, is a complete conversational search assistant leveraged by the Transformer architecture.

In the following section, we discuss the related work. In Sect. 3 we detail the Transformer-based conversational search pipeline: the conversational query rewriting, the re-ranker, and abstractive answer generation. Evaluation is performed in Sect. 4 and Sect. 5 presents the key takeaway messages.

2 Related Work

Open-domain conversational search systems must account for the dialog context to provide a relevant passage. While research on interactive search systems has started long ago [1,4,23], the recent interest in having intelligent conversation assistants (e.g. Alexa, SIRI), has re-ignited this research field. Recent models [9,17,25,31] leverage large open-domain collections (e.g. Wikipedia) to learn rich language-models using self-supervised neural networks. The applicability of these models in conversational search is twofold: grasping the dialog context and passage re-ranking. Recently, the TREC CAsT (Conversational Assistant Track) [6] task introduced a multi-turn passage retrieval dataset, enabling the development and evaluation of such models.

Conversational context-aware search models need to (a) keep track of the dialog context, and (b) select the most relevant passage. To address (a), one approach is to perform query rewriting to obtain context-independent queries. [10] observed that manually rewritten queries from QuAC [2] had enough context to be independently understandable. To automate the process, a sequence-to-sequence (seq2seq) model with attention and a copy mechanism was proposed. The model is given as input a sequence with the full conversation history and the query to be rewritten. In [31], a BERT model [7] is given as input a sequence of all terms of the current and previous queries, and is then fine-tuned on a binary term classification task. Also using both the query and conversation history, in [17], a pre-trained T5 model [26] is fine-tuned on CANARD [10] to construct the context-independent query, and achieved state-of-the-art performance on the query-rewriting task. Task (b) is commonly addressed through re-ranking. Large pre-trained Transformer models, such as BERT [7], RoBERTa [18], and XLNet [36], have been widely adopted for re-ranking due to their generalisation capabilities. Examples of this are present in [12,21,22], where a Transformer-based model is fine-tuned on the question-answering relevance classification task.

Given the dialogue context, the agent must generate a natural language response. In chit-chat dialogue generation, most approaches use an encoder-decoder neural architecture that first encodes utterances and then the decoder generates a response [15,16,28,29,39]. In [15] and [16], reinforcement learning is used to overcome uninformative and general responses of standard seq2seq models. Another alternative is retrieval-based dialogue generation, in which the generator takes as input retrieved candidate documents to improve the comprehensiveness of the generated answer [28,39]. These approaches require a large dataset with annotated dialogues, which is not feasible in our scenario. Alternatively, Transformer models have shown to be highly effective generative language models [14,26,38]. While both T5 [26] and BART [14] are general language models, PEGAGUS [38] focuses on abstractive summarisation, and obtained state-of-the-art results on 12 summarisation tasks.

3 Transformers-Based Conversational Search Assistant

In this section we formulate the open-domain conversational search task and describe the conversational assistant retrieval and answer generation components. The conversational search task is formally defined by a sequence of natural language conversational turns for a topic T, with queries q. For each conversation turn $T = \{q_1, ...q_i, ...q_n\}$, the conversational search task is to find relevant passages p_k for each query q_i, satisfying the user's information need for that turn according to the conversational context. The proposed approach uses a four-stage architecture: (a) context tracking, (b) retrieval, (c) re-ranking, and (d) answer generation. An overview of the system's architecture can be seen in Fig. 1 which we will detail in the following sections.

3.1 Conversational Query Rewriting Transformer

Due to the evolving nature of a conversational session, the current query may not include all the information needed to retrieve the answer that the user is looking for. This challenge is illustrated in the conversation presented in Table 1: in conversation turn 2, the system needs to understand that "its" refers to "Lucca's" (explicit coreference) and in turn 3, where the important monuments should be focused in Lucca, although there is no direct evidence (implicit coreference), which makes the task even more challenging. We tackle this challenge by rewriting queries, using previous turns, making the current query context-independent.

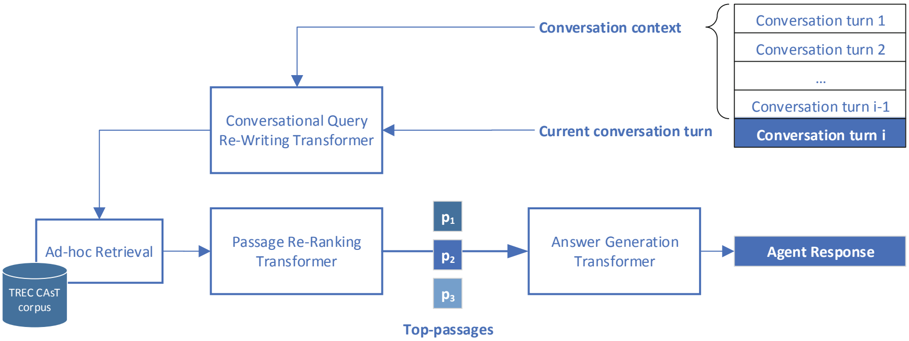


Fig. 1. The proposed Transformer-based conversational search assistant.

Table 1. Conversation example about a specific topic, in this case the city of Lucca.

Turn	Conversational query	Context-independent query
1	How is the climate in Lucca?	How is the climate in Lucca?
2	Tell me about its origins	Tell me about Lucca's origins
3	What monuments should I visit?	What monuments should I visit in Lucca?

To perform the query rewriting task, we need a model capable of performing coreference resolution and include context from previous turns. The Text-to-text Transfer Transformer (T5) [26] can be fine-tuned to reformulate conversational queries [17] by providing as input the sequence of conversational queries and passages, and as target, the rewritten query. The training input sequence is constructed as:

$$“q_i\ [CTX]\ q_1\ p_1\ [TURN]\ q_2\ p_2\ [TURN]\ \ldots\ [TURN]\ q_{i-1}\ p_{i-1}”, \tag{1}$$

where i is the current turn, q is a query, p_k is a passage retrieved from the index by the retrieval model, and $[CTX]$ and $[TURN]$ are special tokens. $[CTX]$ is used to separate the current query from the context (previous queries and passages) and $[TURN]$ is used to separate the historical turns (query-passage pair).

3.2 Passage Re-Ranking Transformer

With the new pre-trained neural language models, such as BERT [7] and others [18,36], it is possible to generate contextual embeddings for a sentence and each of its tokens. These embeddings can be used as input to a model to perform passage re-ranking [21,22]. This re-ranking step allows going beyond term matching, as the model has some understanding of both individual terms semantics as well as their interactions between queries and passages. As such, it is able to judge more thoroughly if a passage is relevant to a query.

Following this rationale, we tackle the passage re-ranking task with a BERT model [7], fine-tuned on the passage ranking task [21], through a binary relevance classification task, where positive examples are relevant passages, and negative examples are non-relevant passages. To obtain the embedding of the query q, and passage p, a sequence with N tokens is given as input to BERT:

$$emb = BERT(\text{``}[CLS]\ q\ [SEP]\ p\text{''}), \tag{2}$$

where $emb \in \mathbb{R}^{N \times H}$ (H is BERT embedding's size) is the embeddings matrix of all tokens, and *[CLS]* and *[SEP]* are special tokens in BERT's vocabulary, representing the classification and separation tokens, respectively. From *emb* we extract the embedding of the first token, which corresponds to the embedding of the *[CLS]* token, $emb_{[CLS]} \in \mathbb{R}^H$. This embedding is then used as input to a single layer feed-forward neural network (FFNN), followed by a *softmax*, to obtain the probability of the passage being relevant to the query:

$$P(p|q) = softmax(\text{FFNN}(emb_{[CLS]})). \tag{3}$$

With $P(p|q)$ calculated for each passage p given a query q, the final rank is obtained by re-ranking according to the probability of being relevant.

3.3 Abstractive Search-Answer Generation Transformer

Having identified a set of candidate passages according to the scores given by the re-ranker model (Eq. 3), the goal is to generate a natural language response that combines the information comprised in each of the passages. To address this, we follow an abstractive summarisation approach, which unlike extractive summarisation that just selects existing sentences, it can portray both reading comprehension and writing abilities, thus allowing the generation of a concise and comprehensive digest of multiple input passages.

The Transformer [30] architecture has proved to be highly effective at modelling large dependency windows of textual sequences. Text-to-text approaches [14,26,38], trained over large and comprehensive collections, become effective at *understanding* different topics and retaining language regularities useful for several language tasks. Thus, to generate the agent's response using a transformer model, we give as input the following sequence:

$$\text{``}p_1\ p_2\ \dots\ p_N\text{''}, \tag{4}$$

where each p_k corresponds to one of the top-N candidate passages. With this strategy, we implicitly bias the answer generation by asking the model to summarise the passages that are deemed as more relevant according to the retrieval component.

The implicit bias of the top passages is crucial to steer the Transformer response generation. The sequence of passages of Eq. 4 is given as input to the Transformer, which will then attend to the different passages. As the multi-head attention layers look across the different passages, redundant parts will be merged, while the remaining information will be summarised, leading to a concise but comprehensive answer. The following Transformer models were considered for the task of abstractive summarisation:

- **Text-to-Text Transfer Transformer (T5)** [26] is a text-to-text model based on the encoder-decoder Transformer architecture, pre-trained on the large C4 corpus, which was derived from Common Crawl[1]. A masked language modelling objective is used, where the model is trained to predict corrupted randomly sampled tokens, of varying sizes.
- **BART** [14] is a denoising autoencoder, that combines Bidirectional and Auto-Regressive Transformers. Pre-training consists of corrupting text with an arbitrary noising function and learning an autoencoder to reconstruct the original text. The best performing noise functions were text infilling (using single mask tokens to mask random sampled spans of text), and sentence shuffling (changing the order of sentences in passages).
- **PEGASUS** [38] specialises on the abstractive summarisation task. Multiple important sentences are masked and used as targets, i.e., the model is trained to generated each omitted sentence as output. As in T5, this model is not trained to reconstruct sequences.

4 Evaluation

4.1 Datasets and Protocol

CANARD Dataset [10]. This dataset was used to train and evaluate the query rewriting method. It was created by manually rewriting the queries in QuAC [2] to form non-conversational queries. The training, development, and test sets have 31.538, 3.418, and 5.571, query-rewrites respectively.

TREC CAsT Dataset [5]. This dataset was used to evaluate both the conversational search and answer generation components. There are 50 evaluation topics, each with about 10 turns. Of those in total, 20 conversational topics were labelled on average until turn depth 8 using a graded relevance that ranges from 0 (not relevant) to 4 (highly relevant). The passage collection is composed by MS MARCO [19], TREC CAR [8], and WaPo [20] datasets, which creates a complete pool of close to 47 million passages.

Experimental Protocols. To analyse query rewriting performance, we used the BLEU-4 score [24] between the model's output and the queries rewritten by humans, on the CANARD dataset.

In the passage retrieval experiment, we used the TREC CAsT setup and the official metrics, nDCG@3 (normalised Discounted Cumulative Gain at 3), MAP (Mean Average Precision), and MRR (Mean Reciprocal Rank), along with Recall and P@3 (Precision at 3).

In the answer generation experiment, we used METEOR and the ROUGE variant ROUGE-L. For each query in TREC CAsT, we use as reference passages, all the passages with a relevance judgement of 3 and 4. Hence, the goal is to generate answers that cover, as much as possible, the information contained in all relevant passages, in one concise and summarised answer.

[1] https://commoncrawl.org/.

4.2 Implementation

Query Rewriting. We fine-tuned the T5 [26] model according to [17] and used the CANARD's training set [10], providing as input the concatenation of the conversational queries and passages, and as target the rewritten query. In particular, we used the T5-BASE model and trained for 4000 steps, using a maximum input sequence length of 512 tokens, a maximum output sequence length of 64 tokens, a learning rate of 0.0001, and batches of 256 sequences.

First-Stage Retrieval. To index and search, we used the well tuned Anserini framework [35], in particular, the Python implementation Pyserini[2]. We applied stop word removal, using Lucene's default list, and stemming using Kstem[3]. We experimented with: BM25 [27], language models with Dirichlet (LMD) and Jelinek-Mercer (LMJM) smoothing [37] and from our initial analysis, LMD showed better results. This confirms previous knowledge [37] and matches the shorter queries that we observe in a conversational search scenario. Hence, LMD was the model used in all experiments.

BERT Passage Re-Ranker. To perform re-ranking, we used the BERT model implementation from Huggingface [33]. Following the state-of-the-art [21,22], we used the LARGE version of BERT with a classification layer (feed-forward neural network) on top, that takes as input the query-passage *CLS token* embeddings vector generated by BERT, and classifies the passage as relevant or non-relevant to that query. This model was trained following [21] on the MS MARCO dataset [19]. In testing, we truncate the concatenation of the query, passage, and separator tokens to a maximum of 512 tokens (the maximum number of tokens for the BERT model).

Transformer Based Answer Generation. To generate the summarised answers, we employed the T5-BASE, BART-LARGE and PEGASUS models [33]. The T5-BASE has about 220 million parameters with 12 layers, 768 hidden-state size, 3072 feed-forward hidden-states and 12 heads. BART-LARGE holds about 406 million parameters, with a 12-layer, 1024 hidden state size and 16-head architecture. The PEGASUS model has the biggest number of parameters, 568 million, with 16 layers, 1024 hidden state size and 16-heads.

All models were fine-tuned on the summarising task with the CNN/Daily Mail dataset [13]. To generate the summary, we use 4 beams, restrict the n-grams of size 3 to only occur once, and allow for beam search early stopping when at least 4 sentences are generated. Additionally, we fix the maximum length of the summary to be of the same length of the input given to the models (which corresponds to 3 passages) and vary the minimum length from 20 to 120 words.

[2] https://github.com/castorini/pyserini.

[3] http://lexicalresearch.com/kstem-doc.txt.

4.3 Results and Discussion

Conversation-Aware Query Rewriting. In Table 2, we show the BLEU-4 scores obtained in CANARD's test set and in TREC CAsT's 2019 manually rewritten queries. The rows "Human" and "Raw" are from [10], the row "T5-BASE" is from [17]. The last row corresponds to our implementation. Our results are on par with [17], being lower in the CANARD dataset but higher in TREC CAsT. We believe the minor differences in performance between our T5-Base model and the T5-BASE from [17] are due to the use of different input sequences, as the exact method of constructing the input is not specified in [17].

Table 2. BLEU-4 scores for the CANARD test set and for TREC CAsT using the manually rewritten queries of the evaluation set.

	CANARD	TREC CAsT
Human [10]	59.92	-
Raw [10]	47.44	-
T5-BASE [17]	**58.08**	75.07
Our T5-BASE	56.84	**79.67**

From the analysis of the BLEU-4 scores and outputs, we can conclude that the model is performing both coreference and context resolution, approximating the queries in a conversational format to context-independent queries. Examples of the inputs, targets, and predicted queries, are presented in Table 3. In TREC CAsT, the historical utterances do not depend on the responses of the system, so the answer is not provided as input. As we can see, T5 is capable of resolving ambiguous queries by co-reference resolution, as in example 1, but sometimes mistakes similar co-references when multiple are involved, as evidenced in example 2 and in [17], where the model predicts "throat cancer" instead of "lung cancer". We can also note that this model is more robust than just coreference resolution, as seen in example 3, where it includes the words "Bronze Age Collapse", even though there is no explicit mention (implicit coreference).

Transformer-Based Passage Search. Table 4 shows the results of retrieval on the TREC CAsT dataset. *Original* are the conversational queries (lower-bound), *Manual* is a baseline where the queries were manually rewritten (upper-bound), *T5* is using our query rewriting method, and the other two lines are the results of baselines retrieved from [6]. *clacBase* [3] is a method that uses AllenNLP coreference resolution [11] and a fine-tuned BM25 model with pseudo-relevance feedback, and HistoricalQE [34] is a method that uses a query expansion algorithm based on session and query words together with a BERT LARGE model for re-ranking. The latter was the best performing method in terms of nDCG@3 in TREC CAsT 2019 [6].

Table 3. Example of query rewriting inputs, targets and predictions.

CANARD	
Original query	What was his agreement with McMahon?
T5 Input query	What was his agreement with McMahon? [CTX] Superstar Billy Graham. Return to WWWF (1977–1981) [TURN] Why did he return to the WWWF? An agreement with promoter Vincent J. McMahon Senior
T5 predicted query	What was Superstar Billy Graham's agreement with McMahon?
Target query	What was Billy Graham's agreement with McMahon?
TREC CAsT 2019	
Original query	What are its symptoms?
T5 Input query	What are its symptoms? [CTX] What is throat cancer? [TURN] Is throat cancer treatable? [TURN] Tell me about lung cancer
T5 predicted query	What are throat cancer's symptoms?
Target query	What are lung cancer's symptoms?
Original query	What are some of the possible causes?
T5 Input query	What are some of the possible causes? [CTX] Tell me about the Bronze Age collapse? [TURN] What is the evidence for the Bronze Age collapse?
T5 predicted query	What are some of the possible causes for the Bronze Age collapse?
Target query	What are some of the possible causes of the Bronze Age collapse?

The first observation that emerges from Table 4 is the clear need for a query rewriting method to maintain the conversational context, evidenced by the low scores on all metrics using the original conversational queries. Rewriting queries (with the *T5* model) outperforms the original conversational queries by a $5-20\%$ margin (nDCG@3), thus showing the effectiveness of this approach. The second clear observation is again the considerable improvement when Transformers are used for re-ranking. In this case, the improvement is in the 10–15% range over standard retrieval metrics. This is due to the better understanding that the fine-tuned BERT model has of the interactions between the query and passage terms.

Finally, the largest gains emerge when we combine the two Transformers to deliver state-of-the-art results. With the proposed Transformers we outperform the best TREC CAsT 2019 baseline by 3.9% in terms of nDCG@3. We consider that this improvement is mainly due to the use of a better query-rewriting method that allows the retrieval model to retrieve passages given the conversational context, providing the re-ranker with more relevant passages.

Table 4. Results of retrieval on the TREC CAsT evaluation set. The HistoricalQE [34] was the best performing model in TREC CAsT 2019.

Queries	Re-ranker	Recall	P@3	MAP	MRR	nDCG@3
Original	–	0.454	0.262	0.141	0.336	0.167
Original	BERT	0.454	0.385	0.181	0.456	0.272
T5	–	0.697	0.474	0.251	0.597	0.322
T5	BERT	0.697	0.632	**0.310**	**0.739**	**0.475**
		TREC CAsT baselines				
clacBase [3]	–	–	–	0.246	0.640	0.360
HistoricalQE [34]	BERT	–	–	0.267	0.715	0.436
		Manual baselines				
Manual	-	0.820	0.590	0.327	0.694	0.406
Manual	BERT	0.820	0.757	0.389	0.857	0.577

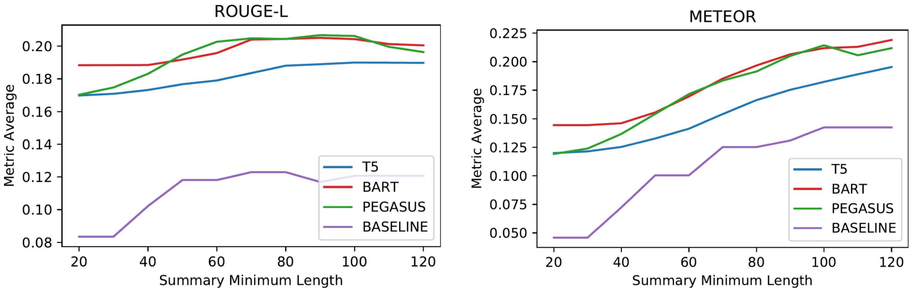


Fig. 2. Performance of the answer generation results under different metrics.

Conversational Answer Generation. Figure 2 shows the result of the answer generation step according to the ROUGE-L and METEOR metrics. The baseline is composed by the concatenation of the top 3 passages, cropped to the maximum length of the passage according to the "Summary Minimum Length" value, respecting sentence endings. In Fig. 2 all answer generation models were better than the retrieval baseline method. According to ROUGE-L the top performance is achieved around 60–90 word length answers. Since the goal is to generate short and informative answers, we were not interested in answers longer than 100 words. Actually, we believe that answers with fewer than 50 words are more natural for conversational scenarios. According to these results we observe that BART was the best answer generation method.

In Fig. 3 we analyse the retrieval and the answer generation performance over conversation turns. We see that peak performance is achieved on the first turn, which was expected given that the first turn that establishes the topic. As the conversation progresses, retrieval performance decreases, but surprisingly, answer generation performance is stable until the 6th turn. We also observed that the decreases in performance are linked to sub-topic shifts within the same conversation topic.

An interesting observation from Fig. 3 is that PEGASUS is the method that exhibits a stronger correlation with retrieval performance. We believe this is related to its generation process that has a behaviour closer to extractive summarisation, while BART and T5 demonstrate a more abstractive behaviour.

Finally, in Table 5 we illustrate the answer generation with all three Transformers. This table further confirms the abstractive versus extractive summarisation behaviours of the different Transformer-based architectures. In this example we see that T5 tries to generate new sentences by combining different sentences.

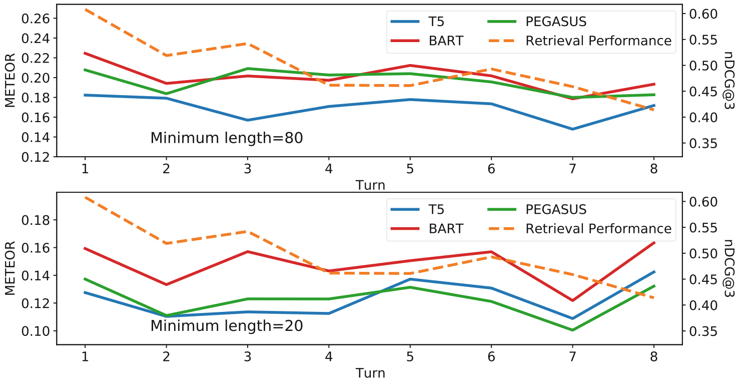


Fig. 3. Answer generation versus retrieval performance per conversation turn. The minimum length is 80 and 20 in the top and bottom graphs respectively.

Table 5. Answer generation example for the turn *"What was the first artificial satellite?"*. Summary minimum length is set to 90. Blue sentences illustrate abstractive, green sentences illustrate extractive, and red sentences illustrate wrong summaries.

Method	Answer
Retrieval passage 1	The first artificial satellite was Sputnik 1, launched by the Soviet Union on October 4, 1957, and initiating the Soviet Sputnik program, with Sergei Korolev as chief designer (there is a crater on the lunar far side which bears his name). This in turn triggered the Space Race between the Soviet Union and the United States
Retrieval passage 2	The first artificial Earth satellite was Sputnik 1. Put into orbit by the Soviet Union on October 4, 1957, it was equipped with an on-board radio-transmitter that worked on two frequencies: 20.005 and 40.002 MHz. Sputnik 1 was launched as a step in the exploration of space and rocket development. While incredibly important it was not placed in orbit for the purpose of sending data from one point on earth to another. And it was the first artificial satellite in the steps leading to today's satellite communications
Retrieval passage 3	The first artificial satellite was Sputnik 1. It was the size of a basketball and was made by the USSR (Union of Soviet Socialist Republics) or Russia. It was launched on October 4, 1957.
T5	the first artificial satellite was Sputnik 1, launched by the u.s. or Russia. it was the size of a basketball and launched on October 4, 1957. the satellite was equipped with a radio-transmitter that worked on two frequencies. incredibly important it was not placed in orbit for sending data from one point on earth to another. in turn, it triggered the space race between the united states and the soviet union.
BART	The first artificial satellite was Sputnik 1, launched by the Soviet Union on October 4, 1957. It was equipped with an on-board radio-transmitter that worked on two frequencies: 20.005 and 40.002 MHz. This in turn triggered the Space Race between the Soviet Union and the United States. The size of a basketball, it was not placed in orbit for the purpose of sending data from one point on earth to another. And it was the first Artificial satellite in the steps leading to today's satellite communications.
PEGASUS	The first artificial satellite was Sputnik 1, launched by the Soviet Union on October 4, 1957. Sputnik 1 was launched as a step in the exploration of space and rocket development. It was not placed in orbit for the purpose of sending data from one point on earth to another. This in turn triggered the Space Race between the USSR and the U.S. There is a crater on the lunar far side which bears his name.

5 Conclusions

In this paper we investigated how Transformer architectures can address different tasks in open-domain conversational search, with particular emphasis on the search-answer generation task. The key findings are:

- **Transformers-based Conversational Search.** Transformers can solve a number of tasks in conversational search, leading to new state-of-the-art results by outperforming the best TREC-CAsT 2019 baseline by 3.9% in terms of nDCG@3. This result is rooted on a fine-tuned bi-directional Transformer model [26] for conversational query re-writing, which attained an improvement of 5–20% (nDCG@3) over raw conversational queries. Similarly, the re-ranking task using a fine-tuned BERT LARGE model [21] improved results by 10–15% (nDCG@3) over an LMD model.
- **Search-Answer Generation.** Experiments showed that search systems can be improved with agents that abstract the information contained in multiple documents to provide a single and informative search answer. In terms of ROUGE-L we concluded that all answer generation models [14,26,38] performed better than the retrieval baseline.
- **Abstractive vs Extractive Answer Generation.** The examined answer generation Transformers revealed different behaviours. BART was the most effective in generating answers that were rewritten with information from different passages. This approach turned out to be better than extractive methods that copy and paste sentences from different passages.

As future research, we plan to improve conversational query rewriting methods, re-rankers with a notion of the context of the conversation, and mine possible conversation paths to steer the answer generation process towards further helping the user in exploring alternative aspects of the searched topic.

Acknowledgement. This work has been partially funded by the iFetch project, Ref. 45920 co-financed by ERDF, COMPETE 2020, NORTE 2020, the CMU Portugal project GoLocal Ref. CMUP-ERI/TIC/0046/2014 and by the project NOVA LINCS Ref. UID/CEC/04516/2013.

References

1. Belkin, N.J.: Anomalous states of knowledge as a basis for information retrieval. Can. J. Inf. Sci. **5**(1), 133–143 (1980)
2. Choi, E., et al.: Quac : Question answering in context. CoRR abs/1808.07036 (2018). http://arxiv.org/abs/1808.07036
3. Clarke, C.L.A.: Waterlooclarke at the TREC 2019 conversational assistant track. In: Voorhees, E.M., Ellis, A. (eds.) Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC 2019, Gaithersburg, Maryland, USA, 13–15 November, 2019. NIST Special Publication, vol. 1250. National Institute of Standards and Technology (NIST) (2019). https://trec.nist.gov/pubs/trec28/papers/WaterlooClarke.C.pdf

4. Croft, W.B., Thompson, R.H.: I3r: a new approach to the design of document retrieval systems. JASIST **38**(6), 389–404 (1987)
5. Dalton, J., Xiong, C., Callan, J.: The trec conversational assistance track (cast) (1 2020). http://www.treccast.ai/
6. Dalton, J., Xiong, C., Callan, J.: TREC cast 2019: The conversational assistance track overview. CoRR abs/2003.13624 (2020). https://arxiv.org/abs/2003.13624
7. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR abs/1810.04805 (2018). http://arxiv.org/abs/1810.04805
8. Dietz, L., Gamari, B., Dalton, J.: Trec car 2.1: A data set for complex answer retrieval (7 2018). http://trec-car.cs.unh.edu
9. Dinan, E., Roller, S., Shuster, K., Fan, A., Auli, M., Weston, J.: Wizard of wikipedia: Knowledge-powered conversational agents. CoRR abs/1811.01241 (2018). http://arxiv.org/abs/1811.01241
10. Elgohary, A., Peskov, D., Boyd-Graber, J.L.: Can you unpack that? learning to rewrite questions-in-context. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November, 2019. pp. 5917–5923. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1605
11. Gardner, M., et al.: Allennlp: a deep semantic natural language processing platform. CoRR abs/1803.07640 (2018). http://arxiv.org/abs/1803.07640
12. Han, S., Wang, X., Bendersky, M., Najork, M.: Learning-to-rank with BERT in tf-ranking. CoRR abs/2004.08476 (2020). https://arxiv.org/abs/2004.08476
13. Hermann, K.M., et al.: Teaching machines to read and comprehend. In: Advances in Neural Information Processing Systems, pp. 1693–1701 (2015)
14. Lewis, M., et al.: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. CoRR abs/1910.13461 (2019). http://arxiv.org/abs/1910.13461
15. Li, J., Monroe, W., Ritter, A., Jurafsky, D., Galley, M., Gao, J.: Deep reinforcement learning for dialogue generation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 1192–1202. Association for Computational Linguistics, Austin, Texas (November 2016). https://doi.org/10.18653/v1/D16-1127, https://www.aclweb.org/anthology/D16-1127
16. Li, J., Monroe, W., Shi, T., Jean, S., Ritter, A., Jurafsky, D.: Adversarial learning for neural dialogue generation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 2157–2169. Association for Computational Linguistics, Copenhagen, Denmark (September 2017). https://doi.org/10.18653/v1/D17-1230, https://www.aclweb.org/anthology/D17-1230
17. Lin, S., Yang, J., Nogueira, R., Tsai, M., Wang, C., Lin, J.: Conversational question reformulation via sequence-to-sequence architectures and pretrained language models. CoRR abs/2004.01909 (2020). https://arxiv.org/abs/2004.01909
18. Liu, Y., et al.: Roberta: A robustly optimized BERT pretraining approach. CoRR abs/1907.11692 (2019), http://arxiv.org/abs/1907.11692
19. Nguyen, T., et al.: MS MARCO: A human generated machine reading comprehension dataset. CoRR abs/1611.09268 (2016). http://arxiv.org/abs/1611.09268
20. NIST: Trec washington post corpus (12 2019). https://trec.nist.gov/data/wapost/
21. Nogueira, R., Cho, K.: Passage re-ranking with BERT. CoRR abs/1901.04085 (2019). http://arxiv.org/abs/1901.04085

22. Nogueira, R., Yang, W., Cho, K., Lin, J.: Multi-stage document ranking with BERT. CoRR abs/1910.14424 (2019). http://arxiv.org/abs/1910.14424
23. Oddy, R.N.: Information retrieval through man-machine dialogue. J. Documentation **33**(1), 1–14 (1977)
24. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA (July 2002). https://doi.org/10.3115/1073083.1073135, https://www.aclweb.org/anthology/P02-1040
25. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.: Open-retrieval conversational question answering. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 539–548. SIGIR 2020, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3397271.3401110
26. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. CoRR abs/1910.10683 (2019). http://arxiv.org/abs/1910.10683
27. Robertson, S., Zaragoza, H.: The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends in Information Retrieval **3**(4), 333–389 (2009). https://doi.org/10.1561/1500000019
28. Song, Y., et al.: An ensemble of retrieval-based and generation-based human-computer conversation systems. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, pp. 4382–4388. International Joint Conferences on Artificial Intelligence Organization (7 2018). https://doi.org/10.24963/ijcai.2018/609
29. Tian, Z., Bi, W., Li, X., Zhang, N.L.: Learning to abstract for memory-augmented conversational response generation. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 3816–3825. Association for Computational Linguistics, Florence, Italy (July 2019). https://doi.org/10.18653/v1/P19-1371, https://www.aclweb.org/anthology/P19-1371
30. Vaswani, A., et al: Attention is all you need. CoRR abs/1706.03762 (2017). http://arxiv.org/abs/1706.03762
31. Voskarides, N., Li, D., Ren, P., Kanoulas, E., de Rijke, M.: Query resolution for conversational search with limited supervision. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (July 2020). https://doi.org/10.1145/3397271.3401130
32. Vtyurina, A., Savenkov, D., Agichtein, E., Clarke, C.L.A.: Exploring conversational search with humans, assistants, and wizards. In: Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems, pp. 2187–2193. CHI EA 2017, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3027063.3053175
33. Wolf, T., et al.: Huggingface's transformers: State-of-the-art natural language processing. CoRR abs/1910.03771 (2019). http://arxiv.org/abs/1910.03771
34. Yang, J., Lin, S., Wang, C., Lin, J., Tsai, M.: Query and answer expansion from conversation history. In: Voorhees, E.M., Ellis, A. (eds.) Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC 2019, Gaithersburg, Maryland, USA, 13–15 November, 2019. NIST Special Publication, vol. 1250. National Institute of Standards and Technology (NIST) (2019). https://trec.nist.gov/pubs/trec28/papers/CFDA_CLIP.C.pdf

35. Yang, P., Fang, H., Lin, J.: Anserini: enabling the use of lucene for information retrieval research. In: Kando, N., Sakai, T., Joho, H., Li, H., de Vries, A.P., White, R.W. (eds.) Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, 7–11 August, 2017, pp. 1253–1256. ACM (2017). https://doi.org/10.1145/3077136.3080721
36. Yang, Z., Dai, Z., Yang, Y., Carbonell, J.G., Salakhutdinov, R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. CoRR abs/1906.08237 (2019). http://arxiv.org/abs/1906.08237
37. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 334–342. SIGIR 2001, Association for Computing Machinery, New York, NY, USA (2001). https://doi.org/10.1145/383952.384019
38. Zhang, J., Zhao, Y., Saleh, M., Liu, P.J.: PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. CoRR abs/1912.08777 (2019). http://arxiv.org/abs/1912.08777
39. Zhuang, Y., Wang, X., Zhang, H., Xie, J., Zhu, X.: An ensemble approach to conversation generation. In: Huang, X., Jiang, J., Zhao, D., Feng, Y., Hong, Yu. (eds.) NLPCC 2017. LNCS (LNAI), vol. 10619, pp. 51–62. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73618-1_5

Complement Lexical Retrieval Model with Semantic Residual Embeddings

Luyu Gao[1(✉)], Zhuyun Dai[1(✉)], Tongfei Chen[2(✉)], Zhen Fan[1(✉)],
Benjamin Van Durme[2(✉)], and Jamie Callan[1(✉)]

[1] Carnegie Mellon University, Pittsburgh, USA
{luyug,zhuyund,zhenfan,callan}@cs.cmu.edu
[2] Johns Hopkins University, Baltimore, USA
{tongfei,vandurme}@cs.jhu.edu

Abstract. This paper presents CLEAR, a retrieval model that seeks to complement classical lexical exact-match models such as BM25 with semantic matching signals from a neural embedding matching model. CLEAR explicitly trains the neural embedding to encode language structures and semantics that lexical retrieval fails to capture with a novel residual-based embedding learning method. Empirical evaluations demonstrate the advantages of CLEAR over state-of-the-art retrieval models, and that it can substantially improve the end-to-end accuracy and efficiency of reranking pipelines.

1 Introduction

State-of-the-art search engines adopt a multi-stage retrieval pipeline system: an efficient first-stage *retriever* uses a query to fetch a set of documents from the entire document collection, and subsequently one or more *rerankers* refine the ranking [28]. The retriever needs to run fast with high efficiency in order to scan through the entire corpus with low latency. As a result, retrievers have remained simple and give only mediocre performance. With recent deep neural models like BERT [10] rerankers pushing reranking accuracy to new levels, first-stage retrievers are gradually becoming the bottleneck in modern search engines.

Typical first-stage retrievers adopt a bag-of-words retrieval model that computes the relevance score based on heuristics defined over the *exact word overlap* between queries and documents. Models such as BM25 [32] remained state-of-the-art for decades and are still widely used today. Though successful, lexical retrieval struggles when matching goes beyond surface forms and fails when query and document mention the same concept using different words (*vocabulary mismatch*), or share only high-level similarities in topics or language styles.

An alternative approach for first-stage retrieval is a neural-based, dense embedding retrieval: query words are mapped into a single vector query representation to search against document vectors. Such methods learn an inner product space where retrieval can be done efficiently leveraging recent advances in maximum inner product search (MIPS) [12,15,34]. Instead of heuristics, embedding

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 146–160, 2021.
https://doi.org/10.1007/978-3-030-72113-8_10

retrieval learns an encoder to understand and encode queries and documents, and the encoded vectors can softly match beyond text surface form. However, single vector representations have limited capacity [1], and are unable to produce granular token-level matching signals that are critical to accurate retrieval [11,33].

We desire a model that can capture both token-level and semantic-level information for matching. We propose a novel first-stage retrieval model, *Complementary Retrieval Model (*CLEAR *)*, that uses dense embedding retrieval to *complement* exact lexical retrieval. CLEAR adopts a single-stage-multi-retriever design consisting of a lexical retrieval model based on BM25 and an embedding retrieval model based on a Siamese framework that uses BERT [10] to generate query/document embedding representations. Importantly, unlike existing techniques that train embeddings directly for ranking independently [4,40], CLEAR explicitly trains the embedding retrieval model with a *residual* method: the embedding model is *trained* to build upon the lexical model's exact matching signals and to fix the mistakes made by the lexical model by supplementing semantic level information, effectively learning semantic matching not captured by the lexical model, which we term the un-captured residual.

Our experiments on large-scale retrieval data sets show the substantial and consistent advantages of CLEAR over state-of-the-art lexical retrieval models, a strong BERT-based embedding-only retrieval model, and a fusion of the two. Furthermore, CLEAR's initial retrieval provides additive gains to downstream rerankers, improving end-to-end accuracy and efficiency. Our qualitative analysis reveals promising improvements as well as new challenges brought by CLEAR.

2 Related Work

Traditionally, first-stage retrieval has relied on bag-of-words models such as BM25 [32] or query likelihood [19], and has augmented text representations with n-grams [25], controlled vocabularies [30], and query expansion [20]. Bag-of-words representations can be improved with machine learning techniques, e.g., by employing machine-learned query expansion on bag-of-sparse-features [5,39], adjusting terms' weights [8] with BERT [10], or adding terms to the document with sequence-to-sequence models [29]. However, these approaches still use the lexical retrieval framework and may fail to match at a higher semantic level.

Neural models excel at semantic matching with the use of dense text representations. Neural models for IR can be classified into two groups [11]: *interaction-based* and *representation-based* models. Interaction-based models model interactions between word pairs in queries and documents. Such approaches are effective for reranking, but are cost-prohibitive for first-stage retrieval as the expensive document-query interactions must be computed online for all ranked documents.

Representation-based models learn a single vector representation for the query or the document and use a simple scoring function (e.g., cosine or dot product) to measure their relevance. Representation-based neural retrieval models can be traced back to efforts such as LSI [9], Siamese networks [2], and MatchPlus [3]. Recent research investigated using modern deep learning techniques to build vector representations: [21] and [13] used BERT-based retrieval to find passages for QA; [4] proposes a set of pre-training tasks for sentence retrieval. Representation-based models enable low-latency, full-collection retrieval with a dense index. By representing queries and documents with dense vectors, retrieval is reduced to a maximum inner product search (MIPS) [34] problem. In recent years, there has been increasing effort on accelerating maximum inner product and nearest neighbor search, which led to high-quality implementations of libraries for nearest neighbor search such as hnsw [24], FAISS [15], and SCaNN [12]. Notably, with these technologies, nearest neighbor search can now scale to millions of candidates with millisecond latency [12,15], and has been successfully used in large-scale retrieval tasks [13,21]. They provide the technical foundation for fast embedding retrieval of our proposed CLEAR model.

The effectiveness of representation-based neural retrieval models for standard ad-hoc search is mixed [11,40]. All of the representation-based neural retrieval models share the same limitation – they use a fixed number of dimensions, which incurs the specificity vs. exhaustiveness trade-off as in all controlled vocabularies [33]. Most prior research on hybrid models has focused on the reranking stage [26]. Some very recent research begins to explore hybrid lexical/embedding models. Its focus is mainly on improving the embedding part with weak-supervision [18] for low-resource setups, or new neural architectures that use multiple embedding vectors to raise model capacity [23]. In these works, embedding models are all trained independently from the lexical models and rely on simple post-training fusion to form a hybrid score. To the best of our knowledge, ours is the first work that investigates jointly training latent embeddings and lexical retrieval for first-stage ad hoc retrieval.

3 Proposed Method

CLEAR consists of a lexical retrieval model and an embedding retrieval model. Between these two models, one's weakness is the other's strength: lexical retrieval performs exact token matching but cannot handle vocabulary mismatch; meanwhile, the embedding retrieval supports semantic matching but loses granular (lexical level) information. To ensure that the two types of models work together and fix each other's weakness, we propose a *residual*-based learning framework that teaches the neural embeddings to be complementary to the lexical retrieval.

3.1 Lexical Retrieval Model

Lexical retrievers are designed to capture token level matching information. They heuristically combine token overlap information, from which they compute a matching score for query document pairs. Decades of research have produced many lexical algorithms such as vector space models, Okapi BM25 [32], and query likelihood [19]. We use BM25 [32] given its popularity in existing systems.

Given a query q and document d, BM25 generates a score based on the overlapping words statistics between the pair.

$$s_{\text{lex}}(q,d) = \text{BM25}(q,d) = \sum_{t \in q \cap d} \text{rsj}_t \cdot \frac{\text{tf}_{t,d}}{\text{tf}_{t,d} + k_1 \left\{ (1-b) + b\frac{|d|}{l} \right\}}. \tag{1}$$

t is a term, $tf_{t,d}$ is t's frequency in document d, rsj_t is t's Robertson-Spärck Jones weight, and l is the average document length. k_1 and b are parameters.

3.2 Embedding Retrieval Model

The embedding retrieval model encodes either the query or document text sequence into a dense embedding vector, and matches queries and documents softly by comparing their vector similarity. Generally, the embedding retrieval model can take various neural architectures that encode natural language sequences such as CNN [16], or LSTM [14], as long as the model outputs can be pooled effectively into a single fixed-length vector for any input. A model capable of deeper text understanding is usually desired to produce high-quality embedding.

This work uses a Transformer [35] encoder. We start with pretrained BERT [10] weights and fine-tune the model to encode both queries and documents into vectors in a d-dimension embedding space, i.e., $\mathbf{v}_q, \mathbf{v}_d \in \mathbb{R}^d$. The model has a Siamese structure, where the query and document BERT models share parameters θ in order to reduce training time, memory footprint, and storthe special token $\langle\text{QRY}\rangle$ to queries and $\langle\text{DOC}\rangle$ to documents. For a given query or document, the embedding model computes the corresponding query vector $\mathbf{v}_q$ or document vector $\mathbf{v}_d$, following SentenceBERT [31], by average pooling representations from the encoder's last layers.

$$\mathbf{v}_q = \text{AvgPool}[\text{BERT}_\theta(\langle\text{QRY}\rangle \ ; \ \text{query})] \tag{2}$$

$$\mathbf{v}_d = \text{AvgPool}[\text{BERT}_\theta(\langle\text{DOC}\rangle \ ; \ \text{document})] \tag{3}$$

The embedding matching score $s_{\text{emb}}(q,d)$ is the dot product of the two vectors. We use dot product as the similarity metric as it allows us to use MIPS [12,15] for efficient first-stage retrieval.

$$s_{\text{emb}}(q,d) = \mathbf{v}_q^{\text{T}} \ \mathbf{v}_d \ . \tag{4}$$

3.3 Residual-Based Learning

We propose a novel residual-based learning framework to ensure that the lexical retrieval model and the embedding retrieval model work well together. While BM25 has just two trainable parameters, the embedding model has more flexibility. To make the best use of the embedding model, we must avoid the embedding model "relearning" signals already captured by the lexical model. Instead, we focus its capacity on semantic level matching missing in the lexical model.

In general, the neural embedding model training uses hinge loss [36] defined over a triplet: a query q, a relevant document d^{+}, and an irrelevant document d^{-} serving as a negative example:

$$\mathcal{L} = [m - s_{\text{emb}}(q, d^{+}) + s_{\text{emb}}(q, d^{-})]_{+} \tag{5}$$

where $[x]_{+} = \max\{0, x\}$, and m is a static loss margin. In order to train embeddings that complement lexical retrieval, we propose two techniques: sampling negative examples d^{-} from lexical retrieval errors, and replacing static margin m with a variable margin that conditions on the lexical retrieval's residuals.

Error-Based Negative Sampling. We sample negative examples (d^{-} in Eq. 5) from those documents mistakenly retrieved by lexical retrieval. Given a positive query-document pair, we uniformly sample irrelevant examples from the top N documents returned by lexical retrieval with probability p. With such negative samples, the embedding model learns to differentiate relevant documents from confusing ones that are lexically similar to the query but semantically irrelevant.

Residual-Based Margin. Intuitively, different query-document pairs require different levels of extra semantic information for matching on top of exact matching signals. Only when lexical matching fails will the semantic matching signal be necessary. Our negative sampling strategy does not tell the neural model the degree of error made by the lexical retrieval that it needs to fix. To address this challenge, we propose a new residual margin. In particular, in the hinge loss, the conventional static constant margin m is replaced by a linear residual margin function m_r, defined over $\mathrm{s}_{\text{lex}}(q, d^{+})$ and $\mathrm{s}_{\text{lex}}(q, d^{-})$, the lexical retrieval scores:

$$m_r(\mathrm{s}_{\text{lex}}(q, d^{+}), \mathrm{s}_{\text{lex}}(q, d^{-})) = \xi - \lambda_{\text{train}}(\mathrm{s}_{\text{lex}}(q, d^{+}) - \mathrm{s}_{\text{lex}}(q, d^{-})), \tag{6}$$

where ξ is a constant non-negative bias term. The difference $\mathrm{s}_{\text{lex}}(q, d^{+}) - \mathrm{s}_{\text{lex}}(q, d^{-})$ corresponds to a residual of the lexical retrieval. We use a scaling factor λ_{train} to adjust the contribution of residual. Consequently, the full loss becomes a function of both lexical and embedding scores computed on the triplet,

$$\mathcal{L} = [m_r(\mathrm{s}_{\text{lex}}(q, d^{+}), \mathrm{s}_{\text{lex}}(q, d^{-})) - s_{\text{emb}}(q, d^{+}) + s_{\text{emb}}(q, d^{-})]_{+} \tag{7}$$

For pairs where the lexical retrieval model already gives an effective document ranking, the residual margin m_r (Eq. 6) becomes small or even becomes negative. In such situations, the neural embedding model makes little gradient update, and it does not need to, as the lexical retrieval model already produces satisfying results. On the other hand, if there is a vocabulary mismatch or topic difference, the lexical model may fail, causing the residual margin to be high and thereby driving the embedding model to accommodate in gradient update. Through the course of training, the neural model learns to encode the semantic patterns that are not captured by text surface forms. When training finishes, the two models will work together, as CLEAR.

3.4 Retrieval with CLEAR

CLEAR retrieves from the lexical and embedding index respectively, taking the union of the resulting candidates, and sorts using a final retrieval score: a weighted average of lexical matching and neural embedding scores:

$$s_{\text{CLEAR}}(q, d) = \lambda_{\text{test}} s_{\text{lex}}(q, d) + s_{\text{emb}}(q, d) \tag{8}$$

We give CLEAR the flexibility to take different λ_{train} and λ_{test} values. Though both are used for interpolating scores from different retrieval models, they have different interpretations. Training λ_{train} serves as a global control over the residual based margin. On the other hand, testing λ_{test} controls the contribution from the two retrieval components.

CLEAR achieves low retrieval latency by having each of the two retrieval models adopt optimized search algorithms and data structures. For the lexical retrieval model, CLEAR index the entire collection with a typical inverted index. For the embedding retrieval model, CLEAR pre-computes all document embeddings and indexes them with fast MIPS indexes such as FAISS [15] or SCANN [12], which can scale to millions of candidates with millisecond latency. As a result, CLEAR can serve as a first-stage, full-collection retriever.

4 Experimental Methodology

Dataset and Metrics. We use the MS MARCO passage ranking dataset [27], a widely-used ad-hoc retrieval benchmark with 8.8 millions passages. The training set contains 0.5 million pairs of queries and relevant passages, where each query on average has one relevant passage[1]. We used two evaluation query sets with different characteristics:

- **MS MARCO Dev Queries** is the MS MARCO dataset's official dev set, which has been widely used in prior research [8,28]. It has 6,980 queries. Most of the queries have only 1 document judged relevant; the labels are binary. MRR@10 is used to evaluate the performance on this query set following [27]. We also report the Recall of the top 1,000 retrieved (R@1k), an important metric for first-stage retrieval.
- **TREC2019 DL Queries** is the official evaluation query set used in the TREC 2019 Deep Learning Track shared task [6]. It contains 43 queries that are manually judged by NIST assessors with 4-level relevance labels, allowing us to understand the models' behavior on queries with *multiple, graded relevance judgments* (on average 94 relevant documents per query). NDCG@10, MAP@1k and R@1k are used to evaluate this query set's accuracy, following the shared task.

Compared Systems. We compare CLEAR retrieval with several first-stage lexical retrieval systems that adopt different techniques such as traditional BM25, deep learning augmented index and/or pseudo relevance feedback.

[1] Dataset is available at https://microsoft.github.io/msmarco/.

- **BM25** [32]: A widely-used off-the-shelf lexical-based retrieval baseline.
- **DeepCT** [8]: A state-of-the-art first-stage neural retrieval model. It uses BERT to estimate term importance based on context; in turn these context-aware term weights are used by BM25 to replace *tf* in Eq. 1.
- **BM25+RM3**: RM3 [20] is a popular query expansion technique. It adds related terms to the query to compensate for the vocabulary gap between queries and documents. BM25+RM3 has been proven to be strong [22].
- **DeepCT+RM3**: [7] shows that using DeepCT term weights with RM3 can further improve upon BM25+RM3.

In addition, we also compare with an embedding only model, **BERT-Siamese**: This is a BERT-based embedding retrieval model without any explicit lexical matching signals, as described in Subsect. 3.2. Note that although BERT embedding retrieval models have been tested on several question-answering tasks [4,13,21], their effectiveness for ad hoc retrieval remains to be studied.

Pipeline Systems. To investigate how the introduction of CLEAR will affect the final ranking in state-of-the-art pipeline systems, we introduce two pipeline setups.

- **BM25+BERT reranker**: this is a state-of-the-art *pipelined* retrieval system. It uses BM25 for first-stage retrieval, and reranks the top candidates using a BERT reranker [28]. Both the BERT-BASE and the BERT-LARGE reranker provided by [28] are explored. Note that BERT rerankers use a very deep self-attentive architecture whose computation cost limits its usage to only the reranking stage.
- **clear+BERT reranker**: a similar pipelined retrieval system that uses CLEAR as the first-stage retreiever, followed by a BERT reranker (BERT-BASE or BERT-LARGE reranker from [28]).

Setup. Lexical retrieval systems, including BM25, BM25+RM3, and deep lexical systems DeepCT and DeepCT+RM3, build upon Anserini [38]. We set k_1 and b in BM25 and DeepCT using values recommended by [8], which has stronger performance than the default values. The hyper-parameters in RM3 are found through a simple parameter sweep using 2-fold cross-validation in terms of MRR@10 and NDCG@10; the hyper-parameters include the number of feedback documents and the number of feedback terms (both searched over $\{5, 10, \cdots, 50\}$), and the feedback coefficient (searched over $\{0.1, 0.2, \cdots, 0.9\}$).

Our neural models were built on top of the HuggingFace [37] implementation of BERT. We initialized our models with BERT-BASE-UNCASED, as our hardware did not allow fine-tuning BERT-LARGE models. For training, we use the 0.5M pairs of queries and relevant documents. At each training step, we randomly sample one negative document from the top 1,000 documents retrieved by BM25. We train our neural models for 8 epochs on one RTX 2080 Ti GPU; training more steps did not improve performance. We set $\xi = 1$ in Eq. 6. We fixed $\lambda_{\text{train}} = 0.1$ in the experiments. For λ_{test}, we searched over $\{0, 1, 0.2, \cdots, 0.9\}$ on 500 training queries, finding 0.5 to be the most robust. Models are trained using the Adam optimizer [17] with learning rate 2×10^{-5}, and batch size 28. In pipelined systems, we use BERT rerankers released by Nogueira et al. [28]. Statistical significance was tested using the permutation test with $p < 0.05$.

5 Results and Discussion

We study CLEAR's retrieval effectiveness on a large-scale, supervised retrieval task, its impact on downstream reranking, and its winning/losing cases.

Table 1. First-stage retrieval effectiveness of CLEAR on the MS MARCO dataset, evaluated using two query evaluation sets, with ablation studies. Superscripts 1–6 indicate statistically significant improvements over methods indexed on the left. ↓ indicates a number being statistically significantly lower than CLEAR. *: CLEAR w/ Constant Margin is equivalent to a post-training fusion of BM25 and BERT-Siamese.

Type	Model	MS MARCO Dev		TREC2019 DL		
		MRR @10	R@1k	NDCG @10	MAP @1k	R@1k
Lexical	1 BM25	0.191^{2}	0.864	0.506	0.377^{5}	0.738^{5}
	2 BM25+RM3	0.166	0.861	0.555^{1}	0.452^{135}	0.789^{13}
	3 DeepCT	0.243^{124}	0.913^{12}	0.551^{1}	0.422^{1}	0.756^{1}
	4 DeepCT+RM3	0.232^{12}	0.914^{12}	0.601^{123}	0.481^{123}	0.794^{13}
Embedding	5 BERT-Siamese	0.308^{1-4}	0.928^{123}	0.594^{123}	0.307	0.584
Lexical+ Embedding	6 CLEAR	$\mathbf{0.338}^{1-5}$	$\mathbf{0.969}^{1-5}$	$\mathbf{0.699}^{1-5}$	$\mathbf{0.511}^{1-5}$	$\mathbf{0.812}^{1-5}$
	– w/ Random Sampling	$0.241^{\downarrow}$	$0.926^{\downarrow}$	$0.553^{\downarrow}$	$0.409^{\downarrow}$	$0.779^{\downarrow}$
	– w/ Constant Margin*	$0.314^{\downarrow}$	$0.955^{\downarrow}$	$0.664^{\downarrow}$	$0.455^{\downarrow}$	0.794

5.1 Retrieval Accuracy of CLEAR

In this experiment, we compare CLEAR's retrieval performance with first stage retrieval models described in Sect. 4 and record their performance in Table 1.

Clear vs. Lexical Retrieval. CLEAR outperforms BM25 and BM25+RM3 systems by large margins in both recall-oriented metrics (R@1k and MAP@1k) as well as precision-oriented ones (MRR@10 and NDCG@10). CLEAR also significantly outperforms DeepCT and DeepCT+RM3, two BERT-augmented lexical retrieval models. DeepCT improves over BM25 by incorporating BERT-based contextualized term weighting, but still use exact term matching. The results show that lexical retrieval is limited by the strict term matching scheme, showing CLEAR's advantages of using embeddings for semantic-level soft matching.

Clear vs. BERT-Siamese Retrieval. BERT-Siamese performs retrieval solely relying on dense vector matching. As shown in Table 1, CLEAR outperforms BERT-Siamese with large margins, indicating that an embedding-only retrieval is not sufficient. Interestingly, though outperforming BM25 by a large margin on MSMARCO Dev queries, BERT-Siamese performs worse than BM25 in terms of MAP@1k and recall on TREC DL queries. The main difference between the two query sets is that TREC DL query has multiple relevant documents with graded relevance levels. It therefore requires a better-structured embedding space to capture this, which proves to be harder to learn here. CLEAR circumvents this full embedding space learning problem by grounding in the lexical retrieval model while using embedding as complement.

Table 2. Comparing CLEAR and the state-of-the-art BM25+BERT Reranker pipeline on the MS MARCO passage ranking dataset with two evaluation sets (Dev: MS MARCO Dev queries; TREC: TREC2019 DL queries). We record the most optimal reranking depth for each initial retriever. Superscripts 1–6 indicate statistically significant improvements over the corresponding methods.

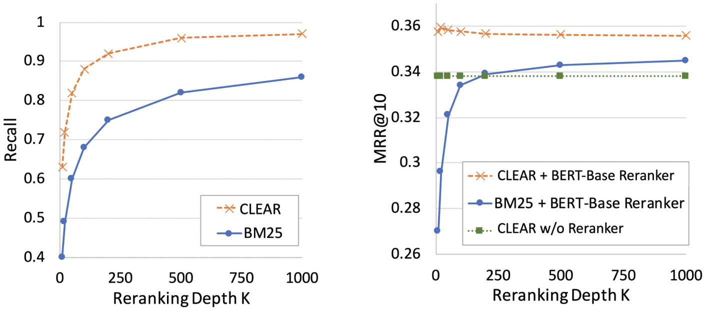
Retriever	Reranker	MSMARCO Dev	TREC DL	Rerank Depth
		MRR@10	NDCG@10	K
1 BM25	–	0.191	0.506	–
2 CLEAR	–	0.338^{1}	0.699^{1}	–
3 BM25	BERT-BASE	0.345^{1}	0.707^{1}	1k
4 CLEAR	BERT-BASE	0.360^{123}	0.719^{12}	20
5 BM25	BERT-LARGE	0.370^{123}	0.737^{123}	1k
6 CLEAR	BERT-LARGE	$\mathbf{0.380}^{1-5}$	$\mathbf{0.752}^{1-5}$	100

Ablation Studies. We hypothesize that CLEAR's residual-based learning approach can optimize the embedding retrieval to *complement* the lexical retrieval, so that the two parts can generate additive gains when combined. To verify this hypothesis, we run ablation studies by **(1)** replacing the error-based negative samples with random negative samples, and **(2)** replacing the residual margin in the loss function with a constant margin, which is equivalent to a fusion of BM25 and BERT-Siamese rankings. Using random negative samples leads to a substantial drop in CLEAR's retrieval accuracy, showing that it is important to train the embeddings on the mistakenly-retrieved documents from lexical retrieval to make the two retrieval models additive. Using constant margins instead of residual margins also lowers the performance of the original CLEAR model. By enforcing a residual margin explicitly, the embedding model is forced to learn to compensate for the lexical retrieval, leading to improved performance. The results confirm that CLEAR is more effective than a post-training fusion approach where the retrieval models are unaware of each other.

5.2 Impacts of CLEAR on Reranking

Similar to other fist-stage retrievers, CLEAR can be incorporated into the state-of-the-art pipelined retrieval system, where its candidate list can be reranked by a deep neural reranker. To quantitatively evaluate the benefit of CLEAR, in the next experiment, we test reranking CLEAR results with BERT rerankers.

Results are listed in Table 2. Here, we compare CLEAR against the widely-used BM25 in a two-stage retrieval pipeline, using current state-of-the-art BERT rerankers [28] as the second stage reranking model. The rerankers use the concatenated query-document text as input to BERT to classify the relevance. We experimented with both BERT-BASE and BERT-LARGE reranker variants provided by [28]. We also investigate the reranking depth for each initial retriever and record the most optimal here.



(a) Retrieval Recall (b) Reranking Accuracy

Fig. 1. Comparison between CLEAR and BM25 pipeline systems on MS MARCO Dev queries. The system uses the BERT-BASE reranker to rerank against various depth K.

The performance of CLEAR *without reranking* is already close to that of the two-stage BM25+BERT-BASE reranker. When adding a reranker, CLEAR pipelines significantly outperforms the BM25 pipelines. We also discover that reranking a truncated top list for CLEAR is sufficient, while top 1000 is required for BM25. Concretely, the required re-ranking depth decreased from K=1,000 to K=20 for BERT-BASE reranker and K=100 for BERT-LARGE reranker, reducing the computational cost by 10×–50×. In other words, CLEAR generates strong initial rankings that systematically raise the position of relevant documents across all queries and help state-of-the-art rerankers to achieve higher accuracy with lower computational costs, improving end-to-end accuracy, efficiency, and scalability.

Figure 1 further plots the recall and reranking accuracy at various reranking depth. Figure 1a shows that CLEAR had higher recall values than BM25 at all depths, meaning that CLEAR can provide more relevant passages to the reranker. Figure 1b shows the performance of a BERT reranker [28] applied to the top K documents retrieved from either BM25 or CLEAR. When applied to BM25, the accuracy of the BERT reranker improved as reranking depth K increases. Interestingly for CLEAR, the reranking accuracy was already high with small K. While increasing K improves global recall, the reranking accuracy shows saturation with larger K, indicating that BERT rerankers do not fully exploit the lower portion of CLEAR candidate lists. We investigate this further in Subsect. 5.3.

5.3 Case Study: The Goods and the New Challenges

In this section, we take a more in-depth look into CLEAR through case studies. We first examine how BM25 ranking changes after being complemented by the dense embedding retrieval in CLEAR, then turn to investigate why the lower part of CLEAR's candidates are challenging for BERT rerankers.

Table 3. Example documents retrieved by CLEAR. We show ranking improvements from pure BM25 to CLEAR's complementary setup .

Query	Document retrieved by CLEAR	BM25 → CLEAR
Weather in danville, ca	Thursday:The Danville forecast for Aug 18 is 85 degrees and **Sunny** . There is 24% chance of **rain** and 10 mph **winds** from the West. Friday:...	989 → 10
brief **government** definition	Legal Definition of brief. 1 1 : a concise statement of a client's case written for the instruction of an **attorney** usually by a **law clerk** ...	996 → 7
population of **jabodatek**	The population of **Jabodetabek**, with an area of 6,392 km2, was over 28.0 million according to the Indonesian Census 2010	Not retrieved → 1

Table 4. Challenging non-relevant documents retrieved only by CRM, not by BM25, through semantic matching. We show in CLEAR initial candidate list ranking as well as after BERT reranking.

Query	Document retrieved by CLEAR	CLEAR → Rerank
Who is robert *gray*	*Grey* started ... dropping his Robert Gotobed alias and using his birthname Robert *Grey*.	Rank 496 → rank 7
What is *theraderm* used for	A *thermogram* is a device which measures heat through use of picture	Rank 970 → rank 8
What is the daily life of *thai people*	Activities of daily living include are the tasks that are required to get going in the morning ... 1 walking. 2 bathing. 3 dressing.	Rank 515 → rank 7

In Table 3, we show three example queries to which the CLEAR brings huge retrieval performance improvement. We see that in all three queries, critical query terms, *weather*, *government* and *jabodatek*, have no exact match in the relevant document, leading to failures in exact match only BM25 system. CLEAR solves this problem, complementing exact matching with high-level semantic matching. As a result, "weather" can match with document content "sunny, rain, wind" and "government" with document content "attorny, law clerk". In the third query, spelling mismatch between query term "jabodatek" and document term "Jabodetabek" is also handled.

While CLEAR improves relevant documents' rankings in the candidate list, it also brings in new forms of non-relevant documents that are not retrieved by lexical retrievers like BM25, and affects downstream rerankers. In Table 4, we show three queries and three corresponding false positive documents retrieved by CLEAR, which are not retrieved by BM25. Unlike in BM25, where false positives mostly share surface text similarity with the query, in the case of CLEAR, the false positives can be documents that are topically related but not relevant. In the first two queries, CLEAR mistakenly performs soft spell matches, while in the third one critical concept "thai people" is ignored.

Such retrieval mistakes further affect the performance of downstream BERT reranker. As BERT also performs semantic level matching without explicit exact token matching to ground, the rerankers can amplify such semantically related only mistakes. As can be seen in Table 4, those false positive documents reside in the middle or at the bottom of the full candidate list of CLEAR. With BERT reranker, however, their rankings go to the top. In general, CLEAR goes beyond exact lexical matching to rely on semantic level matching. While improving initial retrieval, it also inevitably brings in semantically related false positives. Such false positives are inherently more challenging for state-of-the-art neural reranker and require more robust and discriminative rerankers. We believe this also creates new challenges for future research to improve neural rerankers.

6 Conclusion

Classic lexical retrieval models struggle to understand the underlying meanings of queries and documents. Neural embedding based retrieval models can soft match queries and documents, but they lose specific word-level matching information. This paper present CLEAR, a retrieval model that complements lexical retrieval with embedding retrieval. Importantly, instead of a linear interpolation of two models, the embedding retrieval in CLEAR is exactly trained to fix the errors of lexical retrieval.

Experiments show that CLEAR achieves the new state-of-the-art first-stage retrieval effectiveness on two distinct evaluation sets, outperforming classic bag-of-words, recent deep lexical retrieval models, and a BERT-based pure neural retrieval model. The superior performance of CLEAR indicates that it is beneficial to use the lexical retrieval model to capture simple relevant patterns using exact lexical clues, and complement it with the more complex semantic soft matching patterns learned in the embeddings.

Our ablation study demonstrates the effectiveness of CLEAR's residual-based learning. The error-based negative sampling allows the embedding model to be aware of the mistakes of the lexical retrieval, and the residual margin further let the embeddings focus on the harder errors. Consequently, CLEAR outperforms post-training fusion models that directly interpolate independent lexical and embedding retrieval models' results.

A single-stage retrieval with CLEAR achieves an accuracy that is close to popular two-stage pipelines that uses a deep Transformer BERT reranker. We view this as an encouraging step towards building deep and efficient retrieval systems. When combined with BERT rerankers in the retrieval pipeline, CLEAR's strong retrieval performance leads to better end-to-end ranking accuracy and efficiency. However, we observe that state-of-the-art BERT neural rerankers do not fully exploit the retrieval results of CLEAR, pointing out future research directions to build more discriminative and robust neural rerankers.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2015)
2. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a Siamese time delay neural network. Adv. Neural Inf. Process. Syst. **6**, 737–744 (1993)
3. Caid, W.R., Dumais, S.T., Gallant, S.I.: Learned vector-space models for document retrieval. Inf. Process. Manag. **31**(3), 419–429 (1995)
4. Chang, W., Yu, F.X., Chang, Y., Yang, Y., Kumar, S.: Pre-training tasks for embedding-based large-scale retrieval. In: 8th International Conference on Learning Representations (2020)
5. Chen, T., Van Durme, B.: Discriminative information retrieval for question answering sentence selection. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, pp. 719–725 (2017)
6. Craswell, N., Mitra, B., Yilmaz, E., Campos, D.: Overview of the TREC 2019 deep learning track. In: TREC (to appear) (2019)
7. Dai, Z., Callan, J.: Context-aware document term weighting for ad-hoc search. In: WWW 2020: The Web Conference 2020, pp. 1897–1907 (2020)
8. Dai, Z., Callan, J.: Context-aware term weighting for first-stage passage retrieval. In: The 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (to appear) (2020)
9. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. J. Am. Soc. Inf. Sci. **41**(6), 391–407 (1990)
10. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186 (2019)
11. Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep relevance matching model for ad-hoc retrieval. In: Proceedings of the 25th ACM International Conference on Information and Knowledge Management, pp. 55–64 (2016)
12. Guo, R., et al.: Accelerating large-scale inference with anisotropic vector quantization. In: Proceedings of the 37th International Conference on Machine Learning (2020)

13. Guu, K., Lee, K., Tung, Z., Pasupat, P., Chang, M.: REALM: retrieval-augmented language model pre-training. CoRR abs/2002.08909 (2020)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
15. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs. CoRR abs/1702.08734 (2017)
16. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP (2014)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations (2015)
18. Kuzi, S., Zhang, M., Li, C., Bendersky, M., Najork, M.: Leveraging semantic and lexical matching to improve the recall of document retrieval systems: A hybrid approach. ArXiv abs/2010.01195 (2020)
19. Lafferty, J.D., Zhai, C.: Document language models, query models, and risk minimization for information retrieval. In: SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 111–119 (2001)
20. Lavrenko, V., Croft, W.B.: Relevance-based language models. In: SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 120–127 (2001)
21. Lee, K., Chang, M., Toutanova, K.: Latent retrieval for weakly supervised open domain question answering. In: Proceedings of the 57th Conference of the Association for Computational Linguistics, pp. 6086–6096 (2019)
22. Lin, J.: The neural hype and comparisons against weak baselines. In: SIGIR Forum, pp. 40–51 (2018)
23. Luan, Y., Eisenstein, J., Toutanova, K., Collins, M.: Sparse, dense, and attentional representations for text retrieval. Transactions of the Association of Computational Linguistics (2020)
24. Malkov, Y.A., Yashunin, D.A.: Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE Trans. Pattern Anal. Mach. Intell. **42**(4), 824-836 (2018)
25. Metzler, D., Croft, W.B.: A markov random field model for term dependencies. In: SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 472–479 (2005)
26. Mitra, B., Diaz, F., Craswell, N.: Learning to match using local and distributed representations of text for web search. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1291–1299 (2017)
27. Nguyen, T., et al.: MS MARCO: A human generated machine reading comprehension dataset. In: Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches 2016 Co-Located with the 30th Annual Conference on Neural Information Processing Systems (2016)
28. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv:1901.04085 (2019)
29. Nogueira, R., Yang, W., Lin, J., Cho, K.: Document expansion by query prediction. CoRR abs/1904.08375 (2019)
30. Rajashekar, T.B., Croft, W.B.: Combining automatic and manual index representations in probabilistic retrieval. J. Am. Soc. Inf. Sci. **46**(4), 272–283 (1995)
31. Reimers, N., Gurevych, I.: Sentence-Bert: Sentence embeddings using Siamese Bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pp. 3980–3990 (2019)

32. Robertson, S.E., Walker, S.: Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In: Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp. 232–241 (1994)
33. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill Book Company (1984)
34. Shrivastava, A., Li, P.: Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). Adv. Neural Inf. Process. Syst. **27**, 2321–2329 (2014)
35. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, pp. 5998–6008 (2017)
36. Weston, J., Watkins, C.: Support vector machines for multi-class pattern recognition. In: ESANN 1999, 7th European Symposium on Artificial Neural Networks, pp. 219–224 (1999)
37. Wolf, T., et al.: Huggingface's transformers: State-of-the-art natural language processing. CoRR abs/1910.03771 (2019)
38. Yang, P., Fang, H., Lin, J.: Anserini: enabling the use of Lucene for information retrieval research. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1253–1256 (2017)
39. Yao, X., Van Durme, B., Clark, P.: Automatic coupling of answer extraction and information retrieval. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pp. 159–165 (2013)
40. Zamani, H., Dehghani, M., Croft, W.B., Learned-Miller, E.G., Kamps, J.: From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 497–506 (2018)

Classifying Scientific Publications with BERT - Is Self-attention a Feature Selection Method?

Andres Garcia-Silva(✉) and Jose Manuel Gomez-Perez

Research Lab, Expert.ai, Prof. Waskman 10, 28036 Madrid, Spain
{agarcia,jmgomez}@expert.ai
https://www.expert.ai

Abstract. We investigate the self-attention mechanism of BERT in a fine-tuning scenario for the classification of scientific articles over a taxonomy of research disciplines. We observe how self-attention focuses on words that are highly related to the domain of the article. Particularly, a small subset of vocabulary words tends to receive most of the attention. We compare and evaluate the subset of the most attended words with feature selection methods normally used for text classification in order to characterize self-attention as a possible feature selection approach. Using ConceptNet as ground truth, we also find that attended words are more related to the research fields of the articles. However, conventional feature selection methods are still a better option to learn classifiers from scratch. This result suggests that, while self-attention identifies domain-relevant terms, the discriminatory information in BERT is encoded in the contextualized outputs and the classification layer. It also raises the question whether injecting feature selection methods in the self-attention mechanism could further optimize single sequence classification using transformers.

Keywords: Neural language models · Text classification · Scholarly communications

1 Introduction

The annotation and classification of scientific literature is a crucial task to make scientific knowledge easily discoverable, accessible, and reusable, accelerating scientific breakthroughs by helping scholars locate and understand the right research, making connections, and overcoming information overload. Some examples of efforts to structure scientific literature include scientific search engines like Semantic Scholar [1] and Microsoft Academic [23]. Both rely on knowledge graphs to enable a structured representation of scientific knowledge that supports applications like topic-driven search and recommendation. Similarly, scientific publishers have released knowledge graphs such as SN SciGraph [7] in order to more effectively organize their publications and increase automation.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 161–175, 2021.
https://doi.org/10.1007/978-3-030-72113-8_11

Other efforts like ORKG [8] rely on knowledge graphs to structure the actual contributions described in the publications, making research results on a specific topic comparable across the literature.

Publications are therefore being annotated with information about their content, which includes topics [1], fields of study [23], concepts [7], and research fields [8]. Such metadata is generally based on controlled vocabularies and arranged according to a taxonomy [7,8], thesaurus [1,23] or ontology [21]. In some cases, the annotation process can be fully automatic [1,23]. However, authors are often asked to manually classify their contribution in the right categories, which is tedious and error-prone. In other occasions, this task falls under the responsibility of a reduced number of senior expert editors, making the process expensive and slow [21].

In this paper, we focus on the task of classifying scientific publications against a taxonomy of scientific disciplines. A wide variety of approaches are suitable for this task, including machine learning classifiers that rely on high-dimensional sparse representations [10], deep learning classifiers using dense representations [11], and rule-based or heuristic methods [21]. Encouraged by the success of recent developments in natural language processing and understanding, where pre-trained transformer language models dominate the state of the art [27], herein we focus on BERT [5] and its different flavors specialized in the scientific domain: BioBERT [16] and SciBERT [2].

Our experiments confirm that using transformers to train scientific classifiers generally results in greater accuracies compared to linear classifiers that were until now regarded as strong baselines [11]. We also observe that fine-tuning pre-trained transformers on domain-specific corpora contributes to this goal. However, despite previous research focused on interpreting and understanding how transformers encode information [4,9,15,20,25], the actual mechanism by which fine-tuning impacts on our classification task is still unclear. In an effort to shed light on this matter, we focus on analyzing the self-attention mechanism inherent of the transformer architecture [26]. Our findings show that the last layer of BERT attends to words that are semantically relevant for the scientific fields associated with each publication. This observation suggests that self-attention actually performs some type of feature selection for the fine-tuned model.

We investigate the possible relation between self-attention and feature selection methods from different perspectives, including vocabulary overlap, ranking similarity, domain relevance, feature stability, and classification performance. Our results open a future research path to determine whether injecting feature selection methods in the self-attention mechanism could derive even better results for single sequence classification using transformer architectures.

Our main contributions in this paper are the following:

- We leverage the vertical pattern present in the transformer self-attention mechanism of BERT, SciBERT and BioBERT, where some words receive more attention on average than the rest of the words, and compare it against conventional feature selection methods used in text classification.

- We find that self-attention has interesting properties as a feature selection method. The most attended words are in general more relevant to the publication domain than those found using conventional approaches to feature selection. The stability of the features resulting from self-attention is in line with the results obtained through conventional approaches. However, when used to learn classifiers from scratch, methods like chi-square and information gain contribute to train better classifiers.
- We analyze from a semantic point of view the self-attention mechanism and quantify the amount of domain knowledge it encodes in the hidden states of the last layer. To this purpose, we rely on ConceptNet [24], a commonsense knowledge graph where attended words are mapped to concepts from which we derive their corresponding domains.

The remainder of the paper is structured as follows. Section 2 describes related work in the annotation of scientific publications, classification, transformer language models, and other work focused on the analysis of transformer self-attention. In Sect. 3, we present experimental results classifying research papers into a scientific taxonomy. In Sect. 4, we motivate the analysis of self-attention as feature selection with examples of attended words and scientific categories. In Sect. 5, we quantify the relation between self-attention and feature selection methods. Finally, Sect. 6 concludes the paper[1].

2 Related Work

Annotating research articles with entities such as research fields or topics is addressed in the literature using entity recognition and similarity measures between entity labels and their mentions [3]. In Microsoft Academic Graph [23] the candidate entities (field of study) are identified using string matching between the entity keywords and their paper mentions, then rules are applied to gather more candidates and to filter out the less relevant entities. Similarly, the CSO classifier [21], which assigns articles to concepts in the Computer Science Ontology[2], first identifies concepts explicitly mentioned in the text and then, in an effort to find entities not explicitly mentioned, it uses a similarity measure based on word embeddings. In the Semantic Scholar literature graph [1], an ensemble of tools is used to annotate entities: statistical models for entity span prediction and disambiguation, rules for string-based entity spotting, and off-the-shelf tools[3].

In addition, different models can be used for this task, including SVM [10] or softmax classifiers [14]. Mai et al. [17] proposed classifiers based on convolutional [13] and recurrent neural networks [30] to annotate research articles. However,

[1] Tables, datasets and notebooks to reproduce our experiments are available in https://github.com/expertailab/Is-BERT-self-attention-a-feature-selection-method.

[2] See http://cso.kmi.open.ac.uk/.

[3] https://sobigdata.d4science.org/web/tagme/tagme-help.

such deep learning classifiers need to be trained from scratch and depend on the network architecture. On the contrary, neural language models and particularly transformers like GPT-2 [19] or BERT [5] are pre-trained on a large corpus and then fine-tuned for classification by just adding a linear classifier to the model output. This approach has proven to successfully tackle several NLP tasks [27], including text classification. In the scientific domain, SciBERT [2] and BioBERT [16] have also reported state of the art results. Researchers are investigating the mechanics underlying BERT [20], analyzing its hidden states and outputs [9,25], as well as the self-attention mechanism [4,15]. Unlike previous approaches [4,15], we semantically analyze the words that are attended above average in the last hidden state, leveraging the commonsense knowledge represented in ConceptNet, and quantify the relation between attention and feature selection methods often used in text classification.

3 Fine-Tuning Language Models for Text Classification

We evaluate the use of language models on a text classification task where research articles are labeled with one or more knowledge fields. To this purpose, we choose: i) BERT and GPT-2, pre-trained on a general-purpose corpus, ii) SciBERT, pre-trained solely on scientific documents, and iii) BioBERT, pre-trained on a combination of general and scientific text. Table 1, provides relevant information about each language model, its pre-training and vocabulary. BioBERT uses the same tokenization method and vocabulary as BERT, while SciBERT adopts SentencePiece, based on WordPiece tokenization. The overlap between the vocabularies of BERT and SciBERT is 42%, which shows a substantial difference in the most frequently used words in the scientific domain and general-purpose documents. We choose the base version of BERT models

Table 1. Language models pre-training information.

Model	Tokenizer	Vocabulary	Corpus	Domains	steps/epochs
BERT	WordPIece	30K	BookCorpus (2.5B tokens) + Wikipedia (0.8B tokens)	General	1M steps
BioBERT 1.1	WordPiece	BERT	BERT corpus + PubMed abstracts (4.5B tokens)	General + Biomedic	1M steps
BioBERT 1.0	WordPiece	BERT	BERT Corpus + PubMed abstracts (4.5B tokens) + PMC full-text articles (13.5M tokens)	General + Biomedic	470K steps
SciBERT	SentencePiece	30K	Semantic Scholar (3.17B tokens) (1.14M full text papers)	18% Computer Science and 82% Biomedical	Not reported
GPT-2	Byte Pair Encoding (BPE)	50k	8 million web pages, except Wikipedia (40 GB of text)	General	Not reported

(12 layers, 768 hidden size, 12 attention heads per layer) and a comparable model for GPT-2.

To fine-tune BERT, BioBERT and SciBERT on our multilabel classification task, we follow the guidelines provided by Devlin et al. [5] for single-sentence classification. We take the last layer encoding of the classification token <CLS> and add an N-dimensional linear layer, with N the number of classification labels. We use a binary cross-entropy loss function to allow the model to assign independent probabilities to each label. For GPT-2 we also add a linear layer on top of the last hidden state for the classification token. We train the models for 4 epochs, with batch size 8 and 2e–5 learning rate.

As a baseline, we use an SVM with a linear kernel [6]. We follow a one-vs-all strategy to train a binary SVM classifier per category, with grid search for the regularization parameter. We use WordNet to lemmatize the words, whenever they exist in the WordNet lexicon, and remove stop words. In addition, we use fastText [11] to learn a hierarchical softmax classifier using n-gram embeddings. We learn binary classifiers for each category, with automatic hyperparameter optimization to fix learning rate, number of epochs, and n-gram length.

We gather our dataset of scientific articles from a broad range of knowledge fields in SciGraph [7], where articles are labelled following the ANZSRC[4] taxonomy. This taxonomy comprises 22 first level categories, such as *Economics*, *Law*, and *Computer Science*, each of them with their own subcategory tree. From SciGraph, we extract the titles and abstracts of articles published in 2011 and 2012, as well as their categories. In total, we gather 405K papers, 187K from 2011 and the rest from 2012. In average, each first level category has 20,164 articles with a standard deviation of 31,791, which shows how unevenly the different categories are covered. Some of them are well represented, like *Medical And Health Sciences*, with 138,728 articles, while others, like *Studies In Creative Arts And Writing*, have little over a hundred articles.

We fine-tune the language models to learn to classify papers on any of the 22 first level categories. We train on papers only from 2011 and evaluate using 5-fold cross validation. Table 2 shows that the transformers pre-trained on a scientific corpus generally achieve greater f-measure in this task. The exception is BioBERT-1.0, which scores under BERT. BioBERT-1.0 was pre-trained on a lower number of steps than the other transformers, which could be affecting its performance. GPT-2 is the model producing the lowest f-measure, which shows evidence of a potential mismatch between the vocabulary and quality of the scientific corpus and the Web corpus where it was pre-trained, which may be undermining its performance. Overall, transformers produce more accurate classifiers than the linear methods used as baselines.

To further explore the relation between the pre-training and fine-tuning corpora, we learn classifiers to label articles with second level categories in ANZSRC for some of the first level categories. For this experiment, we enlarge our dataset with articles published in 2012 and evaluate only the best language models, discarding BioBERT 1.0 and GPT-2. The results in Table 2 show that, in general,

[4] Australian and New Zealand Standard Research Classification.

Table 2. Evaluation results of the multilabel classifiers (f-measure) on first level categories (a), and on second level categories (b).

First level categories

Model	f-measure
SciBERT	**0.838**
BioBERT-1.1	0.825
BERT	0.819
BioBERT-1.0	0.818
GPT-2	0.808
SVM	0.807
fastText	0.790

Second level categories

Categories	Articles	Subcat.	Bert	BioBERT-1.1	SciBERT	SVM	fastText
Biological	65340	9	0.883	0.884	**0.887**	0.880	0.871
Medical and Health	58068	18	0.838	0.843	**0.854**	0.836	0.819
Chemical	40837	8	0.858	0.862	**0.865**	0.854	0.847
Mathematical	28723	5	0.886	0.883	**0.891**	0.884	0.878
Computer Sciences	20777	6	0.861	0.862	**0.864**	0.861	0.849
Language	2233	6	**0.911**	0.900	0.903	0.900	0.906
Hist. And Archelogy	2076	4	**0.955**	0.950	0.941	0.946	0.946
Built Environment	140	4	0.495	0.700	0.697	**0.808**	0.804
Creative Arts	132	4	0.639	0.788	0.781	**0.925**	0.828

scientific categories are dominated by SciBERT and BioBERT-1.1. However, for categories in humanities, e.g. *Language*, and *History and Archaeology*, BERT produces better classifiers, providing evidence that the general-purpose knowledge encoded in BERT is more relevant in those cases. Interestingly, when there are few examples, e.g., in categories *Built Environment* and *Creative Arts*, the general knowledge encoded in BERT is of little use for the classifiers, while the scientific knowledge in BioBERT-1.1 and SciBERT contributes to achieve higher f-measure. Linear classifiers outperform transformer-based models in such under-represented categories.

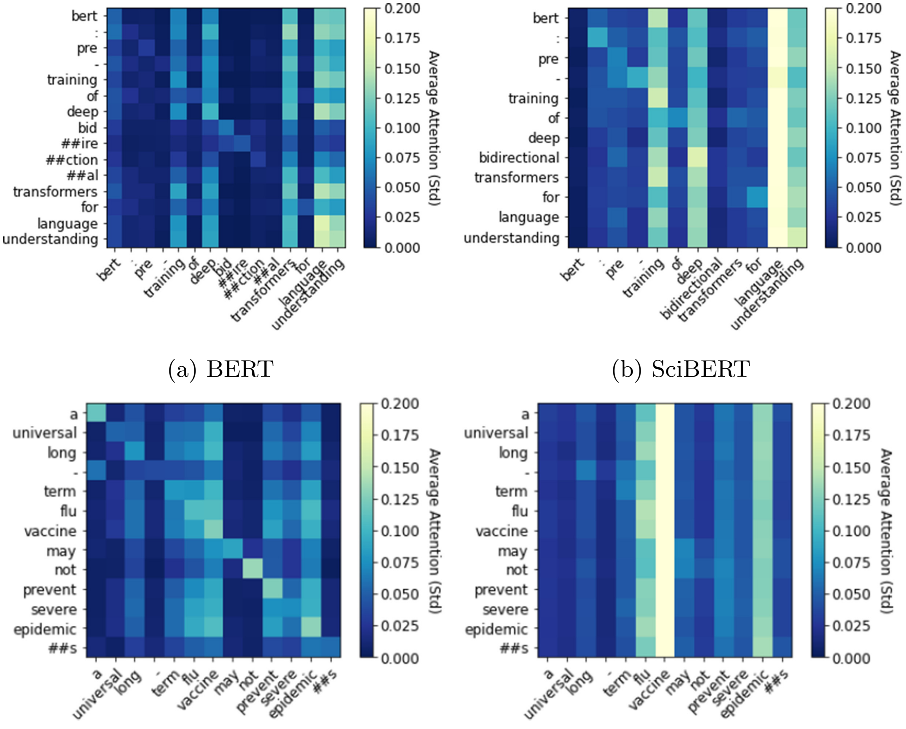
4 Exploring Self-attention Heads

Above we show that BERT-based models are able to produce high performance multilabel classifiers. However, we know little about what makes them good at this task. In this section, we inspect the self-attention mechanism underpinning such models as a key element to understanding this behavior.

According to Clark et al. [4], attention weights indicate how relevant a particular word is when computing the next representation for the current word. To illustrate this statement, Fig. 1, depicts the mean weights of the 12 self-attention heads in the last hidden state of the fine-tuned models for two papers titled *"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"*, and *"A universal long-term flu vaccine may not prevent severe epidemics"*. The plots clearly show the so-called vertical pattern [15], where a few tokens receive most of the attention, such as *training, deep, transformer, language*, and *understanding* in the first sentence, and *flu, vaccine, prevent, severe* and *epidemic* in the second. Note how while the vocabulary captured by SciBERT includes the word *bidirectional*, BERT uses subwords to represent it.

We do not include special tokens <SEP> and <CLS> since the amount of attention received by these tokens makes the attention received by the other tokens barely noticeable. Clark et al. [4] speculate that the attention on <SEP> in one head could indicate that the attention heads function is not applicable, while Rogers et al. [20] interpret the attention on <CLS> as the attention on a pooled sentence-level representation.

From these two examples, we observe that the most attended words in the last hidden state are highly related to the research fields of the articles: *Computer science* and *Medical and Health Sciences*. So, we look into this relation and identify the words that receive most vertical attention in the last hidden state for a subset of our dataset where each first level category is represented with at most 500 papers. First, for each input sequence we calculate the mean weights for the 12 attention heads in the last hidden state. Next, we generate a new weight matrix grouping subwords into words by averaging the subword weights. Finally, we gather the words with a vertical mean attention above the mean attention in the weight matrix. This results in 8,840 attended words for BERT, 17,773 for BioBERT, and 12,265 for SciBERT, corresponding to 16%, 32%, and 22% of the vocabulary managed by each language model.



(a) BERT

(b) SciBERT

(c) BioBERT-1.1

(d) SciBERT

Fig. 1. Average weights in the self-attention heads of the last hidden state.

Table 3 shows the top 20 most frequent attended words in three research fields: *Biology*, *Computer Science* and *History and Archaeology*. As can be noted, most of such words are highly related to the specific research field, appearing along a few punctuation marks and some stop words. While frequent attention to periods and commas was already reported in [4,15], the reason why this happens is not clear yet. Rogers et al. [20] suggest that it must be related to model overparameterization while Clark et al. [4] point at the high frequency of these tokens in the corpus. Stop words are also highly frequent words and the models could be learning to attend to them as in the case of punctuation marks.

Table 3. Most attended words above average attention in the fine-tuned models.

06 - Biological sciences			08 - Computer science			21 - History and archaeology		
BERT	BioBERT	SciBERT	BERT	BioBERT	SciBERT	BERT	BioBERT	SciBERT
,	of	.	,	the	.	,	the	.
Species	The	,	.	of	,	.	of	,
Gene	In	Gene	Data	Data	Data	History	In	History
Cell	.	Species	)	-	Information	)	History	Century
Cells	To	Cell	Image	Time	Algorithm	Historical	-	Historical
Protein	Species	The	network	Information	Network	Archaeological	To	Modern
.	And	Protein	Information	Model	Image	Cultural	Century	The
Genetic	For	Expression	Networks	System	Algorithms	The	.	Archaeological
Plants	-	Genes	Control	Algorithm	As	Social	and	Social
Plant	Gene	Genetic	Images	In	Networks	Archaeology	A	Cultural
Expression	A	Cells	Algorithms	Systems	Systems	Political	Historical	American
Growth	Cell	Growth	Software	Based	Model	Culture	Period	Human
Genes	Protein	Plants	Neural	A	Analysis	Women	Early	Literature
)	Genes	Plant	Optimization	Network	Software	Literary	Modern	Data
Molecular	Cells	Dna	Simulation	To	Time	Heritage	Archaeological	State
Dna	On	Proteins	Learning	Algorithms	Images	Precipitation	World	Women
Stress	With	Molecular	Search	Analysis	Control	Education	Social	Life
Populations	Expression	Populations	Web	Image	Simulation	Identity	On	Period
Population	Genetic	Population	A	Models	Problems	Literature	Years	Political
Genome	Plants	Water	Classification	User	Such	Past	American	Development

5 Feature Selection

In the previous section we show that fine-tuned BERT models concentrate their attention on a subset of the overall vocabulary that ranges between 16% to 32% of the words. Following this observation, we hypothesize that such attention on a selected fragment of the vocabulary is the transformer version of feature selection. However, rather than picking the most interesting features for a classifier, self-attention selects words that heavily influence the representation of the rest of the words in the same sequence. We investigate whether there is a relation between feature selection algorithms commonly used for text classification and the most attended words in the fine-tuned language models.

We center our analysis on four feature selection methods used for text classification [14,18,22]: Chi-square (chi), Information Gain (ig), Document Frequency (df), and Categorical Proportional Difference (pd). Chi-square measures the lack of independence between a word and a class; its value is zero if the word and the class are independent. Information Gain measures the entropy reduction of the dataset when it is split by a feature value. Thus, words with larger information gain discriminate the data ensuring a lower entropy. Document Frequency counts the number of documents where a term appears. Categorical Proportional Difference measures the degree to which a word contributes to differentiating a particular category from others.

We compare the most attended words with those selected by the above-mentioned feature selection methods, and measure how similar the rankings of words sorted by their average attention are to the rankings produced by each feature selection method. In Table 4, we report the vocabulary overlap of the most attended words and feature selection methods after filtering out the stop words. The number of features selected was limited to the top k words, where k is the number of words attended above average by each language model. Indeed, the results indicate a large overlap. Fine-tuned language models for text classification attend up to 64% of the common terms returned by dc, the most simple of our feature selection baselines, which itself performs similarly to ig and chi [29]. For all three models, their most attended words have the largest overlap with document frequency, followed by information gain, chi-square and, finally, proportional difference.

Table 4. Word overlap: most attended vs. feature selection.

LM	FS	%
BERT	dc	60%
	ig	54%
	chi	43%
	pd	12%
BioBERT-1.1	dc	64%
	ig	55%
	chi	44%
	pd	25%
SciBERT	dc	58%
	ig	49%
	chi	42%
	pd	20%

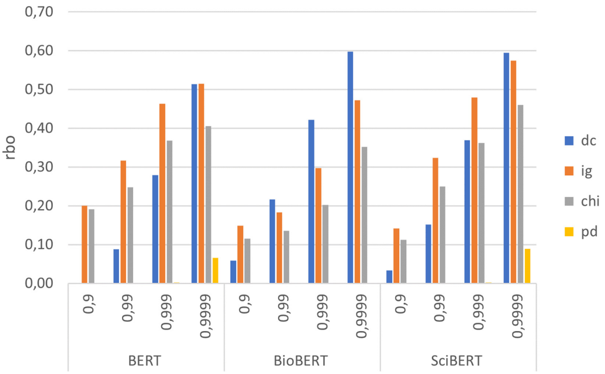


Fig. 2. Rank-biased overlap at different p values between most attended words and feature selection algorithms.

To measure the similarity between rankings we apply the Rank-Biased Overlap (RBO) [28] metric. RBO ranges between 0 to 1, from less to more similar, and was designed for non-conjoint rankings, i.e. both lists may have different items, may be incomplete and with different length. Through the p parameter, RBO models the probability to continue considering the overlap at the next rank, having examined the overlap at the previous rank. Figure 2 shows the RBO for the attention and feature selection rankings. We set p to 0.9, 0.99, 0.999, and 0.9999, indicating the model to assign the first 10, 100, 1,000, and 10,000 ranks respectively, approximately 85% to 86% of the weight of the evaluation.

While the BERT and SciBERT attended words rankings are more similar to the ranking of discriminative words (ig) for p values of 0.9 to 0.999, they finally converge with the ranking of common terms (dc), too. On the other hand, the BioBERT-1.1 ranking is clearly most similar to the common term rankings (dc). We think that the difference between the three models could be related to the subword vocabulary and pre-training corpus. Subword vocabularies are tightly related to the training corpus since they are generated to represent the whole corpus with the minimum number of word pieces. BERT trains its own subword vocabulary on a general corpus and during fine-tuning learns to attend more to discriminative words in the scientific domain. SciBERT also uses its own vocabulary trained on a limited scientific corpus, enabling the model to attend to discriminative words (like BERT) but also to common words due to the domain knowledge it encodes. BioBERT on the other hand reuses the BERT subword vocabulary and therefore many scientific terms are split in a suboptimal number of pieces. This has a negative impact on the ability of the self-attention mechanism to focus on discriminative words, and subsequently on the attention to common terms.

5.1 Domain Knowledge

We investigate the domain relevance of the words that are most attended by the language models and compare it with words produced by the feature selection methods. To this end, we search the words in ConceptNet and leverage the relation *HasContext* to identify the domains where they are commonly used. We manually map the 22 first level categories in ANZSRC to the corresponding concepts in ConceptNet. To deal with morphological variations like plurals and conjugations we use the *FormOf* relation, and to increase the coverage we traverse the *isA* type hierarchy one level up looking for the corresponding concept. For example, the word *networking* is a *FormOf* of the root word *network*, which in turn *HasContext Computer Science* and *Electronics*, and the concept *Electronics isA* type of *Physics*.

For each first level category, we gather the top 100 most attended words, as well as those with the highest scores according to each feature selection method. Then, for each word, we look for the corresponding context according to ConceptNet. Table 5 reports the domain relevance obtained for each category. In BERT and SciBERT, self-attention identifies more domain-relevant words than feature selection methods. However, this is not the case for BioBERT. Recall that in our sample dataset, the set of most attended words produced by BioBERT is the largest (32%) with respect to the vocabulary, which is a clear indication that the model spreads its attention more widely. Weighing the words by their term frequency (TF), attended words remain more domain-relevant than those obtained through feature selection. In fact, the domain relevance of the frequent attended

Table 5. Words per category matching the corresponding ConceptNet context.

	Mean Self-Att.			Feat. Sel.				Self-Att. (TF)			TF				TF/IDF			
Category	BERT	BioB.	SciB.	dc	ig	chi	pd	BERT	BioB.	SciB.	dc	ig	chi	pd	dc	ig	chi	pd
Mathematics	36	18	28	29	18	16	25	60	54	53	51	52	53	33	53	53	55	35
Physics	21	4	22	18	11	13	20	41	42	38	33	33	38	18	41	41	42	18
Chemistry	20	7	18	7	15	16	20	29	30	27	24	24	25	36	27	27	29	37
Biology	18	6	18	11	15	14	11	44	43	38	25	24	28	14	34	33	35	16
Agriculture	1	1	0	0	0	0	1	4	4	4	1	1	1	0	3	3	3	0
Comp. Science	6	4	7	11	5	5	4	20	17	18	14	14	16	11	15	15	16	12
Technology	5	0	3	1	1	1	0	3	2	2	1	1	1	1	1	1	2	0
Medicine	16	13	22	11	12	15	11	30	28	32	19	19	20	17	21	21	22	20
Education	2	1	1	1	1	1	3	4	8	6	4	4	4	1	5	5	5	4
Economics	2	4	1	1	2	2	0	8	10	9	8	8	7	0	9	9	9	0
Commerce	7	4	2	0	1	1	0	6	6	7	3	3	4	2	6	6	6	2
Psychology	2	2	0	4	1	0	2	8	7	5	6	6	7	7	9	9	9	7
Law	5	6	2	6	3	4	7	9	7	8	9	9	8	7	8	8	9	8
Literature	1	0	0	1	1	0	2	0	0	1	1	1	1	1	0	0	0	1
Language	1	0	0	0	2	2	0	2	1	1	2	2	2	0	1	1	1	0
History	10	9	12	23	11	9	11	11	21	21	26	25	26	16	26	25	26	15
Philosophy	16	0	7	15	6	6	10	17	15	18	19	19	18	10	20	20	22	10
Total	169	79	143	139	105	105	127	**296**	**295**	288	246	245	259	174	279	277	**291**	185

words is greater or on pair with those selected when TF/IDF is used to weigh the output of feature selection methods: self-attention takes into account not only the importance of words in the document (TF) but also their importance in the document collection (IDF).

5.2 Feature Evaluation

To evaluate the quality of the resulting features we measure their stability and their classification performance. Stability is the robustness of a feature subset generated from different training sets from the same distribution [12]. To measure stability we compute the mean Jaccard coefficient between the different subsets of words generated by each method. We apply 5-fold cross-validation and process each fold with the fine-tuned language models and the feature selection methods. Stability is reported on Table 6, where we can see that language models attend to the same words with stability values in line with those reported by document count. Attended words are more stable than the rest of the feature selection methods, including chi-square and information gain, which seems to be more volatile across folds.

Table 6. Stability of the features measured using Jackard similarity coefficient

SciBERT	BioBERT	BERT	dc	pd	ig	chi
0.87	0.84	0.83	0.86	0.77	0.65	0.58

In addition, we use the set of features to learn classifiers for the 22 first level categories using Logistic Regression (LR), Naive Bayes (NB), Random Forest (RF), Neural Networks (NN), and SVM. The neural network comprises an embedding matrix of 100 dimensions and a fully connected layer using sigmoid as activation function. For the SVM the regularization parameter is tuned and for the remaining algorithms we use the recommended settings. We evaluate the classifiers using 5-fold cross validation on the subset of documents where each category was represented with up to 500 papers. The f-measure of the classifiers is shown in Fig. 3. In general, we observe that traditional feature selection methods like chi-square and information gain mainly help to learn more accurate classifiers than the set of most attended words by the language models. This observation clearly indicates that the success of BERT models in this task is not only driven by the self-attention mechanism but also by the contextualized outputs of the transformer, which are the input of the added classification layer.

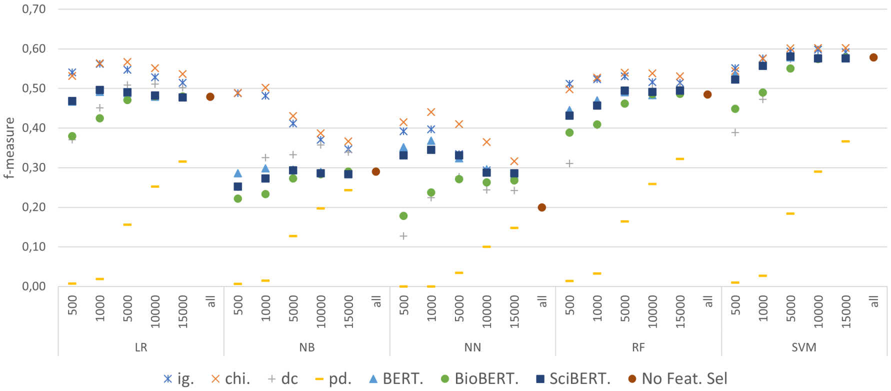


Fig. 3. Classifiers performance using distinct feature sets and number of features.

6 Conclusions

In this paper, we investigate the self-attention mechanism of BERT in a fine-tuning scenario for the classification of scientific articles over a taxonomy of research fields. We observe that attention in the fine-tuned model is focused on words that are highly relevant to the research field of each article. Furthermore, we notice that the most attended words represent just a fraction of the whole vocabulary: a hint that self-attention performs a sort of feature selection.

We systematically compare the most attended words against those resulting from feature selection methods normally used in text classification. We show that language models and feature selection methods like information gain and chi-square share between 42% to 55% of the selected words. We also observe that the attention-based word rankings produced by the transformers are more similar to those obtained using document frequency and information gain.

From our experiments we conclude that self-attention focuses more on words that are relevant to each research domain than the words produced through conventional feature selection. However, self-attention is not as good to learn classifiers from scratch, especially compared to chi-square and information gain. While self-attention identifies domain-relevant terms the discriminatory information in the fine-tuned model is encoded on the output representations and the additional classification layer. As future work, we plan to investigate the impact of integrating, perhaps as part of the loss function, optimal feature selection methods during fine-tuning of transformer for single sequence classification.

Acknowledgment. We gratefully acknowledge the EU Horizon 2020 research and innovation programme under grant agreement No. 825627 (ELG). We also thank Raul Ortega and Cristian Berrio for their contributions to the experimental evaluation.

References

1. Ammar, W., et al.: Construction of the literature graph in semantic scholar. In: NAACL-HLT (2018)
2. Beltagy, I., Lo, K., Cohan, A.: Scibert: pretrained language model for scientific text. In: EMNLP (2019)
3. Chernyak, E.: An approach to the problem of annotation of research publications. In: Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, pp. 429–434. Association for Computing Machinery, New York (2015). https://doi.org/10.1145/2684822.2697032
4. Clark, K., Khandelwal, U., Levy, O., Manning, C.D.: What does BERT look at? an analysis of bert's attention. CoRR abs/1906.04341 (2019). http://arxiv.org/abs/1906.04341
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008)
7. Hammond, T., Pasin, M., Theodoridis, E.: Data integration and disintegration: Managing springer nature sciGraph with SHACL and OWL. In: Nikitina, N., Song, D., Fokoue, A., Haase, P. (eds.) International Semantic Web Conference (Posters, Demos and Industry Tracks). CEUR Workshop Proceedings, vol. 1963. CEUR-WS.org (2017). http://dblp.uni-trier.de/db/conf/semweb/iswc2017p.html#HammondPT17
8. Jaradeh, M.Y., et al.: Open research knowledge graph: next generation infrastructure for semantic scholarly knowledge. In: Proceedings of the 10th International Conference on Knowledge Capture, K-CAP 2019, pp. 243–246. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3360901.3364435
9. Jawahar, G., Sagot, B., Seddah, D.: What does BERT learn about the structure of language? In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3651–3657. Association for Computational Linguistics, Florence, July 2019. https://doi.org/10.18653/v1/P19-1356, https://www.aclweb.org/anthology/P19-1356

10. Joachims, T.: Training linear SVMS in linear time. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2006, pp. 217–226. Association for Computing Machinery, New York (2006). https://doi.org/10.1145/1150402.1150429
11. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
12. Kalousis, A., Prados, J., Hilario, M.: Stability of feature selection algorithms: a study on high-dimensional spaces. Knowl. Inf. Syst. **12**(1), 95–116 (2007)
13. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP (2014)
14. Kou, G., Yang, P., Peng, Y., Xiao, F., Chen, Y., Alsaadi, F.E.: Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. Appl. Soft Comput. **86**, 105836 (2020). https://doi.org/10.1016/j.asoc.2019.105836. http://www.sciencedirect.com/science/article/pii/S1568494619306179
15. Kovaleva, O., Romanov, A., Rogers, A., Rumshisky, A.: Revealing the dark secrets of BERT. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 4365–4374. Association for Computational Linguistics, Hong Kong, November 2019. https://doi.org/10.18653/v1/D19-1445, https://www.aclweb.org/anthology/D19-1445
16. Lee, J., et al.: Biobert: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics **36**(4), 1234–1240 (2020)
17. Mai, F., Galke, L., Scherp, A.: Using deep learning for title-based semantic subject indexing to reach competitive performance to full-text. CoRR abs/1801.06717 (2018). http://arxiv.org/abs/1801.06717
18. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, USA (2008)
19. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8), 9 (2019)
20. Rogers, A., Kovaleva, O., Rumshisky, A.: A primer in bertology: what we know about how bert works. Trans. Assoc. Comput. Linguist. **8**, 842–866 (2020). https://doi.org/10.1162/tacl_a_00349
21. Salatino, A.A., Osborne, F., Birukou, A., Motta, E.: Improving editorial workflow and metadata quality at springer nature. In: Ghidini, C., et al. (eds.) ISWC 2019. LNCS, vol. 11779, pp. 507–525. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30796-7_31
22. Simeon, M., Hilderman, R.J.: Categorical proportional difference: a feature selection method for text categorization. In: AusDM (2008)
23. Sinha, A., et al.: An overview of microsoft academic service (MAS) and applications. In: Proceedings of the 24th International Conference on World Wide Web, WWW 2015 Companion, pp. 243–246. Association for Computing Machinery, New York (2015). https://doi.org/10.1145/2740908.2742839
24. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: an open multilingual graph of general knowledge. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI 2017, pp. 4444–4451. AAAI Press (2017)
25. Tenney, I., Das, D., Pavlick, E.: BERT rediscovers the classical NLP pipeline. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4593–4601. Association for Computational Linguistics, Florence, July 2019. https://doi.org/10.18653/v1/P19-1452, https://www.aclweb.org/anthology/P19-1452

26. Vaswani, A., et al.: Attention is all you need. CoRR abs/1706.03762 (2017). http://arxiv.org/abs/1706.03762
27. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.: GLUE: a multi-task benchmark and analysis platform for natural language understanding. In: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 353–355. Association for Computational Linguistics, Brussels, November 2018. https://doi.org/10.18653/v1/W18-5446, https://www.aclweb.org/anthology/W18-5446
28. Webber, W., Moffat, A., Zobel, J.: A similarity measure for indefinite rankings. ACM Trans. Inf. Syst. (TOIS) **28**(4), 1–38 (2010)
29. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proceedings of the Fourteenth International Conference on Machine Learning, ICML 1997, pp. 412–420. Morgan Kaufmann Publishers Inc., San Francisco (1997)
30. Yin, W., Kann, K., Yu, M., Schütze, H.: Comparative study of CNN and RNN for natural language processing. arXiv preprint arXiv:1702.01923 (2017)

Valuation of Startups: A Machine Learning Perspective

Mariia Garkavenko[1,2(✉)], Hamid Mirisaee[2], Eric Gaussier[1], Agnès Guerraz[2], and Cédric Lagnier[2]

[1] Univ. Grenoble Alpes/CNRS, Grenoble, France
mariia.garkavenko@univ-grenoble-alpes.fr
[2] Skopai, Grenoble, France

Abstract. We address the problem of startup valuation from a machine learning perspective with a focus on European startups. More precisely, we aim to infer the valuation of startups corresponding to the funding rounds for which only the raised amount was announced. To this end, we mine Crunchbase, a well-established source of information on companies. We study the discrepancy between the properties of the funding rounds with and without the startup's valuation announcement and show that the Domain Adaptation framework is suitable for this task. Finally, we propose a method that outperforms, by a large margin, the approaches proposed previously in the literature.

Keywords: Predictive models · Domain adaptation · Startup valuation

1 Introduction

In recent years, startups have radically changed the situation in many different economic ecosystems and have become the pioneers of world-class innovations. The volume of Venture Capital (VC) invested in startups is astonishing - $294.8 billion in 2019, according to Crunchbase[1]. Furthermore, successful startups significantly impact their targeted market and return a considerable profit to their investors. This fact highlights the importance of estimating the worth of startups. However, this issue is a complex, multi-factor problem. Indeed, even though there are dozens of empirical methods proposed by different VC professionals such as Berkus Method[2] and Risk Factor [15], quite often, these methods rely on factors that are hard to measure on their own (such as legal risk, for example).

In this work, we propose a Machine Learning (ML) based approach to predict the valuation assigned to a startup by its investors. However, before going into more details, we provide the following definitions that will be frequently used throughout the paper:

[1] https://news.crunchbase.com/news/the-q4-eoy-2019-global-vc-report-a-strong-end-to-a-good-but-not-fantastic-year/.

[2] https://berkonomics.com/?p=2752.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 176–189, 2021.
https://doi.org/10.1007/978-3-030-72113-8_12

- Startup valuation method: the process of determining how much a startup is valuated economically.
- Equity: percentage of ownership in a company.
- Funding round: a discrete fundraising event for a company, during which the company raises financing at a certain valuation.
- Funding amount: the amount of money invested in a funding round.

Sometimes, funding round announcements include not only the amount of money received by the startup, but also the valuation of the startup. Reading such news, one might be wondering how exactly the entrepreneurs and the VCs come to an agreement about the startup valuation, *i.e.*, how much equity the VC firms get for a certain funding amount. In the literature, only few studies have addressed the startup valuation problem on large-scale datasets and have proposed a data-driven approach to solve it. For instance, [10] performs an empirical study on startup valuation, establishing factors that seem to affect VC and entrepreneur negotiation outcomes.

In this work, we approach startup valuation from a machine learning perspective focusing on European startups. More precisely, our goal is to infer the *the undisclosed valuation of a startup corresponding to a funding round with an announced funding amount*. To do that, we leverage both a large-scale Crunchbase dataset and a novel data source from the Great Britain government registrar, namely Companies House. Our choice to study only European startups is based on the data availability and the possibility of knowledge transfer between countries, which will be discussed in detail in Sects. 3 and 4. We then solve our problem in a Domain Adaptation setting by building a machine learning model which takes into account the discrepancy between the dataset on which the training is performed and the dataset for which we aim to make predictions. Overall, our approach outperforms previously proposed methods by a large margin.

Contribution: Our contribution is thus three-fold: *(i)* we study a novel problem of great practical importance, namely the prediction of startup valuation, *(ii)* we mine heterogeneous data sources including new sources not previously exploited in the literature, and *(iii)* we show that the labeled and unlabeled objects are not aligned and, accordingly, propose to employ a Domain Adaptation setting to train different predictive models.

The rest of the paper is organized as follows: Sect. 2 gives an overview of the existing studies in the field of startup analysis from the ML point of view. The detailed description of data collection and problem formulation are given in Sect. 3. Section 4 details the approach used to solve the problem under consideration. The experimental results are then reported in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Related Work

Over the last few years, various tasks related to startups have been studied with ML methods. [20] is perhaps one of the first attempts to dive into the field of using predictive models for assessing the "success" of companies. In that work, the authors explore the prediction of Merger & Acquisition (M&A) as a proxy for startup success. They consider news pertaining to companies and individuals on TechCrunch. The feature set they used includes company-specific features, such as managerial and financial features, combined with topic-dependent features that have been extracted via Latent Dirichlet Allocation (LDA) from the text of news. In a more recent study, Sharchilev et al. [18] proposed a method, named Web-Based Startup Success Prediction (WBSSP), for startup success prediction. The goal of [18] is to predict whether a startup will secure a funding round within a year or not. In a relatively similar work, Hunter et al. [6] proposed to construct a portfolio of startups in which at least one startup achieves an exit, *i.e.* either gets Initial Public Offering (IPO) or is acquired by another company. Starting from a Brownian motion model, the authors of [6] propose to use a greedy approach to solve the "picking winners" problem.

In [2], the authors study the startup success prediction task in the context of a worldwide startup network. The authors build a startup graph, in which an edge between two startups signifies that a person worked in both startups, and show that the centrality of a startup in such a graph correlate with its success. In that study, success is defined as an exit or the taking over of another firm within seven years. Another study in this context is [5], where the authors show that different stages of fundraising events lead to different success factors. The authors then define venture success as raising another round, getting acquired, or going public in the following two years.

Startup Valuation Problem: As to the empirical analysis of startup valuation, [10] performed a study of the different factors that are generally considered to be important in the valuation process. The authors use regression analysis to identify the most critical factors. In contrast, in our work, we aim to use an ML pipeline to predict the hidden valuation with the best possible accuracy. More recently, [14] analyzed the relation between the funding amount and the startup valuation at different stages. They propose a linear (in log-log space) model for this task. Their proposed method takes into account only the funding amount and the funding round series. Our work, however, is quite different in several aspects: first, we extract a rich feature set for each startup, which includes the information about the previously secured funding rounds, the founders, the team of a startup, etc. The second difference is that we study the distributions of the funding rounds' characteristics with announced *and* unannounced valuations.

3 Data Analysis and Problem Formulation

While stocks of public companies are traded daily, and the value of a company can be calculated at any moment, the shares of a startup are rarely sold, and the valuation of a startup is documented only when particular events occur. These events include funding rounds, Merger and Acquisition deals (M&A), and Initial Public Offerings (IPO). In this study, we focus on the valuations obtained

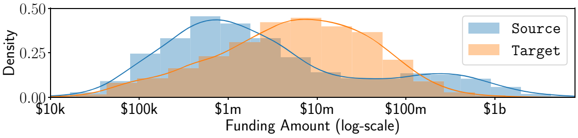


Fig. 1. Comparison of funding amounts between `Source` and `Target`.

during the funding rounds since they are much more frequent than IPO and M&A. Besides, the information about the raised funding amount gives a vital clue about the startup valuation. In the rest of this section, we describe our main repositories for data collection and then illustrate the compatibility between the information taken from different sources.

3.1 `Source` and `Target` Data: Crunchbase

Crunchbase is a well-established data source in startup modeling literature where a wide range of information on startups can be found. The vast majority of studies investigating the field of startups via ML methods leveraged this database [2,5,6,14,18,20]. In our study, the data from Crunchbase plays a critical role as well. We adopt the following strategy to collect data from Crunchbase: First, we extract information about the funding rounds present in the Crunchbase snapshot on July 1, 2020, and then collect the corresponding startups' information. Since we are mostly interested in the traditional venture capital deals for the startups that have not yet gone public, we only collect the following funding rounds: Angel, Pre-Seed, Seed, Series {A, B, C, D, E, G, F, H, I}, Venture, Corporate Round, Private Equity, Undisclosed and Convertible note. Additional information on the startup funding types can be found in [19]. Such procedure leaves us with:

- 11994 funding rounds with known corresponding startup valuation, which will be referred to as `Source` and
- 185943 funding rounds for which the corresponding startup valuation is not disclosed, which will be referred to as `Target` and for which we aim at predicting the valuation.

Distribution Shift. Initial comparison of the funding amount distributions of the `Source` and the `Target` can be seen in Fig. 1. In the case of announced valuations, i.e. `Source`, the distribution is bimodal with the first mode corresponding roughly to $600K raised and the second mode at $250M. Simultaneously, the funding round sizes with undisclosed valuation, i.e. `Target`, have a single mode at $10M. Our goal is to predict the startup valuation on the `Target`, which is for now entirely unlabeled, *i.e.* the valuations are unknown for this set. Given the shift shown in Fig. 1, one needs at least a small portion of `Target` to be annotated. This annotated data then can be used for evaluating the trained models, or even partially for the training purposes, as we will see in Sect. 4. Nevertheless, annotating this kind of data is very difficult. As explained in Sect. 1, determining the valuation of startups requires a wide range of domain expertise. What is even more important is that different investors use different processes to perform the valuation, leading to different valuation numbers for the same startup.

To alleviate this issue, we exploit another data source, namely Companies House[3], the United Kingdom (UK) registrar of companies which, to the best of our knowledge, has not previously been exploited in the startup research literature. In the following section, we briefly describe how the data is collected from Companies House and then illustrate that this data can indeed be used as an additional source of data for the current study.

3.2 $\texttt{Target}_{LAB}$ Data: Companies House

In the UK, companies must file specific documents to Companies House when they participate in an equity fundraising process. What is of great interest to our task is that every time a startup seeks equity funds, it *issues shares*. Furthermore, whenever a company issues shares, it is obliged to file a form, called SH01, which contains the following information: the number of shares allotted, the amount paid on each share, and the total number of shares of the company. Given this information, one can easily calculate the funding amount and the startup's valuation. The funding amount is the number of shares allotted multiplied by the amount paid on each share. The startup's valuation is then calculated as the total number of shares times the amount paid on each share. Finally, the investor's equity is equal to the number of shares allotted divided by the total number of shares.

Thus, for startups in the `Target`which are present in Companies House, one can readily obtain annotations. In the remainder, the annotated part of the `Target` set will be denoted as $\texttt{Target}_{LAB}$ (LAB stands for labeled). To do the cross-referencing between Companies House and Crunchbase, we align company name *and* either legal name, company's address or a name of a person working in a startup. The code for the Companies House data collection is available. [4].

To make sure that $\texttt{Target}_{LAB}$ can be used safely in our study (be it in the training or testing part of the model), one needs to check if $\texttt{Target}_{LAB}$ has the same characteristics as `Target` and, as a result, can be used as a proper evaluation (or further training) data. This point is investigated below.

Geographical Transfer. Although our preliminary studies show that the UK funding rounds and valuations differ from those of China or the USA (as illustrated by the D statistics of a Kolmogorov-Smirnov test, used to assess whether two distributions are similar or not, which amounts to 0.27 for China and to 0.73 for the USA), a reasonable suggestion might be that the investment context in the UK and other countries of the region, namely Europe[5], might be similar.

[3] https://beta.companieshouse.gov.uk/.

[4] https://github.com/garkavem/Company-House-SH01-Parsing.

[5] The list of European countries, referred to as Europe and abbreviated as EU hereafter, is detailed in [1] for space reasons.

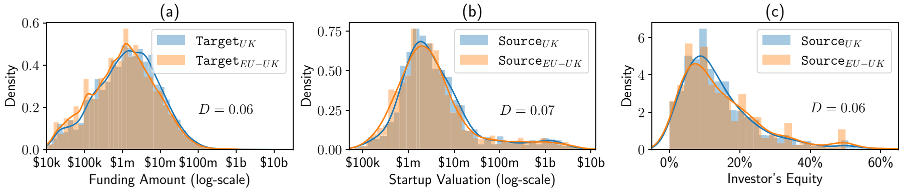


Fig. 2. EU vs. the UK data: (a) funding amounts, (b) valuations, and (c) investors' equities in the funding rounds with announced valuation. The D statistics of the Kolmogorov-Smirnov test is provided in each case.

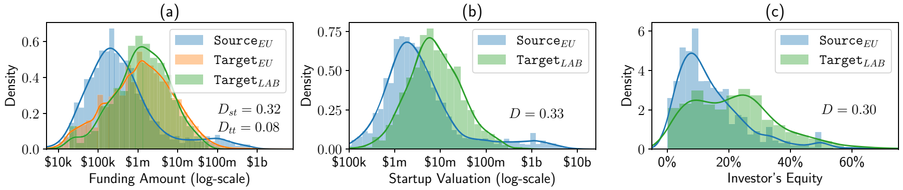


Fig. 3. Comparison of European funding rounds with announced valuation ($\texttt{Source}_{EU}$), unannounced valuation ($\texttt{Target}_{EU}$) and the set of funding rounds for which the valuation was extracted from Companies House ($\texttt{Target}_{LAB}$). (a) funding amounts (b) startups' valuations (c) obtained investors' equities. The D statistics of the Kolmogorov-Smirnov test is provided in each case.

To illustrate this point, we compared three different axes: We first investigated the funding amount distribution difference between the UK startups of `Target`, denoted as $\texttt{Target}_{UK}$, and all other European startups of `Target`, denoted as $\texttt{Target}_{EU-UK}$. This comparison can be seen in Fig. 2(a). We also compared, in Fig. 2(b), the valuation of UK startups from the `Source`, denoted as $\texttt{Source}_{UK}$, with those of all other European countries, denoted as $\texttt{Source}_{EU-UK}$. Finally, Fig. 2(c) illustrates the investor's equity for the same data. As one can note, on the three (sub-)figures, the distributions are very similar. This is confirmed by the D statistics of the Kolmogorov-Smirnov test which amounts to at most 0.07. In contrast, it amounts to 0.2 when comparing `Source` with $\texttt{Source}_{UK}$. These findings lead us to consider that one can treat UK based startups and European startups as similar in terms of funding and valuation. In other words, $\texttt{Target}_{LAB}$ shares the same characteristics as $\texttt{Target}_{EU}$ and, accordingly, can be used in the European startup valuation prediction task. The fact that these two sets are similar in terms of funding amount is crucial to design a valuation model, as we will see in Sect. 5.4.

We compare in Fig. 3(a) the properties of funding amounts of $\texttt{Source}_{EU}$, $\texttt{Target}_{EU}$ and $\texttt{Target}_{LAB}$. This plot shows that $\texttt{Target}_{EU}$ and $\texttt{Target}_{LAB}$ are quite similar to each other ($D = 0.08$), and both are different from $\texttt{Source}_{EU}$ ($D = 0.32$). Such similarity supports our hypothesis that $\texttt{Target}_{LAB}$ is much closer to $\texttt{Target}_{EU}$ than $\texttt{Source}_{EU}$ and, thus, a machine learning model's

Table 1. Summary of the data. CB: Crunchbase, CH: Companies House.

Zone	Valuation announced in CB	Valuation undisclosed in CB	Valuation undisclosed in CB Computed from CH
World	11994 ($\texttt{Source}$)	185943 ($\texttt{Target}$)	
Europe	3177 ($\texttt{Source}_{EU}$)	34622 ($\texttt{Target}_{EU}$)	
UK	1438	12047	969 ($\texttt{Target}_{LAB}$)

performance on $\texttt{Target}_{EU}$ is better approximated by the model's performance on $\texttt{Target}_{LAB}$ than on hold-out $\texttt{Source}_{EU}$.

Additionally, in Fig. 3(b) and (c) we illustrate the comparison of $\texttt{Source}_{EU}$ and $\texttt{Target}_{LAB}$ in terms of valuation and investor's equity. The properties of $\texttt{Target}_{LAB}$ in terms of startup valuation and investor's equity allow us to get some insight into the differences between the funding rounds with announced and unannounced valuations. An interesting observation is that the differences in funding amounts and investor's equity distributions partially compensate each other, and thus the difference in startup valuation distribution is slightly less prominent. This is not really surprising as startups want to be seen as successful and valuable. Thus, when they raise a relatively small amount of money for an unusually small investor's equity, they are more motivated to report its valuation in addition to the funding amount.

3.3 Summary of Dataset

Table 1 summarizes the different sets used in our analysis. It is worth noticing that, according to what has been shown previously, there is no particular reason to restrict the training set to $\texttt{Source}_{EU} \in \texttt{Source}$.

4 Approach

As explained in Sect. 3.1 and illustrated in Fig. 1, there is a significant shift between the $\texttt{Source}$ and the $\texttt{Target}$ distributions. The described problem typically corresponds to a Domain Adaptation (DA) setting. The core of the DA field is to deal with such scenarios where the source and target data come from different distributions. In the literature, there are mainly three types of DA approaches: unsupervised, semi-supervised, and supervised. Unsupervised Domain Adaptation (UDA) refers to a setting in which the model is trained on the labeled data from source domain and unlabeled data from target domain. For a comprehensive overview of UDA methods, we refer the reader to [9]. The setting in which a portion of the target data is annotated and the learning is performed using labeled source data and both labeled and unlabeled target data [17] is known as Semi-Supervised Domain Adaptation (SSDA). Finally, the Supervised Domain Adaptation (SDA) corresponds to the scenario in which both source and target data are labeled and they are both used in the training phase

[11]. Note that, once only a portion of the target domain data is labeled, one can either employ SDA, by ignoring the unlabeled part of the target, or use SSDA by taking into account both the unlabeled and labeled parts of the target. It is also possible to solve the problem in UDA setting using only the unlabeled target data. In order to adapt our data to all these variants, we divide $\texttt{Target}_{LAB}$ into three sets (25%–25%–50% partitions respectively):

- $\texttt{Target}_{LAB(train)}$, with 242 examples, which will be used for the training in SSDA and SDA,
- $\texttt{Target}_{LAB(dev)}$, with 242 examples, which will be used for hyperparameters tuning in SSDA and SDA,
- $\texttt{Target}_{LAB(test)}$, with 485 examples, which will be used to evaluate the models and to report the results on all methods.

4.1 Unsupervised Domain Adaptation

In the UDA setting, we use the Source and Target sets described in Sect. 3 in order to train our model. The technique we use to do that is the one presented in [4] as it has shown outstanding results on different datasets. This model, named Domain-Adversarial training of Neural Networks (DANN), learns a representation that is informative for the main learning task on the source domain and is invariant with respect to the shift between the domains. To this end, the domain classifier is trained to discriminate between the domains. However, a Gradient Reversal Layer incorporated into it passes the signal without a change on the forward pass but reverses the gradients on the backward pass. Thus, the feature extractor parameters are updated in the direction opposite to the one desirable for the domain discrimination task.

4.2 Supervised Domain Adaptation

Supervised Domain Adaptation is a setting in which the labeled examples from the source domain are used along with only the labeled examples from the target domain. Usually, the number of labeled examples from source is much larger than the number of labeled examples from target. That is true in our case as well since $|\texttt{Source}| \gg |\texttt{Target}_{LAB(train)}|$ (11994 vs. 242 examples).

The most straightforward approach for this task is to train a supervised machine learning model on the concatenation of source data and the labeled part of target domain data. The advantage of such an approach is that it can be applied to any base learning model. It has also been shown that even in the presence of abundant unlabeled target domain data and a tiny amount of labeled target data, UDA methods sometimes cannot outperform this simple approach [17]. For this reason, we rely on several supervised machine learning models which will be described in Sect. 5.1.

4.3 Semi-supervised Domain Adaptation

Semi-supervised Domain Adaptation remains a topic slightly less covered in the literature than UDA. Among the recent methods, one could highlight the minmax entropy method proposed by [17] or the domain adaptive adversarial perturbation scheme from [8]. Despite these methods' impressive performance on various benchmarks, adapting them to the regression problem is not straightforward as they rely on class prototypes. Overall, our literature study did not lead to any SSDA method easily adaptable for our task, and we have directly adapted the DANN algorithm for this setting. This adaptation considers, at every iteration, two mini-batches, one consisting of unlabeled target examples and the other of labeled examples, half of which randomly selected from `Source` and the other half from $\texttt{Target}_{LAB(train)}$. Such an adaptation is quite standard, as described in [17], and allows one to bias the model learned towards the target domain.

4.4 Features

Our choice for the features used for the task of startup valuation prediction was based on previous studies [10,18,21] as well as on the available data. Table 2 provides an overview of the features we finally retained, categorised into four main groups: General, Funding Round, Financial History and Social Networks.

The *General* group presents generic features such as age of startup, country of origin, number of founders and employees. The *Funding Round* group merely includes the series and the amount raised during the funding round for which we aim at predicting the valuation. The *Financial History* group includes statistics about the previous funding rounds. The *Social network* features, extracted from Twitter, represent the "importance" of startups on social media. Since many entrepreneurs dedicate a considerable amount of time on online networks in order to reach potential customers, partners or investors, we hypothesise that some characteristics of the startup's activity on social media might be correlated to its maturity and possibly valuation. Although it would be interesting to use other information from other social networks, in this study, we narrow down our monitoring to startups' activities on Twitter since its API is readily available to researchers, contrary to other platforms such as Linkedin or Facebook.

5 Experiments

In this section, we present our experimental results performed on the approaches explained in the previous section as well as some other baselines. We then provide some insight into the contributions of the different features.

5.1 Baselines

The following is the list of baselines that we use in order to illustrate the adaptability of the DA setting to the problem under consideration. Note that to train

Table 2. Startup features used in this study.

Group name	Features	Source
General	Country, age of the startup, number of founders, number of current team members, number of past team members, number of founders with previous experience as founder or top-manager at other companies, number of news talking about the startup	Crunchbase
Funding round	The amount raised in the funding round corresponding to the target valuation, series of the funding round corresponding to the target valuation	Crunchbase
Financial history	Number of previously secured funding rounds, previous funding amount, time since the previous funding round, mean of funding amount raised during the previous funding rounds, max of funding amount raised during the previous funding rounds, funding amount at each series: Seed, Series A, etc.	Crunchbase
Social networks	Number of tweets, mean/max number of likes of tweet, mean/max number of retweets of tweet, number of different users to which startup replied, number of different hashtags used by the startup	Twitter API

these baselines, we only use the `Source` data, i.e., we consider the problem as a classical regression problem.

- **EPoSV** (An Empirical Perspective on Startup Valuations [14]): to the best of our knowledge, it is the only data-driven approach for startup valuation prediction. It consists in finding the best coefficient binding logarithm of the funding amount and the logarithm of startup valuation for each fundraising series.
- **CatBoost:** CatBoost [13] is a popular gradient boosting library. We choose gradient boosting for two reasons: *(i)* it achieves state-of-the-art results on many practical tasks [3,16,22], and *(ii)* this particular implementation has been shown to work well in the startup fundraising prediction task [18]. Although in [18] the authors use CatBoost as the principal component of a task-specific framework that combines several different models, applying a stand-alone CatBoost model to our data also seems appropriate.
- **MLP:** we also use a classical multilayer perceptron with three fully connected hidden layers of 1000, 500 and 250 neurons, ReLU [12] nonlinearities followed by a batch normalization layer [7].

5.2 Experimental Setup and Metrics

We apply $\log_{10}$ *transformation to target values* so as to have them in a reasonable range. For evaluation, we make use of the coefficient of determination R^2 and the root mean squared error (RMSE).

For baseline methods trained on `Source` as well as for DANN in the unsupervised setting, we do not use $\texttt{Target}_{LAB(train)}$ or $\texttt{Target}_{LAB(dev)}$ for training and parameter tuning, since our goal is to find out what is the best performance that one could achieve using only the `Source` (case of baselines) and unlabeled `Target` (case of unsupervised DANN) readily available in Crunchbase.

Table 3. Experimental results. Bold numbers are used for models statistically significantly better than the other models (Wilcoxon signed-rank test with $p < 0.001$). Baselines are separated with a vertical line.

	EPoSV	CatBoost	MLP	DANN	CatBoost	MLP	DANN
	(Source)	(Source)	(Source)	(UDA)	(SDA)	(SDA)	(SSDA)
R^2 ⇑	0.617	0.738	0.769	0.788	**0.817**	0.807	0.807
RMSE⇓	0.347	0.293	0.275	0.263	**0.245**	0.251	0.250

The essential CatBoost parameters, such as learning rate and the number of estimators, were chosen on cross-validation (CV) on Source. In the SDA setting, we use the same learning rate; the number of estimators is chosen based on the Target$_{LAB(dev)}$ metrics. The weights of the Target$_{LAB(train)}$ samples are set to 10 to partially compensate for the differences in Source and Target$_{LAB(train)}$ sizes.

The MLP architecture, as well as the training parameters, including the optimizer, learning rate scheduler, batch size, and the number of epochs, were chosen using CV on Source. The same parameters were used for DANN method. To reduce the hyperparameters influence, all these parameters (except for the number of epochs) are used in SDA and SSDA settings as well. The number of epochs in SDA and SSDA settings is defined by performance on Target$_{LAB(dev)}$.

To robustly estimate the performance of different methods, we repeat this procedure for 20 random splits of the Target$_{LAB}$ set into test and training/dev parts. For MLP and DANN, we repeat the experiment with five different random seeds used for the initialization of weights for each split.

5.3 Results

The results of our experiments are shown in Table 3. In each column, we specify if the method uses only Source data (the first three columns), or if it is supervised, semi-supervised, or unsupervised domain adaptation (SDA, SSDA, and UDA, respectively). As one can observe, among all approaches, EPoSV performs significantly worse. This observation mainly suggests that using a rich set of features and a more powerful model is required for solving the startup valuation task, which is not the case for EPoSV.

The second observation is that all DA based approaches outperform the baselines, which are trained only on Source. This point illustrates that DA setting is indeed a more appropriate approach for solving such a problem. Among all baselines, one can notice that MLP performs the best. The next observation is that in the absence of target domain information, MLP can generalize better to the target domain data. However, once target domain information is introduced, CatBoost achieves better results than MLP. Such improvement is due to the ability of boosting based methods in dealing with complex input data. The SDA version of CatBoost also achieves the best results even among all other DA based approaches.

Table 4. Contribution of different feature groups. Bold numbers are used for models statistically significantly better than the other models (Wilcoxon signed-rank test with $p < 0.05$).

	Full data	$\ominus$General	$\ominus$Funding	$\ominus$Financial	$\ominus$Social net.
$R^2 \Uparrow$	**0.817**	0.789	0.499	0.793	0.811
RMSE$\Downarrow$	**0.245**	0.263	0.405	0.261	0.250

Another finding is that even in the absence of labeled data in the target domain, i.e. $\texttt{Target}_{LAB}$, a UDA approach is a better match than the methods not benefiting from DA. Indeed, DANN in the UDA setting performs better than all baselines, which use only `Source`. The last observation is that DANN in SSDA setting does not improve the results over MLP(SDA). This result is surprising given the significant performance gain that DANN achieves over MLP in the absence of labeled data from target domain. However, a similar outcome, *i.e.* DANN failure in SSDA setting, has been reported previously on different benchmarks [17].

5.4 Feature Group Contributions

In this section, we aim to get some insight into the contributions of the feature groups described in Table 2. To this end, we train our best performing model, i.e. CatBoost(SDA), on different versions of the dataset, each of which containing all the feature groups except for one. The results of this experiment are illustrated in Table 4. The first column of the table (Full data) shows the performance of CatBoost(SDA) on the complete set of features.

As one can expect, the most significant impact comes from the Funding group, which makes sense since the valuation prediction that we considered in this study relies mainly on fundraising events. Nevertheless, even in the absence of information about the funding round, *ca.* 50% of the variability of the dependent variable is accounted for. Another observation is that the second most important group of features is the General group, comprising features such as the startup's age and its country of origin. Without this group, the model loses around 3% and 6.5% in terms of R^2 and RMSE respectively. This group is closely followed by the Financial group. As to the Social network group, its impact is relatively modest, though still statistically significant.

6 Conclusion

In this study, we investigated a real-world task of great importance: finding the undisclosed valuation of startups. To do that, we first collected data from Crunchbase and showed that there is a significant distributional shift between the labeled and the unlabeled data. We then used Companies House to partially annotate the unlabeled data and illustrated that these annotations are compatible with the Crunchbase data distributions. We then proposed to solve this problem in a Domain Adaptation (DA) setting and illustrated that DA based methods perform much better than other baselines. We also provided some insight into the impact of the different feature groups on the model's performance, which shows that, if the funding features are of primary importance to solve the valuation problem, the other groups work hand in hand to provide better valuation predictions.

References

1. List of European countries: Andorra, Albania, Austria, Åland Islands, Bosnia and Herzegovina, Belgium, Bulgaria, Belarus, Switzerland, Cyprus, Czech Republic, Germany, Denmark, Estonia, Spain, Finland, Faroe Islands, France, United Kingdom, Guernsey, Greece, Croatia, Hungary, Ireland, Isle of Man, Iceland, Italy, Jersey, Liechtenstein, Lithuania, Luxembourg, Latvia, Monaco, Moldova, Macedonia, Malta, Netherlands, Norway, Poland, Portugal, Romania, Russian Federation, Sweden, Slovenia, Svalbard and Jan Mayen, Slovakia, San Marino, Ukraine, Vatican City
2. Bonaventura, M., Ciotti, V., Panzarasa, P., Liverani, S., Lacasa, L., Latora, V.: Predicting success in the worldwide start-up network. Sci. Rep. **10**(1), 1–6 (2020)
3. Caruana, R., Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 161–168 (2006)
4. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(1), 2030–2096 (2016)
5. Gastaud, C., Carniel, T., Dalle, J.M.: The varying importance of extrinsic factors in the success of startup fundraising: competition at early-stage and networks at growth-stage (2019)
6. Hunter, D.S., Saini, A., Zaman, T.: Picking winners: A data driven approach to evaluating the quality of startup companies. arXiv preprint arXiv:1706.04229 (2017)
7. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
8. Kim, T., Kim, C.: Attract, perturb, and explore: learning a feature alignment network for semi-supervised domain adaptation. arXiv preprint arXiv:2007.09375 (2020)
9. Kouw, W.M., Loog, M.: A review of domain adaptation without target labels. IEEE Trans. Pattern Anal. Mach. Intell. **43**, 766–785 (2019)
10. Miloud, T., Aspelund, A., Cabrol, M.: Startup valuation by venture capitalists: an empirical study. Venture Capital **14**, 151–174 (2012)
11. Motiian, S., Piccirilli, M., Adjeroh, D.A., Doretto, G.: Unified deep supervised domain adaptation and generalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5715–5725 (2017)
12. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: ICML (2010)
13. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: Catboost: unbiased boosting with categorical features. In: Advances in Neural Information Processing Systems, pp. 6638–6648 (2018)
14. Quintero, S.: An empirical perspective on startup valuations (2019)

15. Reinfeld, P.: START-UP VALUATION Solving the valuation puzzle of new business ventures. Master's thesis, HEC Paris, XXX (2018)
16. Roe, B.P., Yang, H.J., Zhu, J., Liu, Y., Stancu, I., McGregor, G.: Boosted decision trees as an alternative to artificial neural networks for particle identification. Nucl. Instrum. Methods Phys. Res. Sect. A Accelerators Spectrometers Detectors Assoc. Equipment **543**(2–3), 577–584 (2005)
17. Saito, K., Kim, D., Sclaroff, S., Darrell, T., Saenko, K.: Semi-supervised domain adaptation via minimax entropy. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 8050–8058 (2019)
18. Sharchilev, B., Roizner, M., Rumyantsev, A., Ozornin, D., Serdyukov, P., de Rijke, M.: Web-based startup success prediction. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 2283–2291. ACM (2018)
19. Team, C.P.: Glossary of funding types (2020). https://support.crunchbase.com/hc/en-us/articles/115010458467-Glossary-of-Funding-Types, Accessed 14 Sep 2020
20. Xiang, G., Zheng, Z., Wen, M., Hong, J., Rose, C., Liu, C.: A supervised approach to predict company acquisition with factual and topic features using profiles and news articles on techcrunch. In: Sixth International AAAI Conference on Weblogs and Social Media (2012)
21. Zhang, Q., Ye, T., Essaidi, M., Agarwal, S., Liu, V., Loo, B.T.: Predicting startup crowdfunding success through longitudinal social engagement analysis. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1937–1946. ACM (2017)
22. Zhang, Y., Haghani, A.: A gradient boosting method to improve travel time prediction. Transp. Res. Part C Emerg. Technol **58**, 308–324 (2015)

Disparate Impact in Item Recommendation: A Case of Geographic Imbalance

Elizabeth Gómez[1], Ludovico Boratto[2(✉)], and Maria Salamó[1]

[1] Facultat de Matemàtiques i Informàtica, Universitat de Barcelona, Barcelona, Spain
egomezye13@alumnes.ub.edu, maria.salamo@ub.edu

[2] Data Science and Big Data Analytics, Eurecat - Centre Tecnológic de Catalunya, Barcelona, Spain
ludovico.boratto@acm.org

Abstract. Recommender systems are key tools to push items' consumption. Imbalances in the data distribution can affect the exposure given to providers, thus affecting their experience in online platforms. To study this phenomenon, we enrich two datasets and characterize data imbalance w.r.t. the country of production of an item (*geographic imbalance*). We focus on movie and book recommendation, and divide items into two classes based on their country of production, in a majority-versus-rest setting. To assess if recommender systems generate a disparate impact and (dis)advantage a group, we introduce metrics to characterize the visibility and exposure a group receives in the recommendations. Then, we run state-of-the-art recommender systems and measure the visibility and exposure given to each group. Results show the presence of a disparate impact that mostly favors the majority; however, factorization approaches are still capable of capturing the preferences for the minority items, thus creating a positive impact for the group. To mitigate disparities, we propose an approach to reach the target visibility and exposure for the disadvantaged group, with a negligible loss in effectiveness.

Keywords: Recommender systems · Bias · Disparate impact

1 Introduction

Recommender systems learn patterns from users' behavior, to understand what might be of interest to them [37]. Natural imbalances in the data (e.g., in the amount of observations for popular items) might be embedded in the patterns. The produced recommendations can amplify these imbalances and create biases [9]. When a bias is associated to sensitive attributes of the users (e.g., gender or race), negative societal consequences can emerge, such as unfairness [22,23,30,33]. Unfairness can affect all the stakeholders of a system [1,5].

Data imbalances might be inherently connected to the way an industry is composed, e.g., with certain items mainly produced in certain parts of the world,

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 190–206, 2021.
https://doi.org/10.1007/978-3-030-72113-8_13

and with consumption patterns that differ based on the country of the users [4]. In this paper, we focus on geographic imbalance and study the problem of how the country of production of an item can create a disparate impact to providers in the recommendations. We assess disparate impact by considering both the *visibility* received by the providers of a group (i.e., the percentage of recommendations having them as providers) and their *exposure*, which accounts for the position in which items are recommended [41]. Hence, with these two metrics we measure respectively, (*i*) the share of recommendations of a group and (*ii*) the relevance that is given to that group. Both metrics are important to assess disparate impact in this context. Visibility alone might lead a group of providers not being reached by users in case they appear only at the bottom of the list, and exposure alone might not guarantee providers enough sales (a single item at the top of the list would mean these providers are recommended only once).

We assess disparate impact by comparing the visibility and exposure given to a group of providers with the representation of the group in the data. We study two forms of representation, based on (*i*) the amount of items a group offers, or (*ii*) the amount of ratings given to the items of a group.

We consider two of the main domains in which recommender systems operate, namely movies and books. We show, by extending two real-world datasets with the country of production of the items, that both movie and book data is imbalanced towards the United States. To understand the impact of this imbalance, we divide items into two groups, in a majority-versus-rest setting, and study how this imbalance is reflected in the visibility and exposure given to providers of the two groups when producing recommendations.

We consider state-of-the-art recommender systems, covering both model- and memory-based approaches, and point- and pair-wise algorithms. While commonly studied sensitive attributes, such as gender, show a disparate impact effect at the expense of the minority group, our use-case presents several peculiarities. Indeed, user preferences do not reflect these imbalances and users equally like items coming from the majority (the United States) and the minority (the rest of the countries) groups. This leads to disparity scenarios that affect either the majority or the minority group, according to patterns we present in this study.

To mitigate disparities, we propose a re-ranking that optimizes both the visibility and exposure given to providers, based on their representation in the data. Hence, we consider a distributive norm based on *equity* [43]. Our approach introduces in the recommendations items that increase the visibility and exposure of a group, causing the minimum possible loss in user relevance.

Our contributions can be summarized as follows:

- We study, for the first time, the impact of geographic imbalance in the data on the visibility and exposure given to different provider groups;
- We extend two real-world datasets with the country of production of each item and characterize the link between geographic imbalance and disparate impact, uncovering the factors that lead a group to be under-/over-exposed;
- We propose a re-ranking mitigation strategy that can lead to the target visibility and exposure with the minimum possible losses in effectiveness;

- We evaluate our approach, showing we can mitigate disparities with a negligible loss in effectiveness.

The rest of the paper details in Sect. 2 related work, while in Sect. 3 the scenario, metrics, recommenders, and datasets. Section 4 assesses disparate impact phenomena. Section 5 contains our mitigation algorithm and results. Section 6 concludes the paper.

2 Related Work

This section covers related studies, starting from the concepts of visibility and exposure in ranking, and continuing with the impact of recommendation for providers. We conclude by contextualizing our work with the existing studies.

Visibility and Exposure in Rankings. Given a ranking, visibility and exposure metrics respectively assess the amount of times an item is present in the rankings [21,45] and *where* an item is ranked [8,46]. They were introduced in the context of non-personalized rankings, where the objects being ranked are individual users (e.g., job candidates). These metrics can operate at the *individual* level, thus guaranteeing that similar individuals are treated similarly [8,19], or at *group* level, by making sure that users belonging to different groups are given adequate visibility or exposure [19,45,46]. Under the group setting, the visibility/exposure of a group is proportional to its representation in the data [32,35,38,44].

Impact of Recommendations for Providers. The impact of the generated recommendations on the item providers is a concept known as *provider fairness (P-fairness)*. It guarantees that the providers of the recommended objects that belong to different groups or are similar at the individual level, will get recommended according to their representation in the data. In this domain, Ekstrand et al. [20] assessed that collaborative filtering methods recommend books of authors of a given gender with a distribution that differs from that of the original user profiles. Liu and Burke [29] propose a re-ranking function, which balances recommendation accuracy and fairness, by dynamically adding a bonus to the items of the uncovered providers. Sonboli and Burke [42] define the concept of local fairness, to equalize access to capital across all types of businesses. Mehrotra et al. [31] assess unfairness based on the popularity of the providers. Several policies are defined to study the trade-offs between user-relevance and fairness. Kamishima et al. [26] introduce recommendation independence, which leads to recommendations that are statistically independent of sensitive features.

Contextualizing Our Work. While our study draws from metrics derived from fairness, *this work does not directly mitigate fairness for the individual providers*. We study a broader phenomenon, i.e., *if an industry of a country is affected by how recommendations are produced in presence of data imbalance.*

Considering our use-cases, both cinema and literature are powerful vehicles for culture, education, leisure, and propaganda, as highlighted by the UNESCO[1]. Moreover, both domains have an impact on the economy of a country, with (sometimes public) investments for the production of movies/books that are expected to generate a return. Hence, considering how recommender systems can push the consumption of items of a country is a related but different problem w.r.t. provider fairness.

3 Preliminaries

Here, we present the preliminaries, to provide foundations to our work.

3.1 Recommendation Scenario

Let $U = \{u_1, u_2, ..., u_n\}$ be a set of users, $I = \{i_1, i_2, ..., i_j\}$ be a set of items, and V be a totally ordered set of values that can be used to express a preference. The set of ratings is a ternary relation $R \subseteq U \times I \times V$; each rating is denoted by r_{ui}. These ratings can directly feed an algorithm in the form of triplets (point-wise approaches) or shape user-item observations (pair-wise approaches).

To assess the real impact of the recommendations, we consider a temporal split of the data, where a fixed percentage of the ratings of the users (ordered by timestamp) goes to the training and the rest goes to the test set [6].

The recommendation goal is to learn a function f that estimates the relevance ($\hat{r}_{ui}$) of the user-item pairs that do not appear in the training data. We denote as $\hat{R}$ the set of recommendations, and as $\hat{R}_G$ those involving items of a group G.

Let C_i be the set of production countries of an item i. We use it to shape two groups, a majority $M = \{i \in I : 1 \in C_i\}$, and a minority $m = \{i \in I : 1 \notin C_i\}$. Note that 1 identifies the country associated to the majority group.

3.2 Metrics

Representation. The representation of a group is the amount of times that group appears in the data. We consider two forms of representation, based on (i) the amount of items offered by a group and (ii) the amount of ratings collected for that group. We define with $\mathcal{R}$ the *representation* of a group G ($G \in \{M, m\}$) ($\mathcal{R}_I$ denotes an item-based representation, while $\mathcal{R}_R$ a rating-based representation):

$$\mathcal{R}_I(G) = |G|/|I| \tag{1}$$

$$\mathcal{R}_R(G) = |\{r_{ui} : i \in G\}|/|R| \tag{2}$$

Equation (1) accounts for the proportion of items of a group, while Eq. (2) for the proportion of ratings associated to a group. Both metrics are between 0 and 1.

[1] https://publications.parliament.uk/pa/cm200203/cmselect/cmcumeds/667/667.pdf.

The representation of a group is measured by considering only the training set. It is trivial to notice that, given a group G, the representation of the other, $\overline{G}$, can be computed as $\mathcal{R}_*(\overline{G}) = 1 - \mathcal{R}_*(G)$ (where '*' refers to I or R).

Disparate Impact. We assess disparate impact with two metrics.

Definition 1 (Disparate visibility). *The* disparate visibility *of a group is computed as the difference between the share of recommendations for items of that group and the representation of that group:*

$$\Delta\mathcal{V}(G) = \frac{1}{|U|}\sum_{u\in U}\frac{|\{\hat{r}_{ui} : i \in \hat{R}_G\}|}{|\hat{R}|} - \mathcal{R}_*(G) \tag{3}$$

Its range is in $[-\mathcal{R}_*(G), 1 - \mathcal{R}_*(G)]$; it is 0 when there is no disparate visibility, while negative/positive values indicate that the group received a share of recommendations lower/higher than its representation. This metric is based on that considered by Fabbri et al. [21].

Definition 2 (Disparate exposure). *The* disparate exposure *of a group is the difference between the exposure obtained by the group in the recommendation lists [41] and the representation of that group:*

$$\Delta\mathcal{E}(G) = \frac{1}{|U|}\sum_{u\in U}\frac{\sum_{pos=1}^{k}\frac{1}{log_2(pos+1)}, \forall i \in \hat{R}_G}{\sum_{pos=1}^{k}\frac{1}{log_2(pos+1)}} - \mathcal{R}_*(G) \tag{4}$$

where pos is the position of an item in the top-k recommendations.

This metric also ranges in $[-\mathcal{R}_*(G), 1 - \mathcal{R}_*(G)]$; it is 0 when there is no disparate exposure, while negative/positive values indicate that the exposure given to the group in the recommendations is lower/higher than its representation.

Notice that the disparate visibility/exposure of one group can be computed as the opposite of the value obtained for the other group.

Remark. *We do not define a unique "disparate impact" metric, to control both visibility and exposure, so that* providers are recommended enough times and with enough exposure. *A unique metric would not allow us to balance both, by compressing everything in a unique number.*

3.3 Recommendation Algorithms

We consider five state-of-the-art Collaborative Filtering algorithms. As memory-based approaches, we consider the UserKNN [24] and ItemKNN [39] algorithms. For the class of matrix factorization based approaches, we consider the BPR [36], BiasedMF [28], and SVD++ [27] algorithms. To contextualize our results, we also consider two non-personalized algorithms (MostPopular and RandomGuess).

3.4 Datasets

MovieLens-1M (Movies). The dataset provides 1M ratings (range 1–5), provided by 6,040 users, to 3,600 movies. It contains the IMDb ID of each movie, which allowed us to associate it to its country of production thanks to the OMDB APIs[2] (note that *each movie may have more than one country of production*).

Book Crossing (Books). The dataset contains 356k ratings (in the range 1–10), given by 10,409 users, to 14,137 books. The dataset contained the ISBN code of each book, which was used to add information about its countries of production thanks to the APIs offered by the Global Register of Publishers[3].

For both datasets, we encoded the country of production with an integer, with the United States (which represents the majority group in both datasets) having ID 1, and the rest of the countries having subsequent IDs.

4 Disparate Impact Assessment

In this section, we run the algorithms presented in Sect. 3.3 to assess their effectiveness and the disparate impact they generate.

4.1 Experimental Setting

For both datasets presented in Sect. 3.4, the test set was composed by the most recent 20% of the ratings of each user. To run the recommendation algorithms presented in Sect. 3.3, we considered the LibRec library (version 2). For each user, we generate 150 recommendations (denoted in the paper as the top-n) so that we can mitigate disparate impact through a re-ranking algorithm. The final recommendation list for each user is composed by 20 items (denoted as top-k).

Each algorithm was run with the following hyper-parameters:

- **UserKNN.** similarity: Pearson; neighbors: 50; similarity shrinkage: 10;
- **ItemKNN.** similarity: Cosine for Movies and Pearson for Books; neighbors: 200 (Movies), 50 (Books); similarity shrinkage: 10;
- **BPR.** iterator learnrate: 0.1; iterator learnrate maximum: 0.01; iterator maximum: 150; user regularization: 0.01; item regularization: 0.01; factor number: 10; learnrate bolddriver: false; learnrate decay = 1.0;
- **BiasedMF.** iterator learnrate: 0.01; iterator learnrate maximum: 0.01; iterator maximum: 20 (Movies), 1 (Books); user regularization: 0.01; item regularization: 0.01; bias regularization: 0.01; number of factors: 10; learnrate bolddriver: false; learnrate decay: 1.0;
- **SVD++.** iterator learnrate: 0.01; iterator learnrate maximum: 0.01; iterator maximum: 10 (Movies), 1 (Books); user regularization: 0.01; item regularization: 0.01; impItem regularization: 0.001; number of factors: 10; learnrate bolddriver: false; learnrate decay: 1.0.

[2] http://www.omdbapi.com/.

[3] https://grp.isbn-international.org/search/piid_cineca_solr.

To evaluate recommendation effectiveness, we measure the ranking quality of the lists by measuring the *Normalized Discounted Cumulative Gain* (NDCG) [25].

$$DCG@k = \sum_{u \in U} \hat{r}_{ui}^{pos} + \sum_{pos=2}^{k} \frac{\hat{r}_{ui}^{pos}}{log_2(pos)} \qquad NDCG@k = \frac{DCG@k}{IDCG@k} \tag{5}$$

where $\hat{r}_{ui}^{pos}$ is relevance of item i recommended to user u at position pos. The ideal DCG is calculated by sorting items based on decreasing true relevance (true relevance is 1 if the user interacted with the item in the test set, 0 otherwise).

4.2 Characterizing User Behavior

This section characterizes the group representation and users' rating behavior.

Group Representation. In the Movies dataset, $\mathcal{R}_I(m) = 0.3$ and $\mathcal{R}_R(m) = 0.23$. In the Books dataset, instead, $\mathcal{R}_I(m) = 0.12$ and $\mathcal{R}_R(m) = 0.08$. Both datasets show a strong geographic imbalance, with the majority group covering 70% of the items in the first dataset and 88% in the second. This imbalance is worsened when we consider the ratings, since in the movie context the ratings associated to the majority are 77%, while in the book content the rating representation for the majority is 92%. It becomes natural to ask ourselves if the majority group also attracts better ratings, to assess if this exacerbated imbalance is because majority items are perceived as of higher quality.

Table 1. Results of state-of-the-art recommender systems. Normalized Discounted Cumulative Gain (NDCG); Disparate Visibility for the minority group when considering the item representation as a reference ($\Delta\mathcal{V}_I$); Disparate Exposure for the minority group when considering the item representation as a reference ($\Delta\mathcal{E}_I$); Disparate Visibility for the minority group when considering the rating- representation as a reference ($\Delta\mathcal{V}_R$); Disparate Exposure for the minority group when considering the rating representation as a reference ($\Delta\mathcal{E}_R$). The values in bold indicate the best result.

	Movies					Books				
Algorithm	NDCG	$\Delta\mathcal{V}_I$	$\Delta\mathcal{E}_I$	$\Delta\mathcal{V}_R$	$\Delta\mathcal{E}_R$	NDCG	$\Delta\mathcal{V}_I$	$\Delta\mathcal{E}_I$	$\Delta\mathcal{V}_R$	$\Delta\mathcal{E}_R$
MostPop	0.1109	−0.1802	−0.2016	−0.1089	−0.1302	0.0089	−0.1239	−0.1239	−0.0839	−0.0840
RandomG	0.0105	**0.0020**	**0.0027**	0.0733	0.0740	8.91E+11	**0.0013**	**0.0015**	0.0412	0.0415
UserKNN	0.1247	−0.1544	−0.1668	−0.0831	−0.0955	0.0053	−0.0438	−0.0360	**−0.0039**	**0.0039**
ItemKNN	0.1199	−0.1744	−0.1926	−0.1031	−0.1212	0.0075	−0.0799	−0.0790	−0.0400	−0.0390
BPR	**0.1395**	−0.1054	−0.1087	**−0.0340**	**−0.0373**	0.0054	−0.0257	−0.0259	0.0142	0.0141
BiasedMF	0.0588	0.0901	0.0954	0.1614	0.1668	**0.0103**	−0.1239	−0.1239	−0.0840	−0.0840
SVD++	0.0684	0.0742	0.0762	0.1455	0.1475	**0.0103**	−0.1239	−0.1239	−0.0840	−0.0840

Rating Behavior. We considered the average rating associated to the items of each group. In the Movies dataset, the average rating for the majority group is 3.56, while that of the minority group is 3.61. In the Books dataset, we observed an average rating of 4.38 for the majority, and of 4.43 for the minority. This shows that the preference of the users for the two groups does not differ.

Observation 1. *Both datasets expose a big geographic imbalance in the representation of each group, in terms of offered items. The majority group usually attracts more ratings, thus increasing the existing imbalance. However, the minority items are not considered as of lower quality for the users, since the average rating for both groups is the same in both datasets.*

4.3 Assessing Effectiveness and Disparate Impact

We assess disparate impact in terms of visibility and exposure. Table 1 presents the results obtained when generating a top-20 ranking for each user, considering as a reference the minority group. The first phenomenon that emerges is that both groups can be affected by disparate impact and that, when one group receives more visibility, it also receives more exposure; hence, when a group is favored in the amount of recommendations, it is also ranked higher.

Considering the Movies dataset, MostPop, UserKNN, ItemKNN, and BPR present a disparate visibility and exposure that disadvantage the minority, for both forms of representation. The point-wise Matrix Factorization algorithms (BiasedMF and SVD++) and RandomGuess, instead, advantage the minority. This goes in contrast with the literature on algorithmic bias and fairness, where the minority is usually disadvantaged. We conjecture that, since recommender systems do not receive any information about the geographic groups and since users equally prefer the items of the two groups, the point-wise Matrix Factorization approaches create factors that capture user preferences as a whole. Our results align with those of Cremonesi et al. [14], who showed the capability of factorization approaches to recommend long-tail items. Interestingly, when considering disparate visibility and exposure, the best results for the item-based representation are those of RandomGuess; nevertheless, the algorithm is also the least effective in terms of NDCG. No algorithm can offer both effectiveness and adapt to the offer of a country. When considering the rating-based representation, BPR is the most effective and has the lowest disparate visibility and exposure. Hence, the combination between factorization approaches and a pair-wise training can connect effectiveness and equity of visibility and exposure.

In the Books dataset, besides MostPop, all the approaches advantage the majority. This opposite trend in terms of disparate impact of the point-wise Matrix Factorization algorithms (BiasedMF and SVD++) w.r.t. the Movies dataset, can be explained by considering that the items having more ratings will lead to factors that have more weight at prediction stage; here, the majority is much larger than in the Movies dataset, so this leads to the group being advantaged in terms of visibility and exposure. This dataset is much also more sparse, so effectiveness is strongly reduced, and the point-wise Matrix Factorization approaches are the most effective. There is no connection between effectiveness and equity of exposure and visibility. Indeed, RandomGuess and UserKNN are, respectively, the best algorithms when considering the item-/rating-based representation of the groups. This good visibility and exposure provided by UserKNN in the rating-based setting can be connected to phenomena observed by Cañamares and Castells [11] since, under sparsity, the algorithm adapts to item popularity.

Observation 2. *Geographic imbalance almost always affects the minority group, since we feed algorithms with much more instances than their counterpart. Matrix Factorization based approaches can help the minority receive more visibility and exposure, with latent factors that capture preferences also of the minority. However, if the imbalance is too severe, the minority is always affected by disparate impact.*

5 Mitigating Disparate Impact

The previous section allowed us to observe a new phenomenon that departs from the existing algorithmic fairness studies, since *the minority group is not always the disadvantaged one when considering geographic imbalance*. Still, our results show that we can always observe a group receiving a disproportional visibility and exposure with respect to its representation in the data.

In this section, we mitigate these phenomena by presenting a re-ranking algorithm that introduces items of the disadvantaged group in the recommendation list, to reach a visibility and an exposure proportional to its representation.

A re-ranking algorithm is the only option when optimizing ranking-based metrics, like visibility and exposure. An in-processing regularization, such as those presented in [7,26], would not be possible, since at prediction stage the algorithm does not predict *if and where* an item will be ranked in a list. Re-rankings have been introduced to reduce disparities, both for non-personalized rankings [8,13,32,41,45,46] and for recommender systems [10,31], with approaches such as Maximal Marginal Relevance [12]. These algorithms optimize only one property (visibility or exposure), so no direct comparison is possible.

5.1 Algorithm

The foundation behind our mitigation algorithm is to *move up in the recommendation list the item that causes the minimum loss in prediction for all the users*. We start by targeting the desired visibility, to make sure the items of the disadvantaged group are recommended enough times. Then we move items up inside the recommendation list to reach the target exposure.

The mitigation is described in Algorithm 1. The inputs are the recommendations (top-n items), the current visibility and exposure of the disadvantaged

```
Input: recList: ranked list (records contain user, item, prediction, exposure,
       group, position), vis: visibility of disadvantaged group, exp: exposure of
       disadvantaged group, rep: representation of disadvantaged group, advG:
       ID of advantaged group, disadvG: ID of disadvantaged group
Output: reRankedList: ranked list adjusted by visibility and exposure
define optimizeVisibilityExposure (recList, vis, exp, rep)
begin
    reRankedList ← mitigation(recList, vis, rep, advG, disadvG,
    ”visibility”)
    reRankedList ← mitigation(reRankedList, exp, rep, advG, disadvG,
    ”exposure”)
    return reRankedList
end

define mitigation (list, VE, rep, advG, disadvG, rankingType)
begin
    for user ∈ list.users do
        losses.add(calculateLoss(list, user, rankingType, advG, disadvG)
    end
    while $VE < rep$ do
        minLoss ← losses.sortByLoss(0)
        list ← swap(list, minlLoss.itemAdvG, minLoss.itemDisadvG)
        if reRankingType == “visibility” then
            $VE \leftarrow VE + 1$
        else
            $VE \leftarrow (VE - minLoss.itemDisadvG.exposure) +$
            $minLoss.itemAdvG.exposure$
        end
        losses.add(calculateLoss(list, user, rankingType, advG, disadvG))
    end
    return list
end

define calculateLoss (list, user, rankingType, advG, disadvG)
begin
    itemAdvGroup ← getlastItem(list, user,top-k,advGroup)
    if reRankingType == “visibility” then
        itemDisadvGroup ← getfirstItem(list, user,last-n,disadvGroup)
    else
        while itemAdvGroup.position > itemDisadvGroup.position do
            itemDisadvGroup ← getnextItem(list, user,top-k,disadvGroup)
        end
    end
    loss ← itemAdvGroup.prediction - itemDisadvGroup.prediction
    lossUser ← [user, itemAdvGroup, itemDisadvGroup, loss]
    return lossUser
end
```

Algorithm 1: Visibility and exposure mitigation algorithm

group and its representation in the data (our target), and the IDs of the advantaged and disadvantaged groups. The output is the re-ranked list of items.

The *optimizeVisibilityExposure* method (lines 1–6), executes the mitigation, firstly to regulate the visibility of the disadvantaged group (by adding their items to the top-k) and secondly to regulate the exposure (by moving their items up in the top-k). The *mitigation* method (lines 7–23) regulates the visibility and exposure of the recommendation list. First, we loop over the users (lines 9–11) and call the *calculateLoss* method, to calculate the loss (in terms of items' predicted relevance) we would have in each user's list when swapping the items of the two groups. The while loop (lines 12–21) swaps the items until the target visibility/exposure is reached; line 13 returns the user that causes the minimum loss and line 14 swaps their items. If the goal is to reach a target visibility, lines 15–16 increase the visibility of the group by 1; if the swap is done to reach a target exposure, lines 17–19 subtract the exposure of the old item and add that of the new one. Finally, the *calculateLoss* method recalculates the loss for the user object of the swap and returns the re-ranked list.

The *calculateLoss* method (lines 24–37) identifies the user causing the minimal loss of predicted relevance. We select two items in the list of each user. The first is the last item of the advantaged group in the top-k (line 26). If we are

Table 2. Impact of mitigation on recommended lists with item-based representation. Normalized Discounted Cumulative Gain (NDCG); Disparate Visibility ($\Delta\mathcal{V}_I$) for the minority; Disparate Exposure ($\Delta\mathcal{E}_I$) for the minority. We report below gain/loss of each setting w.r.t. the original one (left side of Table 1).

	Mitigation visibility & exposure					
	Movies			Books		
Algorithm	NDCG	$\Delta\mathcal{V}_I$	$\Delta\mathcal{E}_I$	NDCG	$\Delta\mathcal{V}_I$	$\Delta\mathcal{E}_I$
MostPop	0.1052	−0.0017	−0.0017	0.0087	−0.0039	−0.0039
(gain/loss)	−0.0057	0.1785	0.1999	−0.0002	0.1200	0.1200
RandomG	0.0106	−0.0017	−0.0017	8.91E+11	−0.0039	−0.0039
(gain/loss)	0.0001	−0.0036	−0.0043	3.24E+09	−0.0052	−0.0055
UserKNN	0.1205	−0.0017	−0.0017	0.0050	−0.0039	−0.0039
(gain/loss)	−0.0042	0.1528	0.1652	−0.0003	0.0399	0.0321
ItemKNN	0.1173	−0.0017	−0.0017	0.0075	−0.0039	−0.0039
(gain/loss)	−0.0027	0.1727	0.1909	0.0000	0.0760	0.0751
BPR	**0.1372**	**−0.0017**	**−0.0017**	0.0055	−0.0039	−0.0039
(gain/loss)	−0.0023	0.1037	0.1070	0.0001	0.0218	0.0220
BiasedMF	0.0623	−0.0017	−0.0017	**0.0119**	**−0.0039**	**−0.0039**
(gain/loss)	0.0035	−0.0918	−0.0971	0.0016	0.1200	0.1200
SVD++	0.0712	−0.0017	−0.0017	0.0113	−0.0039	−0.0039
(gain/loss)	0.0028	−0.0759	−0.0779	0.0011	0.1200	0.1200

regulating visibility, lines 27–28 select the first item of the disadvantaged group out of the top-k (denoted as last-n). Lines 29–33 mitigate for exposure; the while selects an item of the disadvantaged group that in the top-k is currently ranked lower than that of its counterpart. Once we obtain the pair of items for the user, we calculate the loss by considering the *prediction* attribute (line 34). Finally, line 35 collects the loss of the user, which is returned in line 36.

5.2 Impact of Mitigation

In this section, we assess the impact of our mitigation. Since we split data temporally, we cannot run statistical tests to assess the difference in the results, so we highlight the gain/loss obtained for each measure.

Results are reported in Tables 2 and 3 separating them between item- and rating-based representation of the groups. Trivially, given a target representation and a dataset, all algorithms achieve the same disparate visibility/exposure. Let us consider the trade-off between disparate visibility/exposure and effectiveness. Considering the Movies dataset, in both representations of the groups, BPR is the algorithm with the best trade-off between effectiveness and equity of visibility and exposure. It was already the most accurate algorithm, and thanks to our mitigation based on the minimum-loss principle, the loss in NDCG was

Table 3. Impact of mitigation on recommended lists with rating-based representation. Normalized Discounted Cumulative Gain (NDCG); Disparate Visibility ($\Delta\mathcal{V}_R$) for the minority; Disparate Exposure ($\Delta\mathcal{E}_R$) for the minority. We report below gain/loss of each setting w.r.t. the original one (left side of Table 1).

	Mitigation visibility & exposure					
	Movies			Books		
Algorithm	NDCG	$\Delta\mathcal{V}_R$	$\Delta\mathcal{E}_R$	NDCG	$\Delta\mathcal{V}_R$	$\Delta\mathcal{E}_R$
MostPop	0.1076	−0.0003	−0.0003	0.0089	−0.0040	−0.0040
(gain/loss)	−0.0032	0.1085	0.1299	−0.0006	0.0800	0.0800
RandomG	0.0112	−0.0003	−0.0003	8.54E+11	−0.0040	−0.0040
(gain/loss)	0.0006	−0.0736	−0.0743	−2.37E+10	−0.0452	−0.0455
UserKNN	0.1239	−0.0003	−0.0003	0.0050	−0.0040	−0.0040
(gain/loss)	−0.0008	0.0828	0.0952	−0.0003	−0.0001	−0.0079
ItemKNN	0.1185	−0.0003	−0.0003	0.0075	−0.0040	−0.0040
(gain/loss)	−0.0015	0.1027	0.1209	0.0001	0.0360	0.0351
BPR	**0.1390**	**−0.0003**	**−0.0003**	0.0053	−0.0040	−0.0040
(gain/loss)	−0.0005	0.0337	0.0370	−0.0001	−0.0182	−0.0180
BiasedMF	0.0648	−0.0003	−0.0003	**0.0122**	**−0.0040**	**−0.0040**
(gain/loss)	0.0060	−0.1618	−0.1671	0.0016	0.0800	0.0800
SVD++	0.0735	−0.0003	−0.0003	0.0113	−0.0040	−0.0040
(gain/loss)	0.0051	−0.1459	−0.1479	0.0011	0.0800	0.0800

negligible. In the Books dataset, BiasedMF confirms to be the best approach, in both effectiveness and equity of visibility and exposure. It is interesting to observe that, in both scenarios, MostPop is the second most effective algorithm and now provides the same visibility and exposure as the other algorithms; this is due to popularity bias phenomena [2], and their analysis is left as future work.

Observation 3. *When providing a re-ranking based on minimal predicted loss, the effectiveness remains stable, but disparate visibility and disparate exposure are mitigated.*

6 Conclusions and Future Work

In this paper, we considered data imbalance in the items' country of production of items (*geographic imbalance*). We considered a group setting based on a majority-versus-rest split of the items and defined measures to assess disparate visibility and disparate exposure for groups. The results of five collaborative filtering approaches show that the minority group is not always disadvantaged.

We proposed a mitigation algorithm that produces a re-ranking, by adding to the recommendation lists items that cause the minimum loss in predicted relevance. Results show that *thanks to our approach, any recommendation algorithm can bring equity of visibility and exposure to providers, without impacting the end-users in terms of effectiveness.*

Future work will study geographic imbalance in education, to explore country-based disparities for teachers [3,16–18]. Moreover, we will evaluate divergence-based disparity metrics [15]) and consider multi-class group settings. Other issues emerging from imbalanced groups, such as bribing [34,40], will be considered.

Acknowledgments. This research was partially funded by project 2017-SGR-341, MISMIS-LANGUAGE (grant No. PGC2018-096212-B-C33) from the Spanish Ministry of Science and Innovation, and NanoMoocs (grant No. COMRDI18-1-0010) from ACCIÓ. L. Boratto acknowledges Agència per a la Competivitat de l'Empresa, ACCIÓ, for their support under project "Fair and Explainable Artificial Intelligence (FX-AI)".

References

1. Abdollahpouri, H., et al.: Multistakeholder recommendation: survey and research directions. User Model. User-Adap. Interact. **30**(1), 127–158 (2020). https://doi.org/10.1007/s11257-019-09256-1
2. Abdollahpouri, H., Mansoury, M.: Multi-sided exposure bias in recommendation (2020)
3. Barra, S., Marras, M., Fenu, G.: Continuous authentication on smartphone by means of periocular and virtual keystroke. In: Au, M.H., et al. (eds.) NSS 2018. LNCS, vol. 11058, pp. 212–220. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02744-5_16

4. Bauer, C., Schedl, M.: Global and country-specific mainstreaminess measures: definitions, analysis, and usage for improving personalized music recommendation systems. PLOS ONE **14**(6), 1–36 (2019). https://doi.org/10.1371/journal.pone.0217389
5. Bauer, C., Zangerle, E.: Leveraging multi-method evaluation for multi-stakeholder settings. CoRR abs/2001.04348 (2020)
6. Bellogín, A., Castells, P., Cantador, I.: Statistical biases in information retrieval metrics for recommender systems. Inf. Retrieval J. **20**(6), 606–634 (2017). https://doi.org/10.1007/s10791-017-9312-z
7. Beutel, A., et al.: Fairness in recommendation ranking through pairwise comparisons. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, pp. 2212–2220. ACM (2019). https://doi.org/10.1145/3292500.3330745
8. Biega, A.J., Gummadi, K.P., Weikum, G.: Equity of attention: amortizing individual fairness in rankings. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, pp. 405–414. ACM (2018). https://doi.org/10.1145/3209978.3210063
9. Boratto, L., Fenu, G., Marras, M.: The effect of algorithmic bias on recommender systems for massive open online courses. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11437, pp. 457–472. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15712-8_30
10. Burke, R., Sonboli, N., Ordonez-Gauger, A.: Balanced neighborhoods for multi-sided fairness in recommendation. In: Conference on Fairness, Accountability and Transparency, FAT 2018, Proceedings of Machine Learning Research, vol. 81, pp. 202–214. PMLR (2018)
11. Cañamares, R., Castells, P.: A probabilistic reformulation of memory-based collaborative filtering: implications on popularity biases. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 215–224. ACM (2017). https://doi.org/10.1145/3077136.3080836
12. Carbonell, J.G., Goldstein, J.: The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: SIGIR 1998: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 335–336. ACM (1998). https://doi.org/10.1145/290941.291025
13. Celis, L.E., Straszak, D., Vishnoi, N.K.: Ranking with fairness constraints. In: 45th International Colloquium on Automata, Languages, and Programming, ICALP 2018. LIPIcs, vol. 107, pp. 28:1–28:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018). https://doi.org/10.4230/LIPIcs.ICALP.2018.28
14. Cremonesi, P., Koren, Y., Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In: Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, pp. 39–46. ACM (2010). https://doi.org/10.1145/1864708.1864721
15. Deldjoo, Y., Anelli, V.W., Zamani, H., Kouki, A.B., Noia, T.D.: Recommender systems fairness evaluation via generalized cross entropy. In: Burke, R., Abdollahpouri, H., Malthouse, E.C., Thai, K.P., Zhang, Y. (eds.) Proceedings of the Workshop on Recommendation in Multi-stakeholder Environments Co-located with the 13th ACM Conference on Recommender Systems (RecSys 2019), Copenhagen, Denmark, 20 September 2019, CEUR Workshop Proceedings, vol. 2440. CEUR-WS.org (2019)

16. Dessì, D., Dragoni, M., Fenu, G., Marras, M., Reforgiato Recupero, D.: Evaluating neural word embeddings created from online course reviews for sentiment analysis. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, pp. 2124–2127. ACM (2019). https://doi.org/10.1145/3297280.3297620
17. Dessì, D., Fenu, G., Marras, M., Reforgiato Recupero, D.: Leveraging cognitive computing for multi-class classification of E-learning videos. In: Blomqvist, E., Hose, K., Paulheim, H., Ławrynowicz, A., Ciravegna, F., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10577, pp. 21–25. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70407-4_5
18. Dessì, D., Fenu, G., Marras, M., Reforgiato Recupero, D.: COCO: semantic-enriched collection of online courses at scale with experimental use cases. In: Rocha, Á., Adeli, H., Reis, L.P., Costanzo, S. (eds.) WorldCIST'18 2018. AISC, vol. 746, pp. 1386–1396. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77712-2_133
19. Diaz, F., Mitra, B., Ekstrand, M.D., Biega, A.J., Carterette, B.: Evaluating stochastic rankings with expected exposure. CoRR abs/2004.13157 (2020)
20. Ekstrand, M.D., Tian, M., Kazi, M.R.I., Mehrpouyan, H., Kluver, D.: Exploring author gender in book rating and recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, pp. 242–250. ACM (2018). https://doi.org/10.1145/3240323.3240373
21. Fabbri, F., Bonchi, F., Boratto, L., Castillo, C.: The effect of homophily on disparate visibility of minorities in people recommender systems. In: Proceedings of the Fourteenth International AAAI Conference on Web and Social Media, ICWSM 2020, pp. 165–175. AAAI Press (2020)
22. Fenu, G., Lafhouli, H., Marras, M.: Exploring algorithmic fairness in deep speaker verification. In: Gervasi, O., et al. (eds.) ICCSA 2020. LNCS, vol. 12252, pp. 77–93. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58811-3_6
23. Hajian, S., Bonchi, F., Castillo, C.: Algorithmic bias: from discrimination discovery to fairness-aware data mining. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2125–2126. ACM (2016). https://doi.org/10.1145/2939672.2945386
24. Herlocker, J.L., Konstan, J.A., Riedl, J.: An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Inf. Retrieval **5**(4), 287–310 (2002). https://doi.org/10.1023/A:1020443909834
25. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. **20**(4), 422–446 (2002). https://doi.org/10.1145/582415.582418
26. Kamishima, T., Akaho, S., Asoh, H., Sakuma, J.: Recommendation independence. In: Conference on Fairness, Accountability and Transparency, FAT 2018, Proceedings of Machine Learning Research, vol. 81, pp. 187–201. PMLR (2018)
27. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426–434. ACM (2008). https://doi.org/10.1145/1401890.1401944
28. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Comput. **42**(8), 30–37 (2009). https://doi.org/10.1109/MC.2009.263
29. Liu, W., Burke, R.: Personalizing fairness-aware re-ranking. CoRR abs/1809.02921 (2018)

30. Marras, M., Korus, P., Memon, N.D., Fenu, G.: Adversarial optimization for dictionary attacks on speaker verification. In: Kubin, G., Kacic, Z. (eds.) Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15–19 September 2019, pp. 2913–2917. ISCA (2019). https://doi.org/10.21437/Interspeech.2019-2430
31. Mehrotra, R., McInerney, J., Bouchard, H., Lalmas, M., Diaz, F.: Towards a fair marketplace: counterfactual evaluation of the trade-off between relevance, fairness & satisfaction in recommendation systems. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, pp. 2243–2251. ACM (2018). https://doi.org/10.1145/3269206.3272027
32. Patro, G.K., Biswas, A., Ganguly, N., Gummadi, K.P., Chakraborty, A.: FairRec: two-sided fairness for personalized recommendations in two-sided platforms. In: WWW 2020: The Web Conference 2020, pp. 1194–1204. ACM/IW3C2 (2020). https://doi.org/10.1145/3366423.3380196
33. Ramos, G., Boratto, L.: Reputation (in)dependence in ranking systems: demographics influence over output disparities. In: Huang, J., et al. (eds.) Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020, Virtual Event, China, 25–30 July 2020, pp. 2061–2064. ACM (2020). https://doi.org/10.1145/3397271.3401278
34. Ramos, G., Boratto, L., Caleiro, C.: On the negative impact of social influence in recommender systems: a study of bribery in collaborative hybrid algorithms. Inf. Process. Manag. **57**(2), 102058 (2020). https://doi.org/10.1016/j.ipm.2019.102058
35. Ramos, G., Caleiro, C.: A novel similarity measure for group recommender systems with optimal time complexity. In: Boratto, L., Faralli, S., Marras, M., Stilo, G. (eds.) BIAS 2020. CCIS, vol. 1245, pp. 95–109. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-52485-2_10
36. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp. 452–461. AUAI Press (2009)
37. Ricci, F., Rokach, L., Shapira, B.: Recommender systems: introduction and challenges. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 1–34. Springer, Boston, MA (2015). https://doi.org/10.1007/978-1-4899-7637-6_1
38. Sapiezynski, P., Zeng, W., Robertson, R.E., Mislove, A., Wilson, C.: Quantifying the impact of user attentionon fair group representation in ranked lists. In: Companion of The 2019 World Wide Web Conference, WWW 2019, pp. 553–562. ACM (2019). https://doi.org/10.1145/3308560.3317595
39. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the Tenth International World Wide Web Conference, WWW 10, pp. 285–295. ACM (2001). https://doi.org/10.1145/371920.372071
40. Saúde, J., Ramos, G., Caleiro, C., Kar, S.: Reputation-based ranking systems and their resistance to bribery. In: Raghavan, V., Aluru, S., Karypis, G., Miele, L., Wu, X. (eds.) 2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, 18–21 November 2017, pp. 1063–1068. IEEE Computer Society (2017). https://doi.org/10.1109/ICDM.2017.139
41. Singh, A., Joachims, T.: Fairness of exposure in rankings. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, pp. 2219–2228. ACM (2018). https://doi.org/10.1145/3219819.3220088

42. Sonboli, N., Burke, R.: Localized fairness in recommender systems. In: Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization, UMAP 2019, pp. 295–300. ACM (2019). https://doi.org/10.1145/3314183.3323845
43. Walster, E., Berscheid, E., Walster, G.W.: New directions in equity research. J. Pers. Soc. Psychol. **25**(2), 151 (1973)
44. Yang, K., Stoyanovich, J.: Measuring fairness in ranked outputs. In: Proceedings of the 29th International Conference on Scientific and Statistical Database Management, pp. 22:1–22:6. ACM (2017). https://doi.org/10.1145/3085504.3085526
45. Zehlike, M., Bonchi, F., Castillo, C., Hajian, S., Megahed, M., Baeza-Yates, R.: FA*IR: a fair top-k ranking algorithm. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, pp. 1569–1578. ACM (2017). https://doi.org/10.1145/3132847.3132938
46. Zehlike, M., Castillo, C.: Reducing disparate exposure in ranking: a learning to rank approach. In: WWW 2020: The Web Conference 2020, pp. 2849–2855. ACM/IW3C2 (2020). https://doi.org/10.1145/3366424.3380048

You Get What You Chat: Using Conversations to Personalize Search-Based Recommendations

Ghazaleh H. Torbati(✉), Andrew Yates, and Gerhard Weikum

Max-Planck Institute for Informatics, Saarland Informatics Campus,
Saarbrücken, Germany
{ghazaleh,ayates,weikum}@mpi-inf.mpg.de

Abstract. Prior work on personalized recommendations has focused on exploiting explicit signals from user-specific queries, clicks, likes and ratings. This paper investigates tapping into a different source of implicit signals of interests and tastes: online chats between users. The paper develops an expressive model and effective methods for personalizing search-based entity recommendations. User models derived from chats augment different methods for re-ranking entity answers for medium-grained queries. The paper presents specific techniques to enhance the user models by capturing domain-specific vocabularies and by entity-based expansion. Experiments are based on a collection of online chats from a controlled user study covering three domains: books, travel, food. We evaluate different configurations and compare chat-based user models against concise user profiles from questionnaires. Overall, these two variants perform on par in terms of NCDG@20, but each has advantages on certain domains.

Keywords: Search-based recommendation · User modeling · Personalization

1 Introduction

Motivation: Recommender systems are at the heart of *personalized* shopping and online services for music and video streaming, hotels and restaurants, or food recipes [6,18,32]. *Search-based recommendation* is a setting where the user starts with a query and the recommendation model determines the result ranking based on the user's interests and preferences. This paper considers medium-grained queries about product entities (books, food recipes, and travel destinations) such as *paranormal romance* or *wine lover destinations* – in contrast to coarse-grained queries such as *love novels* or *Europe* and fine-grained queries such as *similar to Stephenie Meyer's Twilight* or *vineyards of the Bourgogne*. Results are assumed to come from a search engine (restricted to suitable domains for the respective vertical). Therefore, the personalization amounts to *re-ranking* the top results with regard to a model of the user's individual tastes.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 207–223, 2021.
https://doi.org/10.1007/978-3-030-72113-8_14

For this setting, the *user model* or *profile* can be represented explicitly in a personal knowledge base [5] or implicitly in a latent model [22,51]. These models can be constructed from various kinds of observations on user behavior:

A: Explicit signals like clicks, likes, ratings and purchases.
B: User profiles such as adssettings.google.com where users can see and check or un-check topics (even if the profile itself is learned from other signals).
C: Implicit signals from other online behavior, like social media posts or conversations with other users.

Option A is most widely used in practice (e.g., [19,25,52]) and includes standard recommenders based on collaborative filtering [35]. However, this rich kind of data is available only to major service providers, such as music streaming where playlists and other I-like-the-song signals are abundant. Option B operates on concise digests of user interests and item properties, for example, a list of topics and tags (e.g., [45]). This is less informative than A, but has the advantage that the user can easily interpret her profile and adjust it at her discretion (e.g., dropping a topic that is unwanted). Option C has been studied for recommending news and discussions, but the best signals are still the user histories of clicks, dwell times and likes (e.g., [28,44]). For search-based recommendation of product entities, C has not been explored at all, except for the specific case of leveraging product reviews (e.g., [7,14,34]).

This paper focuses on option C. It investigates how online chats between users can be leveraged for personalization in the outlined setting. To the best of our knowledge, it is the first work that studies chats as a source for search-based recommendation.

Research Questions: We investigate the following research questions:

- RQ0: How can we leverage signals from *user-user chats* to personalize search-based recommendations across a *variety of domains*: books, food recipes, and travel destinations?
- RQ1: How do methods that tap into individual *conversations* compare to methods that merely access *concise user profiles*?
- RQ2: How important is it to *customize* the per-user models to the *specific domain* at hand, for example, books vs. travel?
- RQ3: How much added value can we get from *entity awareness*: detecting named entities in user chats, mapping them to a background knowledge base, and using that information for expansion of user models and re-ranking techniques?

Contributions: We devise techniques for constructing language models and using them for re-ranking, with various components derived from chats: i) computing domain-specific vocabularies and ii) entity detection and entity-based expansions. The chats are recorded real-time conversations, gathered in a substantial user study with 14 students and 83 pair-wise chats (with 9,797 utterances and 59k tokens in total and a total duration of 93 h). We contrast chat-based personalization against techniques that merely build on concise user profiles derived from short questionnaires [43]. The paper makes the following contributions:

- It is the first approach to consider user chats as a source for search-based recommendation across a variety of vertical domains. Chats are a rich source of information about individual interests and tastes. In contrast to latent models learned from clicks, likes, ratings, etc., a user can more easily interpret and edit/censor this information to selectively restrict its usage for privacy reasons.
- We systematically compare chat-based personalization against a more restrictive approach that merely uses concise user profiles based on short questionnaires. In our experiments, both show advantages in certain domains, and perform on par overall.
- We devise techniques for per-domain customization by controlling the vocabulary and appropriate weighting of terms, and report on their experimental effectiveness.
- We devise techniques to harness entities and background knowledge in the construction of user models, and report on their experimental effectiveness.
- We release a dataset consisting of filled questionnaires, pair-wise user chats, document URLs, and search result assessments by users for three domains (books, travel, food). The data is available at http://personalization.mpi-inf.mpg.de/.

2 Computational Model and Re-ranking

We approach the personalization of entity-search answers by re-ranking a pool of initial non-personalized results using three different methods for scoring and ranking: the BM25 family, statistical language models, and neural ranking. Beginning with these ranking methods, we incorporate a user model to personalize results and domain-specific term weights to identify terms that are informative with a domain. We additionally apply expansion techniques to expand entities found in the user model. Rerankers thus consider a user model in addition to queries and documents. In our setting,

- **Queries** are short, medium-grained bags of words (or phrases), such as "scandinavian suspense" (for the books domain) or "wine lover destinations" (travel).
- **Documents** are entity-level answers obtained from specific websites that provide comprehensive contents about three domains: goodreads.com for books, wikivo-yage.org for travel, and allrecipes.com for food. Each answer has a key entity that can be easily identified (e.g., from the URL string or page title) and comprises an informative description of the entity. Two of the sites include also extensive reviews and discussion by their communities.

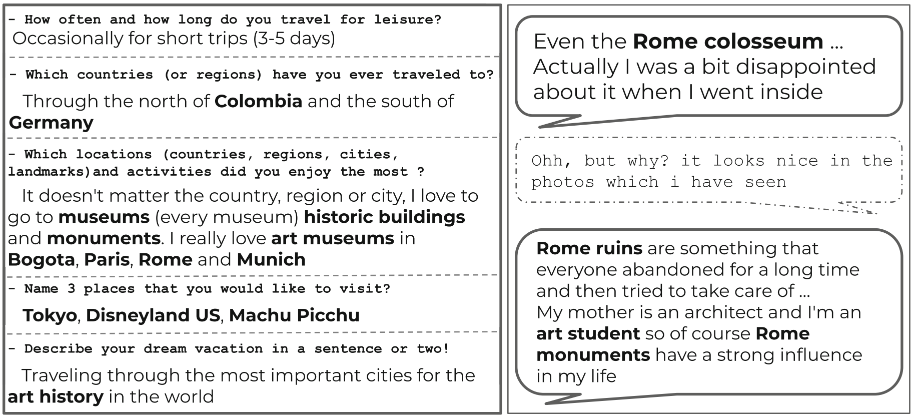
- **User models** represent a user's interests and tastes as a bag of words (or n-grams) taken from either a short *questionnaire/profile* filled in by the user or a *collection of online chats* with other users. Both of these options are further refined by instructing users to focus on specific scopes: general, books, travel, and food. This yields 8 basic options for the user model, which we further augment with techniques for domain-specific vocabularies and entities.

For illustration, Fig. 1 shows excerpts from the questionnaire and the chat collection for an example user. For the query "temples and culture", this user-specific information led to high ranks of travel destinations like Borobudur, Delphi and Ellora – all confirmed as very good recommendations by that user.

2.1 Re-ranking Methods

Given a query q, a user model u, and a document d from a pool of non-personalized results, we personalize the results by re-ranking them according to the user model. We explore three re-ranking methods for doing this.

Language Models: The first variant for re-ranking is based on language models (LMs) [50], which provide a natural way to incorporate the user model. We compute the Kullback-Leibler divergence between a query model and a document model with Dirichlet smoothing over unigrams or n-grams. In pilot experiments, unigrams outperformed bigrams and trigrams; hence we focus on the unigram case. To personalize for a specific user, we compute the Kullback-Leibler divergence i) between the query q and the document d and ii) between the user model u and the document d. These two components are combined into a linear mixture with hyper-parameter λ. Additionally, we incorporate a background model C for smoothing, based on ClueWeb'09. That is,



- How often and how long do you travel for leisure?
Occasionally for short trips (3-5 days)

- Which countries (or regions) have you ever traveled to?
Through the north of **Colombia** and the south of **Germany**

- Which locations (countries, regions, cities, landmarks) and activities did you enjoy the most ?
It doesn't matter the country, region or city, I love to go to **museums** (every museum) **historic buildings** and **monuments**. I really love **art museums** in **Bogota**, **Paris**, **Rome** and **Munich**

- Name 3 places that you would like to visit?
Tokyo, **Disneyland US**, **Machu Picchu**

- Describe your dream vacation in a sentence or two!
Traveling through the most important cities for the **art history** in the world

Even the **Rome colosseum** ... Actually I was a bit disappointed about it when I went inside

Ohh, but why? it looks nice in the photos which i have seen

Rome ruins are something that everyone abandoned for a long time and then tried to take care of ... My mother is an architect and I'm an **art student** so of course **Rome monuments** have a strong influence in my life

Fig. 1. Excerpts from user questionnaire and chat on travel domain (with recognized named entities and concepts in boldface)

$$
\begin{aligned}
score(q,d,u) \propto -(\lambda div(\theta_q \| \theta_d) + (1-\lambda) div(\theta_u \| \theta_d)) \propto \\
-\lambda \sum_{w \in V_q} \text{spy}(w) \cdot p(w|\theta_q) \log \frac{p(w|\theta_q)}{(p(w|\theta_d) + \mu p(w|\theta_C)) \;/\; (|d| + \mu)} \\
-(1-\lambda) \sum_{w \in V_u} \text{spy}(w) \cdot p(w|\theta_u) \log \frac{p(w|\theta_u)}{(p(w|\theta_d) + \mu p(w|\theta_C)) \;/\; (|d| + \mu)}
\end{aligned}
\tag{1}
$$

where V_u and V_q are the vocabularies of the user and query models, and θ_q, θ_u, θ_d and θ_C denote the multinomial parameters of query, user, document and background models, with Dirichlet smoothing parameter μ set to the average document length. We introduce additional weights spy(t) which reflect the specificity of a term t for a given domain (books, food or travel), as described in Sect. 3. This can be viewed as conditioning the query and user models with a domain model.

Optionally, we integrate word embeddings by using the cosine distance between precomputed word2vec embeddings [31] as a term-term similarity score $sim(w,t)$. This is plugged into the document model by means of a translation model largely following [24], with per-term contributions $p(w|\theta_d)$ replaced by summing over all similar terms (above a threshold): $\sum_{t:w\sim t} sim(w,t) \cdot p(t|\theta_d)$.

BM25: The second variant for re-ranking is the Okapi BM25 model [33]. We incorporate the user model by query expansion. In principle, all terms from the entire chat collection of a user are added to the query. We will discuss ways of reducing noise and focusing the query in Sects. 3 and 4. That is,

$$
score(q \cup u,\; d) \propto \sum_{w \in V_{q \cup u}} \text{spy}(w) \cdot \text{idf}(w) \cdot \frac{\text{tf}(w,d) \cdot (k_1 + 1)}{\text{tf}(w,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)} \tag{2}
$$

with domain-specificity weight spy(w), document length $|d|$, average document length $avgdl$, and BM25 parameters b and k_1.

Neural Ranking with KNRM: The third variant for re-ranking is the KNRM neural method [46] which takes a bag-of-words query as input. KNRM produces a query-document relevance score by comparing embedding similarities between query and document terms. During training, KNRM learns how to weigh different embedding similarity levels. As with BM25, we incorporate the user model by query expansion.

3 Domain-Specific Vocabulary Weighting

As described in the previous section, the ranking models are further augmented by awareness of domain-specific vocabularies, customizing the user models and document models to books, travel or food, respectively. The intuition is that terms in a user chat are informative if they refer to a certain meaning within a particular domain. For example, terms like “history” or “museum” are good cues about a user’s travel interests, whereas terms like “price” or “bargain” are uninformative – although all these terms have comparable idf values in large corpora.

We incorporate this idea of domain specificity by computing per-domain weights for terms, and weighing term contributions by the various ranking models accordingly (or even eliminating low-weight terms). To this end, we estimate the conditional probability of a term occurring in a domain-specific context (document or chat) given that it occurs in a general corpus:

$$\text{spy}(w) = P(w \in Dom | w \in All) \propto \frac{tf(w \in Dom)/|Dom|}{tf(w \in All)/|All|} \tag{3}$$

As underlying text collections for this estimator, we use the pool of all retrieved documents per domain (e.g., all answers for book search, including book descriptions and user reviews) against the pool of documents for all three domains together. We also experimented with term weighting for user-specific vocabularies, contrasting all chats by the same user against a universal corpus. This did not lead to significant changes in the empirical results, though, and is disregarded in the following.

4 Entity Expansion

Named Entity Recognition and Disambiguation (NER/NED): Among all terms and phrases in the user’s chats and questionnaires, entities and concepts deserve specific treatment. We ran standard NER (stanfordnlp.github.io) and NED (github.com/ambiverse-nlu) tools to link text spans to uniquely identified entities in the YAGO knowledge base, which in turns links most of these to Wikipedia. However, the NER stage produced both many false positives and false negatives. This is largely caused by the very colloquial nature of user chats, with short-hands, misspellings, ungrammatical utterances and ad-hoc choice between upper-case and lower-case. To mitigate this effect, we hired crowdsourcing workers to mark up text spans for entities and also general concepts that exist in YAGO and Wikipedia (e.g., “history“ or “Buddhist art”). This way we eliminated nearly all NER errors. As a result, the NED stage performed well, with precision reaching approximately 0.83 (estimated by sampling). We obtained this perfect mark-up only for NER as this is much easier for crowd workers than NED.

User Model Expansion: Rather than adding the names of these detected and linked entities to the user model directly, which is likely to overfit given that we deal with many long-tail entities (e.g., lesser-known books or special travel destinations), we experimented with expanding entities using embeddings and Wikipedia descriptions. We first conducted pilot experiments with entity embeddings using Wikipedia2vec [47,48] to achieve proper generalization, but this did not perform well: many terms that are highly related by Wikipedia2vec are quite uninformative if not misleading (e.g., history being most related to literature; modern, natural, and wine being most related to coffee, beer, food). Ultimately, to avoid this noise and topical drift, we expanded the entities using their descriptions from (the first paragraph of) their Wikipedia articles. This captures, for example, content sketches of books, highlights of travel destinations, etc. The resulting terms were added to general as well as domain-specific user models. For the latter, we computed the domain specificity of an entity and its descriptive terms, using the weighting model of Sect. 3.

Selective Expansion by Domain-Specificity: Some of the extracted entities may be poor cues for a certain target domain (e.g. a user chatting about "Italian cuisine" is not helpful for books and could even be misleading for travel). To counter this potential dilution, we use the domain-specificity of entities to filter the candidate entities before expanding the user model.

To this end, we construct a domain model for each of the three domains using Wikipedia2vec embeddings which capture both entity-level linkage and textual descriptions [47,48]. Candidate entities are mapped into the same latent space, and the cosine similarity between entity and domain is used to select entities above a threshold. Specifically, the domain vectors are computed by a weighted average of the $m = 50$ words and entities that are most related to the Wikipedia articles on "book", "travel" and "food", respectively, with weights proportional to cosine between vectors. For selective entity expansion of per-domain user models, we pick entities whose similarity to the respective domain model is above a specified threshold.

This approach introduces several thresholds and hyper-parameters: per-domain numbers of related terms for the domain model and similarity thresholds for pruning entities. We tuned these via grid search with the objective of maximizing the area under the precision-recall curves for entity detection and disambiguation. We used the manually annotated entities in the domain-specific questionnaires as ground-truth for domain relatedness.

5 Data Collection

We gathered personal data in a 4-week user study with 14 students who were paid ca. 10 Euros per hour. We randomly paired two users for 3 chats per week. For the first week, users were instructed to chat generally, like mutual introductions. During the remaining weeks users were asked to chat about specific topical domains: users' interests and tastes in books and their experience and interests in traveling and food. On average, each user had 2.8 sessions for each domain, totaling to ca. 11 sessions overall, with an average of 653 utterances and 3934 tokens per user. In addition, each user filled in several questionnaires upfront:

a general one with 18 questions about demographics, general interests and personality, and one for each of the themes books, travel and food with 2, 5 and 10 questions, respectively (see left side of Fig. 1 for an example excerpt). The general questionnaire included personality-oriented questions such as "What are your hobbies?", "What makes you happy?", and "Your golden rule?".

6 Experimental Studies

6.1 Setup

The 14 users from whom we collected questionnaire and chat data also participated in an assessment study of personalized search results. To this end, we compiled 75 medium-grained keyword queries (25 per domain). Example queries are shown in Table 1.

All queries were issued to a commercial search engine with site restrictions as described in document models (Sect. 2). The top-100 answers were retrieved, keeping only those that were about specific entities and discarding general list pages – this left us with 90 or more answers for each query.

The users were asked to identify around 5 queries for each domain on topics that looked potentially appealing to them. This way we avoided personalized judgements on topics that the user does not care about. For each query, a user assessed 20 results that were sampled uniformly at random (to avoid ranking bias) and, additionally, the top-10 results from the original ranking (with the risk of bias). We asked for subjective, graded assessments with labels: 2 = strongly interested, 1 = mildly interested, 0 = uninterested, and discarded all "I don't know" assessments. We required the users to enter justification sentences along with their judgements. In total, we obtained 2673 individual assessments for 113 user-query pairs with 73 distinct queries.

Evaluation Metrics: The primary metric is **NDCG@20**, which we use to refer to methods' effectiveness when re-ranking the 20 randomly sampled query results. In addition, we report on **precision@1** where we compare the highest-ranked results from the 20 random samples against a user judgement of 1 or 2 (= strongly or mildly interested). For completeness, we also consider **NDCG@top10** for the top-10 results of the original, potentially biased, rankings from a commercial search engine.

Table 1. Example queries by domain

Books	Scandinavian suspense	Novels made into movies	Personal development
Travel	Weekend trip for festival	Best wine lover destination	Epic road trip
Food	Perfect breakfast	Iron rich vegetarian recipes	15-minute meal recipes

Methods Under Comparison: We cover the following methods and configurations.

- **LM** denotes the language model approach. To isolate the effect of the user model in the re-ranking, and as our initial pool of entities are to some extent relevant to the query, we either set the λ to 0 or 1. When $\lambda = 1$ the input to the re-ranker is the query model and when $\lambda = 0$ only the user model is given as input.
- **LM-embed** is the language-model method with word embeddings using word2vec. The term-term similarity threshold is set to 0.5.
- **BM25** is the BM25 method with parameters set to the following values widely used in the literature: $b = 0.75, k_1 = 1.5$.
- **KNRM** is the neural ranker, with the maximum query and document lengths set to 50 and 5000, respectively. The terms for the query/user model are obtained by tf order, selecting the top 50 distinct terms. Document terms are the top-5000 terms. Models are trained on data per domain with 504, 772 and 806 assessments for book, food and travel, respectively.
  As this training is fairly low-end, we also study a variant **KNRM-all** where we combine all domains into a single training set with 2082 labeled samples. We report on ten-fold cross-validation with 8, 1 and 1 folds for training, validation and test, respectively.
- **SE** is the initial ranking from a commercial search engine.

6.2 User Models: None vs. Chats vs. Questionnaires (RQ0 and RQ1)

Table 2 shows the NDCG@20 results for the influence of different user models. The top part of Table 2 gives the overall results across all domains (averaged over the 113 user-query pairs). The other parts show per-domain results. The user models under comparison here are query-only vs. questionnaires-based vs. chats-based. For the latter two, we varied the specific setting by deriving models from all available inputs regardless of the domains (*All*), using only general questionnaires or chats (*Gen*, see Sect. 5), using only domain-specific inputs (*Dom*), or using both general and per-domain inputs (*Dom* + *Gen*). In this comparison, all methods were configured without entity expansion and without domain-specific vocabularies (which will be discussed in the next subsections).

Overall Results (Top Part of Table 2): The overriding observation is that almost all rankers with different degrees of personalization improve over the SE baseline and that both questionnaire-based and chat-based user models achieve notable gains over the query-only rankers: in the order of 2 to 4% points in NDCG@20. While the effect size of personalization is only moderate, the relative gains are statistically significant and come at little cost for the ranker efficiency. For significance, two-tailed paired t-tests in comparison to the Query-Only baselines mostly had p-values < 0.05. These results are marked with an asterisk in Table 2.

Table 2. NDCG@20 for different rankers and user models. Best results per row are in boldface. Statistically significant improvements over the Query-Only baselines are marked with an asterisk.

Ranker	User model								
	Query only	Questionnaires				Chats			
		All	Gen	Dom	Dom+Gen	All	Gen	Dom	Dom+Gen
LM	0.796	0.816	0.804	0.823*	**0.824***	0.811	0.806	0.822*	0.817*
LM-embed	0.794	0.791	0.787	**0.811**	0.798	0.782	0.777	0.795	0.784
BM25	0.785	0.823*	0.815*	0.827*	**0.833***	0.819*	0.816*	0.827*	0.821*
KNRM	**0.807**	0.791	0.805	0.798	0.794	0.780	0.786	0.784	0.785
KNRM-all	**0.810**	0.807	0.796	–	–	0.788	0.791	–	–
SE	0.786	–	–	–	–	–	–	–	
Books									
LM	0.825	0.829	0.823	0.822	0.834	0.846	**0.854**	0.844	0.847
LM-embed	**0.818**	0.795	0.803	0.801	0.799	0.811	0.799	0.813	0.806
BM25	0.814	0.843	0.846	0.834	0.847	0.846	0.849	**0.851**	0.850
KNRM	0.826	0.827	**0.832**	0.817	0.816	0.790	0.810	0.790	0.809
SE	0.777	–	–	–	–	–	–	–	
Travel									
LM	0.818	0.821	0.815	**0.854***	0.838	0.826	0.813	0.841	0.835
LM-embed	0.813	0.799	0.787	**0.849***	0.814	0.785	0.782*	0.803	0.796
BM25	0.794	0.837*	0.833*	**0.857***	0.849*	0.836*	0.837*	0.844*	0.838*
KNRM	**0.838**	0.806	0.833	0.827	0.801	0.800	0.800	0.805	0.800
SE	0.794	–	–	–	–	–	–	–	
Food									
LM	0.753	0.802*	0.779	0.790	**0.803***	0.772	0.766	0.785	0.777
LM-embed	0.757	0.778	0.775	0.777	**0.780**	0.758	0.755	0.773	0.757
BM25	0.756	0.793	0.775	0.791	**0.806***	0.783	0.770	0.793*	0.782
KNRM	0.761	0.751	0.756	0.755	**0.771**	0.752	0.753	0.757	0.753
SE	0.785	–	–	–	–	–	–	–	

Interestingly, LM-embed did not improve over LM. The term-term relatedness by word2vec seems to be too crude for our task and dilutes the query focus. KNRM and KNRM-all were inferior to the Query-Only case. The combination of small training data and limited input size is the likely cause for this disappointing result.

When comparing questionnaire-based vs. chat-based personalization, the former performs slightly better than the latter, but the differences are minor. For both, the best variants were the ones with user models *Dom* or *Dom+Gen*, indicating awareness of the domain is beneficial. *Dom* is almost always preferable to *Dom+Gen* in the case of chats, but there is no clear trend when using questionnaires. This is likely due to the fact that the general questionnaires were designed to reveal user personalities, whereas the general chats were mostly introductory and less informative. These gains are not always statistically significant, but the best cases are: for example, the improvement for LM from 0.811 with chats-*All* to 0.822 with chats-*Dom* had a p-value of 0.0018.

Per-Domain Results: The results vary among the different domains in an interesting way. We base the discussion on LM and BM25 as they achieved the best results. For books and travel, the gains from user models are most pronounced. For books, the chat-based models achieved a small but notable and significant improvement over the questionnaire-based ones. We observe that for questionnaire-based models $Dom + Gen$ outperforms Dom. This is due to the low coverage of the book domain with only two questions on the user's favorite books and genres, whereas the general questionnaire includes demographics and personal traits. On the other hand, for the travel domain with 5 specific questions, Dom performs better than $Dom + Gen$ in both questionnaire-based and chat-based models, with the former giving the best results.

For food, personalization led to gains, but the absolute NDCG scores were substantially lower than for the other two domains. Here, the SE performed better than the re-rankers with the query-only model. However, using $Dom{+}Gen$ questionnaire-based profiles, we achieved up to 2% improvement over the SE results. It seems that the food domain is inherently difficult to understand, as its vocabulary mixes specific and very common words with a strong influence of the latter on tastes and sentiments (e.g., "hot", "terrific" etc.).

As for precision@1, the overall gains by personalization were nearly 10%: considering the best-performing rankers on overall results, the LM improved from 70% with query-only models to 81% with questionnaire-based models, and BM25 went up from 66% to 83%. Again, the gains were most substantial for books and travel, but here food as well showed notably improved precision@1. We further evaluated NDCG@top10: not surprisingly, the SE baseline was stronger for this metric, but was still outperformed by re-ranking with personalization. The best values for our method were comparable to those for NDCG@20, around 83% across all domains and up to 87% for travel.

6.3 Domain Vocabularies (RQ2)

Recall from Sect. 3 that we optionally incorporate domain-specific term weighting to reduce the influence of irrelevant wording from the user chats. Table 3 shows NDCG@20 results with this awareness of domain vocabularies, for the four chat-based configurations All, Gen, Dom and $Dom + Gen$. We show only overall results across all domains, but for each domain, all user-model terms were weighted by the respective $spy(w)$ domain model. For brevity, we restrict ourselves to the LM-based ranker; the findings were similar for the other two rankers.

Table 3 indicates that there are small gains from this domain-specific weighting, but the effect size is marginal and not statistically significant (p-value > 0.1). It seems that chats are not sufficiently focused on domain-specific topics. Humans do jump between topics, so chats naturally have a high level of thematic diversity.

Table 3. NDCG@20 for LM-based ranker with domain-specific vocabularies

Domain specificity	Chats			
	All	Gen	Dom	Dom+Gen
Disabled	0.811	0.806	0.822	0.817
Enabled	0.821	0.813	0.83	0.826

Table 4. NDCG@20 for LM-based ranker across all domains with entity expansion. Best results per column are in boldface. Statistically significant improvements over no entity awareness None baselines are marked with an asterisk.

Entity expansion	User models							
	Questionnaires				Chats			
	All	Gen	Dom	Dom+Gen	All	Gen	Dom	Dom+Gen
None	0.816	0.804	0.823	0.824	0.811	0.806	0.822	0.817
All	0.817	**0.816**	0.826	0.822	**0.821**	0.814	0.828	0.824
Domain	0.823	0.812	**0.827**	0.824	**0.821***	0.814*	0.829	0.824
NE-all	0.823	0.81	0.826	0.829	0.818*	0.813	0.829	**0.825***
NE-dom	**0.829***	0.809	0.825	**0.833**	0.819*	**0.815***	**0.83**	0.824*

6.4 Entity Expansion (RQ3)

To study the influence of entity expansion for the user models, we compared different settings against the previously reported configurations without entity awareness: *all* expands all entities including concepts (in Wikipedia, such as "history" or "Buddhist art"); *domain* restricts the entities to those that are related to the respective domain (see Sect. 4); *NE-all* uses only named entities (i.e., discarding general concepts); *NE-dom* uses only named entities with domain relatedness above a threshold.

Table 4 shows the overall NDCG@20 for these settings with the different configurations for the user-model construction. We observe that almost none of the expansion methods significantly improve the models derived from questionnaires. The reason is that these models are already very concise given their high-quality inputs. For chat-based user models, on the other hand, entity expansion led to small, but notable and statistically significant (p-values < 0.05), improvements of ca. 1%.

7 Related Work

Recommender systems are ubiquitous in search, e-commerce and social content sharing. Most state-of-the-art systems learn from massive amounts of user-behavior signals: queries, clicks, likes, ratings, etc. (e.g., [19,25,52]). To a lesser extent, product reviews are considered as well (see, e.g., [7,14]), but recent studies [34,40] indicate that there is considerable noise in user reviews and limited benefit from such additional input. In the opposite direction, [6] made the point that user models for personalized recommendations should be scrutable and, therefore, use as little information as possible and make the derived models transparent and user-interpretable. The work [43] pursued this rationale by building on explicit user profiles from short questionnaires. The current paper's experiments include comparisons to that approach. None of the prior works has considered user-user chats as a source for capturing user interests and tastes. Note that interactive and conversational recommenders [26,36] are a very different approach, as they build on dialogs between user and system, not among users.

Specialized recommenders that tap textual contents have been investigated for domains like e-learning, literature exploration or tourism (e.g., [3,20,27,30]). These are based on rich context models of user history and interests. However, they are not query-based, disregarding the additional component of search results on behalf of the user.

Personalized ranking of search results has been addressed from two angles (see [17] for a survey): i) building user models from user queries and browsing histories (e.g., [1,15,23,37,41]), and ii) exploiting such models for ranking, query expansion or auto-completion (e.g., [12,29,38]). For the first task, the seminal work of [41] analyzed user activities reflected in queries, clicks and emails, all the way to news and other contents read by a user. For personalized ranking, language models were enhanced with user-specific priors [39]. The interplay of long-term behavior and short-term sessions of a user was studied by [8,10]. Other work [9,42] investigated the issue of when to personalize and when to disregard user profiles. None of these prior works is specifically geared for entity search, and none considers user models derived from chats.

Entity search (e.g., [4,16]) has been studied for personalization only in limited settings. The CLEF competition on book recommendations [21] relied on extensive data (posts, tags, reviews, ratings) by large user communities at LibraryThing and Amazon. Most related to our work is [2] on personalized product search, based on embeddings for users and products in a joint latent space. That method exploited user reviews on product pages. In contrast, our approach is based on user-user chats, an unintrusively observable asset disregarded in prior works.

Query expansion is a well established methodology in IR (see, e.g., [13] for a survey). Personalization has been studied in this context along various routes. Notable examples are [11,53] based on user-provided tags, and [23] based on email histories and utilizing word embeddings learned from email contents. Recently, [49] has pursued the theme of personalized word embeddings further, based on query histories.

8 Conclusion

To the best of our knowledge, this is the first work that explores leveraging *user-to-user* conversations as a source for personalization of search-based recommendations. We compared chat-based user models against models derived from concise questionnaires. Both achieved substantial improvements over both the original search-engine ranking and non-personalized query-only re-rankings.

Between chat-based and questionnaire-based re-rankings, there is no clear winner. The two paradigms of user models each have specific benefits:

- Questionnaries are transparent and scrutable for users. However, they require an explicit effort. Most users seem fine with a one-time questionnaire, but few seem ready for periodic updating as their interests and tastes evolve.
- Chats, on the other hand, require no effort at all from the user side, and could be easily updated without user intervention. However, the derived models are less transparent to humans and not easily adjustable by users themselves. Also, chat data comes with higher privacy risks.

The additional enhancements devised in this paper – domain awareness and entity expansion – further improved the NDCG scores, but only to a small extent. On the other hand, focusing on entities in conversations and casting them into an explicit user model is a step towards making chat-based profiles more transparent and scrutable for users.

References

1. Agichtein, E., Brill, E., Dumais, S., Ragno, R.: Learning user interaction models for predicting web search result preferences. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2006)
2. Ai, Q., Zhang, Y., Bi, K., Chen, X., Croft, W.B.: Learning a hierarchical embedding model for personalized product search. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2017. ACM (2017)
3. Bai, X., Wang, M., Lee, I., Yang, Z., Kong, X., Xia, F.: Scientific paper recommendation: a survey. IEEE Access **7**, 9324–9339 (2019)
4. Balog, K.: Entity-Oriented Search. INRE, vol. 39. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93935-3
5. Balog, K., Kenter, T.: Personal knowledge graphs: a research agenda. In: Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR (2019)
6. Balog, K., Radlinski, F., Arakelyan, S.: Transparent, scrutable and explainable user models for personalized recommendation. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR (2019)
7. Bauman, K., Liu, B., Tuzhilin, A.: Aspect based recommendations: recommending items with the most valuable aspects based on user reviews. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2017)
8. Bennett, P.N., Radlinski, F., White, R.W., Yilmaz, E.: Inferring and using location metadata to personalize web search. In: Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (2011)

9. Bennett, P.N., Shokouhi, M., Caruana, R.: Implicit preference labels for learning highly selective personalized rankers. In: Proceedings of the 2015 International Conference on the Theory of Information Retrieval, ICTIR (2015)
10. Bennett, P.N., et al.: Modeling the impact of short- and long-term behavior on search personalization. In: The 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (2012)
11. Biancalana, C., Gasparetti, F., Micarelli, A., Sansonetti, G.: Social semantic query expansion. ACM TIST **4** (2013)
12. Cai, F., de Rijke, M.: Selectively personalizing query auto-completion. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (2016)
13. Carpineto, C., Romano, G.: A survey of automatic query expansion in information retrieval. ACM Comput. Surv. **44** (2012)
14. Chen, L., Chen, G., Wang, F.: Recommender systems based on user reviews: the state of the art. User Model. User-Adapted Interact. **25**(2), 99–154 (2015). https://doi.org/10.1007/s11257-015-9155-5
15. Chirita, P., Firan, C.S., Nejdl, W.: Personalized query expansion for the web. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2007)
16. Dietz, L.: ENT rank: retrieving entities for topical information needs through entity-neighbor-text relations. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (2019)
17. Ghorab, M.R., Zhou, D., O'Connor, A., Wade, V.: Personalised information retrieval: survey and classification. User Model. User-Adapted Interact. **23**, 381–443 (2013). https://doi.org/10.1007/s11257-012-9124-1
18. Guo, Q., et al.: A survey on knowledge graph-based recommender systems. CoRR (2020)
19. Jiang, J., et al.: End-to-end deep attentive personalized item retrieval for online content-sharing platforms. In: WWW 2020: The Web Conference 2020. ACM/IW3C2 (2020)
20. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z., Jain, L.C.: E-Learning Systems. ISRL, vol. 112. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-41163-7
21. Koolen, M., et al.: Overview of the CLEF 2016 social book search lab. In: Fuhr, N., et al. (eds.) CLEF 2016. LNCS, vol. 9822, pp. 351–370. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44564-9_29
22. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**, 30–37 (2009)
23. Kuzi, S., Carmel, D., Libov, A., Raviv, A.: Query expansion for email search. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2017)
24. Kuzi, S., Shtok, A., Kurland, O.: Query expansion using word embeddings. In: Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM (2016)
25. Lalmas, M.: Personalizing the listening experience (invited talk), slides at https://prs2019.splashthat.com/
26. Lei, W., He, X., de Rijke, M., Chua, T.: Conversational recommendation: formulation, methods, and evaluation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2020)

27. Li, X., Chen, Y., Pettit, B., de Rijke, M.: Personalised reranking of paper recommendations using paper content and user behavior. ACM Trans. Inf. Syst. **37** (2019)
28. Liu, J., Dolan, P., Pedersen, E.R.: Personalized news recommendation based on click behavior. In: Proceedings of the 15th International Conference on Intelligent User Interfaces, IUI. ACM (2010)
29. Matthijs, N., Radlinski, F.: Personalizing web search using long term browsing history. In: Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM (2011)
30. Menk, A., Sebastia, L., Ferreira, R.: Recommendation systems for tourism based on social networks: a survey. CoRR (2019)
31. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems NIPS (2013)
32. Ricci, F., Rokach, L., Shapira, B. (eds.): Recommender Systems Handbook. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6
33. Robertson, S.E., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. Found. Trends Inf. Retrievel **3**, 333–389 (2009)
34. Sachdeva, N., McAuley, J.: How useful are reviews for recommendation? A critical review and potential improvements. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020. Association for Computing Machinery (2020)
35. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web (2001)
36. Schnabel, T., Amershi, S., Bennett, P.N., Bailey, P., Joachims, T.: The impact of more transparent interfaces on behavior in personalized recommendation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2020)
37. Shen, X., Tan, B., Zhai, C.: Implicit user modeling for personalized search. In: Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management (2005)
38. Shokouhi, M.: Learning to personalize query auto-completion. In: The 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (2013)
39. Sontag, D., Collins-Thompson, K., Bennett, P.N., White, R.W., Dumais, S.T., Billerbeck, B.: Probabilistic models for personalizing web search. In: Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM (2012)
40. Stratigi, M., Li, X., Stefanidis, K., Zhang, Z.: Ratings vs. reviews in recommender systems: a case study on the amazon movies dataset. ADBIS 2019. CCIS, vol. 1064, pp. 68–76. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30278-8_9
41. Teevan, J., Dumais, S.T., Horvitz, E.: Personalizing search via automated analysis of interests and activities (SIGIR test-of-time award 2017). In: Proceedings of the 28th Annual International ACM SIGIR (2005)
42. Teevan, J., Dumais, S.T., Liebling, D.J.: To personalize or not to personalize: modeling queries with variation in user intent. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2008)

43. Torbati, G.H., Yates, A., Weikum, G.: Personalized entity search by sparse and scrutable user profiles. In: Proceedings of the 2020 Conference on Human Information Interaction and Retrieval, CHIIR 2020. Association for Computing Machinery (2020)
44. Wu, F., et al.: MIND: a large-scale dataset for news recommendation. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL. Association for Computational Linguistics (2020)
45. Wu, L., Grbovic, M.: How Airbnb tells you will enjoy sunset sailing in Barcelona? Recommendation in a two-sided travel marketplace. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2020)
46. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2017)
47. Yamada, I., Asai, A., Shindo, H., Takeda, H., Takefuji, Y.: Wikipedia2Vec: an optimized tool for learning embeddings of words and entities from Wikipedia. CoRR (2018)
48. Yamada, I., Shindo, H., Takeda, H., Takefuji, Y.: Joint learning of the embedding of words and entities for named entity disambiguation. In: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL. ACL (2016)
49. Yao, J., Dou, Z., Wen, J.: Employing personal word embeddings for personalized search. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2020)
50. Zhai, C.: Statistical language models for information retrieval: a critical review. Found. Trends Inf. Retrieval (2008)
51. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**, 1–38 (2019)
52. Zhao, Z., et al.: Recommending what video to watch next: a multitask ranking system. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys. ACM (2019)
53. Zhou, D., Wu, X., Zhao, W., Lawless, S., Liu, J.: Query expansion with enriched user profiles for personalized search utilizing folksonomy data. IEEE Trans. Knowl. Data Eng. **29**, 1536–1548 (2017)

Joint Autoregressive and Graph Models for Software and Developer Social Networks

Rima Hazra[1](✉), Hardik Aggarwal[1], Pawan Goyal[1], Animesh Mukherjee[1](✉), and Soumen Chakrabarti[2]

[1] IIT Kharagpur, Kharagpur, India
to_rima@iitkgp.ac.in, {pawang,animeshm}@cse.iitkgp.ac.in
[2] IIT Bombay, Mumbai, India
soumen@cse.iitb.ac.in

Abstract. Social network research has focused on hyperlink graphs, bibliographic citations, friend/follow patterns, influence spread, etc. Large software repositories also form a highly valuable networked artifact, usually in the form of a collection of packages, their developers, dependencies among them, and bug reports. This "social network of code" is rarely studied by social network researchers. We introduce two new problems in this setting. These problems are well-motivated in the software engineering community but not closely studied by social network scientists. The first is to identify packages that are most likely to be troubled by bugs in the immediate future, thereby demanding the greatest attention. The second is to recommend developers to packages for the next development cycle. Simple autoregression can be applied to historical data for both problems, but we propose a novel method to integrate network-derived features and demonstrate that our method brings additional benefits. Apart from formalizing these problems and proposing new baseline approaches, we prepare and contribute a substantial dataset connecting multiple attributes built from the long-term history of 20 releases of Ubuntu, growing to over 25,000 packages with their dependency links, maintained by over 3,800 developers, with over 280k bug reports.

Keywords: Ubuntu packages · Software dependency network · Bug urgency prediction · Developer recommendation

1 Introduction

A rapidly growing, rich, complex and immensely valuable social network has garnered surprisingly little attention compared to the WWW hyperlink graph, follower-followee and retweet/repost/reply networks in social media platforms etc. This network is formed by software packages, the dependency graph that links them, their developers, and bug reports and discussions concerning them.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 224–237, 2021.
https://doi.org/10.1007/978-3-030-72113-8_15

As a case in point, Linux, with its many flavors and adaptations, is a huge public software repository. It has tens of thousands of packages, connected by dependency and other links. Thousands of developers contribute to these packages, forming another aspect of the network. In fact, the developer ecosystem evolves organically, rather than via central command-and-control chains. The network is highly dynamic, with accurately-maintained trace of evolution along with detailed logs of bug reports pertaining to different packages. Business realities have made open-source software development viable even for commercial organizations, with notable examples like Tensorflow, ZFS, Ubuntu, Java, Postgres, etc. A comparatively nascent and chaotic version of such self-organization of software networks can be found on github, gitlab and bitbucket.

In this work we focus on the Ubuntu code repository. Ubuntu, a Linux based distribution is a collection of many open source software/packages. The project encourages the community to contribute to the development and maintenance of one or more packages. For every package, there is a set of developers (often one) who are responsible for the maintenance of the package and keep track of all the changes to the package in a changelog[1], recording the sequence of bug fixes or other updates related to the package.

Unlike traditional social network tasks of centrality/prestige computation, influence or cascade prediction, the social network of software comes with novel tasks having strong motivation and relevance in the software management community.

Bug Urgency Ranking: The task is to rank packages that are likely to be most afflicted by bugs in the immediate future. Since there is no central command, the developer community has to autonomously discover the trouble spots.

Developer Recommendation: For each package, the task is to propose the developers best suited to contribute in the immediate future. Compared to software corporations with top-down management, the developer community shows high levels of churn, making such prediction difficult. We know of no widely used public domain tools for predicting bug urgency or recommending developers for a given package. While there are several articles on developer/commenter recommendation [11] in various community question answering sites, to our knowledge, none of them attempt to build a model to recommend the developers in software development platforms like Ubuntu.

Our Contributions and Results

A New Dataset: We contribute a substantial new dataset[2] connecting multiple software and developer artifacts built from the long-term history of 20 releases of Ubuntu, growing to over 25k packages with their dependency links, maintained by over 3800 developers, with over 280k bug reports. There are 25k unique nodes (packages) and 120k dependency links among these packages across the Ubuntu releases.

[1] http://changelogs.ubuntu.com/changelogs/.

[2] http://doi.org/10.5281/zenodo.4092623.

Algorithms for Bug Urgency Ranking: We propose autoregressive baselines that predict future bug urgency as a regression based on recent history, then augment them with novel ways to incorporate inter-package dependency graph signals, which result in enhanced ranking accuracy. We are able to achieve high rank correlation between gold and system rankings, for both the autoregressive and autoregressive+dependency models. For the most recent distribution (i.e., Zesty) in our data set, Spearman's rank correlation ρ@25 and Kendall's τ@25 values are respectively 0.582 and 0.451 using only the autoregressive features. Inclusion of dependency features further improves both the correlation values ($\rho@25 = 0.60$ and $\tau@25 = 0.466$). If one considers the full rank list then we obtain $\rho = 0.35$, $\tau = 0.33$ for the autoregressive case and $\rho = 0.38$, $\tau = 0.35$ in case of autoregressive+dependency. For the full rank list the differences in the results between the autoregressive and autoregressive+dependency schemes are statistically significant ($p < 0.01$ for both ρ and τ as per Mann-Whitney U test [6]).

Algorithms for Developer Recommendation: In its most basic form, recommending developers for a package may be modeled as predicting a set given a sequence of past sets [2]. However, our data set has richer signals in both space (i.e., graph structure) and time, as well as features from bug reports, bug fix changelogs, etc. Even a simple autoregressive approach is able to take advantage of these features and outperform baselines. For the most recent distribution, the Mean Reciprocal Rank (MRR) for the autoregressive approach is $\sim$0.788 as compared to 0.772 for the best performing baseline. Additional benefits are also obtained from the dependency relations (MRR $\sim$0.793). Subject to some reasonable assumptions, we also compute upper bounds for autorgressive and autoregressive+dependency schemes as 0.8096 and 0.8445 respectively, which gives ample scope of improvement in future.

2 Related Work

Recommendation systems are nowadays becoming available to assist developers in various activities—from reusing code [4] to writing effective bug reports [1,8].

Developer Recommendation Approaches: We witness a growing volume of literature on developer recommendation for crowdsourced tasks. Mao et al. [7] employed content-based recommendation techniques to automatically match tasks and developers for the TopCoder platform. Related work [12] recorded a task-quitting rate of 82.9% among TopCoder developers. Ye et al. [13] proposed four problems that limit the effectiveness of existing methods at recommending suitable developers. Tunio et al. [10] studied the impact of personality on task selection in crowdsourcing software development.

Package Dependency Networks: De Sausa et al. [9] presented an analysis of the package dependency on Debian GNU/Linux. Kikas et al. [5] studied the structure and evolution of package dependency networks of JavaScript, Ruby, and Rust ecosystems. Decan et al. [3] showed that experimental results related to software packages belonging to a single software ecosystem fail to generalise to other ecosystems because of the diversity of their structure.

We know of no widely used approach that uses package dependency networks for developer recommendation. Also, we find limited research on Dirichlet based sampling approach in recommendation and ranking. In this paper, we combine the two paradigms for the two tasks that we solve—bug urgency prediction and developer recommendation.

3 Dataset

Ubuntu is a free and open-source Linux distribution based on Debian, released for Desktop, Server and IoT deployment[3]. It is released every six months, with long-term support (LTS) releases every two years. Our data consists of three parts: (i) Ubuntu packages and the dependencies among these packages, (ii) developers of Ubuntu packages responsible for bug fixes and other updates and the maintenance of change logs, and (iii) bug(s) associated with each package. For our experiments we only use 20 non-LTS versions (binary-amd64) published between April 2004 and April 2017. Dataset details follow in this section.

Ubuntu Packages and Their Dependencies: Each Ubuntu distribution contains a collection of binary packages. Binary packages are made for different types of architectures like AMD64, i386 etc. For each distribution, we collected binary packages and their dependencies. The most prevalent form of dependency between a pair of binary packages is referred to as *depends*[4]. A binary package P_i *depends* on another binary package P_j if P_j is required to build and install P_i. "Dependee" denotes a binary package (P_j) on which another binary package (P_i) depends. In the rest of this paper the dependency network that we refer to is built from this *depends* relation.

A source package, on building, may generate a set of binary packages[5]. E.g., the source package "0ad"[6] contains the binary packages, "0ad" and "0ad-dbg". We consider source packages and their dependencies in our experiments. We chose source packages instead of binaries since the source packages have a unique identify with source codes unlike binary packages which may correspond to compiled codes from different architectures. We present all our results for the three most recent distributions – 'Wily', 'Yakkety' and 'Zesty' which have 22799, 24609 and 25648 source packages respectively. An example dependency graph is shown in Fig. 1. The package 'systemd' and its dependees 'libseccomp', 'glibc' and 'iptables' are shown in gray in the figure.

The number of "depends" package dependencies across the Ubuntu distributions are reported in Fig. 2(a) (increases as time progresses).

[3] https://www.ubuntu.com/#download.

[4] There are other types of relations also in the dataset like *recommends*, *suggests* and *conflicts* which are very infrequent.

[5] https://askubuntu.com/questions/357295/what-is-difference-between-binary-and-source-file.

[6] https://packages.ubuntu.com/source/xenial/0ad.

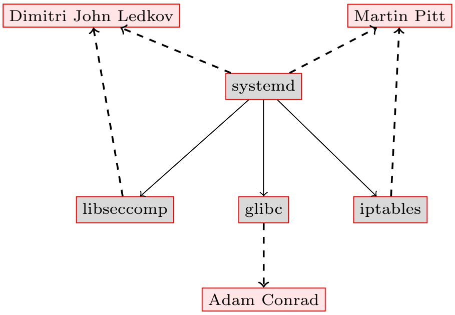


Fig. 1. Dependees and developers of the ‘systemd’ package. Gray nodes: software packages, Red nodes: developers. Solid and dotted lines represent *‘source dependency’* and *‘contributed by’* relationships, respectively. ‘Dimitri John Ledkov’ is a developer associated with both ‘systemd’ and its dependee ‘libseccomp’.

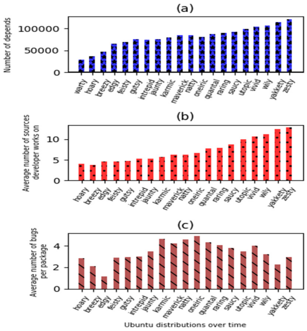


Fig. 2. (a) The no. of “depends” dependencies in each Ubuntu distribution. (b) The average no. of source packages the developers worked on across the distributions. (c) The average no. of bugs per source package (with non-zero bugs).

Developers of Ubuntu Packages: The ‘changelog’[7] of a source package contains add/remove/update information about the codebase and bugs (resolved bugs) associated with the package. It also contains the urgency level, name of the developer and timestamp of that *change*. Packages evolve at diverse paces, and distributions take snapshots at discrete points in time. We have collected changelogs of source packages and mapped them to Ubuntu distributions. These change logs allow us to associate every developer with one or more packages for each distribution. An example of the developer-package relation is illustrated in Fig. 1. In our dataset, we observed that a particular source typically (but not always) has a single developer in each distribution. Over time the average number of packages a developer contributes to is reported in Fig. 2(b) (shows an increasing trend).

Bugs Associated with Ubuntu Packages: Ubuntu releases do not provide a straightforward way to recover the bugs (along with their meta data) that are associated with a particular distribution. We therefore collected 280k bugs along with all available information from Launchpad.[8] We next associated a bug with a particular distribution if that bug had been created within six months from the release of that distribution. Next we mapped these bugs associated with

[7] E.g., http://changelogs.ubuntu.com/changelogs/pool/universe/0/0ad/0ad_0.0.20-1/changelog.

[8] https://launchpad.net/.

a distribution to the corresponding source packages. The number of bugs per package, averaged over all packages and all distributions, is ~ 3.4 (considering all packages that have non-zero reported bugs). The average number of bugs per package over time is reported in Fig. 2(c). The plot shows that in early versions of Ubuntu, fewer bugs were reported, followed by a sharp increase and then a final decline. This possibly indicates that as the software became more complex and popular, the number of bugs reported grew quickly. However, it settled down at later time point due to the consolidated rectification efforts made by developers.

4 Notation and Preliminaries

We have a set of T **distributions** of a large software system (such as Ubuntu) indexed as $t \in [T] = \{1, \ldots, T\}$, where t represents discrete, ordinally comparable time and equivalently, distributions. An example is vivid < wily where vivid and wily are the Ubuntu distributions. There is a universe of S **packages** indexed by $s \in [S] = \{1, \ldots, S\}$. An example is $s =$ glibc. (By 'packages', we will mean "source packages" in the context of Ubuntu. A source package may build to multiple binary packages, but developers are naturally assigned to source, not binary packages.) Not all packages s may be present in all distributions t. Packages can be removed and later restored. For each package s, there is a package size $\mathrm{ps}(s,t)$ which is the sum of the sizes of its binaries at distribution t. There is a universe of D **developers**, indexed as $d \in [D] = \{1, \ldots, D\}$. An example is $d =$ torvalds@linux.org. A developer may contribute to many packages at various time steps (distributions). Let $\mathrm{devs}(s,t) \subset [D]$ denote the set of developers associated with package s at time t. There is a universe of B bug records, indexed as $b \in [B] = \{1, \ldots, B\}$. One bug record attaches to a single package at a single distribution. Let $\mathrm{bugs}(s,t) \subset [B]$ denote the set of bugs associated with package s at time t. We denote the heterogeneous graph constructed at each time step t as G_t. G_t comprises two types of nodes—source packages s and developers d. Edges $s \to s'$ represent *'source dependency'* relationships. The term "dependent" corresponds to the source package (s) that depends on another source package. "Dependee" represents the source package (s') on which the dependent depends. Edges $s \to d$ represent *'contributed by'* relationships. Figure 1 shows an illustrative graph fragment at $t =$ zesty for a source package 'systemd'. Note that a developer can work on multiple packages at the same time. For example, 'Dimitri John Ledkov' is developer of both 'systemd' and its dependee package 'libseccomp'. Let the set of in-neighbors and out-neighbors of the target source package s at time point t be $S_{\mathrm{IN},t}$ and $S_{\mathrm{OUT},t}$ respectively.

5 Bug Urgency Ranking

Suppose we observe the evolution of the software ecosystem from time step 1 through $t-1$. In other words, we observe G_τ for $\tau \in [1, t-1]$ along with developer and bug sets associated with each package and time step. Now, for time step t, our goal is to predict $|\mathrm{bugs}(s,t)|$ for all packages s. More practically, we want to sort packages s in decreasing order of $|\text{bugs}(s,t)|$ and report the top ranks to attract the attention of central members of the developer community, so that they can solicit and allocate more programmer resources. With that motive in mind, we are generally interested in predicting only the *relative bug density* among packages in the next release. While evaluating, we naturally have access to the gold $\text{bugs}(s,t)$, and we can therefore compare the system and gold rankings using various rank correlation measures.

Pure Autoregressive Approach: In this approach, we attempt to estimate the rank of source packages based on the bugs reported at earlier time points. From the bugs information, we computed the number of bugs ($|\text{bugs}(s,t)|$) of each source package (s) for each time point t. Specifically, we extract autoregressive features from earlier time points and predict the $|\text{bugs}(s,t)|$ for the current time point. For each source package, we consider two autoregressive features from the previous two time points, i.e., $|\text{bugs}(s,t-1)|$, $|\text{bugs}(s,t-2)|$, $\text{ps}(s,t-1)$ and $\text{ps}(s,t)$.

We also tried to use the same features from even earlier time points. However, their contribution to the overall prediction performance is negligible compared to the last two time points and hence they are ignored. We observe that the bug history of two previous time points always contributes more than the package sizes in the prediction. Our intuition behind utilizing package size is that, if the package size changes from the last time point to the current time point, then the package should contain some new updates. For example, for the package "systemd" in the "Zesty" distribution, the package size and the number of bugs are 6.12 MB and 21 respectively. In the previous distribution "Yakkety", the package size and number of bugs were 4.43 MB and 15 respectively. This and other similar observations made us hypothesize that the package size at time point t might have potential correlation with the number of bugs at t.

Inclusion of Network Features: We hypothesize that the bugs in a particular source package could potentially induce bugs in its dependees as well as dependents. For instance, in distribution "Zesty", the "systemd" package has 21 bugs whereas in the immediate previous distribution ("yakkety") this number is 15. The observed rise may be attributed to the very large number of bugs (60) associated with one of the in-neighbours ("linux") of "systemd" in the previous distribution "Yakkety". Overall, across our full dataset, the Pearson's correlation between the bugs of a source package at time point t and the bugs of its in-neighbors/out-neighbors at the previous time point $t-1$ lies between approximately 0.18 and 0.28. This makes us further confident that positive benefits could be obtained by considering the previous time point bugs of in-neighbours and out-neighbours as additional features.

Therefore, along with the autoregressive features, we also use the dependency features, i.e., the number of bugs of the in-neighbors and the out-neighbors. We deduce four such features detailed below.

In-Neighbor Bugs: We use the bugs of the in-neighbor source packages of s from the previous time point as features. In particular, we consider the following two features: $\max_{s' \in S_{\mathrm{IN},t}}(|\mathrm{bugs}(s', t-1)|)$ and $\mathrm{median}_{s' \in S_{\mathrm{IN},t}}(|\mathrm{bugs}(s', t-1)|)$ which are respectively the maximum and the median bug counts of the in-neighbours of the package s from the previous time point $(t-1)$.

Out-Neighbor Bugs: Similarly, as above, we use the bugs of the out-neighbor source packages of s from the previous time point as features. Here we consider $\max_{s' \in S_{\mathrm{OUT},t}}(|\mathrm{bugs}(s', t-1)|)$ and $\mathrm{median}_{s' \in S_{\mathrm{OUT},t}}(|\mathrm{bugs}(s', t-1)|)$ which are respectively the maximum and the median bug counts of the out-neighbours of the package s from the previous time point $(t-1)$.

Note that in this case we predict $|\mathrm{bugs}(s, t)|$ using both sets of features above as well as the autoregressive features, and, thereby, rank the source packages.

6 Developer Recommendation

Suppose we observe the evolution of the software ecosystem from time step 1 through $t-1$. In other words, we observe G_τ for $\tau \in [1, t-1]$ along with developer and bug sets associated with each package and time step. Now, for time step t, our goal is to predict $D_{s,t}$. This time, we are interested in *ranking* developers by decreasing suitability for (s, t). Suppose the system returns a ranked order $R_{s,t}$ over a suitable subset of developers. From the gold developer set $D_{s,t}$, we know the 'relevant' or 'good' positions, and can use any ranking evaluation measure such as MRR.

For this experiment, we consider two polices for creating candidate set of developers for (s, t). (1) main list: This list contains the developers who worked on the same source package s in the previous distributions. (2) also use the dependency list, which contains developers who worked on the neighbors (in-neighbors, out-neighbors) of the source package s in the previous distributions. While the first policy goes well with the autoregressive features, the second policy is used while making use of dependency graph.

Model Architecture and Inference: Our objective is to rank a set of candidate developers for each source package and assign the top ranked developer in the test distro for that source package. Let us fix a source package s. $D_{s,\leq t}$ is developer set for package s up to time t. The developers could be collected from s's history only or accessed via network. "$\leq t$" may mean $[t-K, t]$ depending on sliding window width K. Next we train a globally shared model θ for each such horizon h (see Algorithm 1) We observe $D_{s,<h}$ for each package s. Next, we predict $D_{s,h}$, incur any loss and update θ. Model θ induces a score on every developer $d \in D_{s,<h}$. For simplicity call this score $\theta(d)$. For all $d_+ \in D_{s,h}, d_- \notin D_{s,h}$, we want $\theta(d_+) \gg \theta(d_-)$. In our evaluation protocol, all gold developer assignment at time T are used as instances. For evaluating a system at time T alone, apply model θ on candidate set $D_{s,<T}$ (note, not T) and predict ranking $R_{s,T}$ (meaning, sort by decreasing $\theta(d)$) which is evaluated wrt $D_{s,T}$. We categorize the developers present in candidate set in two clusters (i) positive developers, (ii) negative developers. Positive developers are the developers who are present in $D_{s,<h}$ as well as in $D_{s,h}$. Negative developers are the developers who are

present in $D_{s,<h}$ but may leave for other reasons in $D_{s,h}$. Let $\boldsymbol{x}_{s,d+}$ denote feature vectors representing developers in the positive developer set. Similarly let $\boldsymbol{x}_{s,d-}$ denote feature vectors representing developers in the negative developer set. We outline a top level overview of the model architecture in Algorithm 1.

Algorithm 1. Top-level model architecture for developer recommendation.

```
initialize $\theta$
prepare batch loss expression (see below)
for horizon $h = T - K, \ldots, T - 1$ do
    for each package $s$ do
        collect $D_{s,<h}$
        two policies: either same package or via network;
        positive devs $D^{+}_{s,h}$ are $D_{s,<h} \cap D_{s,h}$
        negative devs $D^{-}_{s,h}$ are $D_{s,<h} \setminus D_{s,h}$
        if $D^{+}_{s,h} \neq \varnothing$ and $D^{-}_{s,h} \neq \varnothing$ then
            represent each developer $d$ wrt package $s$ as $\boldsymbol{x}_{s,d}$,
            "an instance" $\langle (s,h); \{\boldsymbol{x}_{s,d} : d \in D^{+}_{s,h}\}, \{\boldsymbol{x}_{s,d} : d \in D^{-}_{s,h}\} \rangle$
            batch loss has been drawn depending on the model chosen (LR/MLP)
            call SGD optimizer for one batch to update $\theta$
        end if
    end for
end for
trained model $\theta$ available at this point
for each package $s$ do
    collect $D_{s,<T}$
    prepare feature vectors $\boldsymbol{x}_{s,d}$ for each $d \in D_{s,<T}$ and apply $\theta(\boldsymbol{x}_{s,d})$
    sort candidate $d$s by decreasing score
    evaluate ranking $R_{s,T}$ wrt gold $D_{s,T}$
end for
```

We employ two different models – (i) Logistic Regression (LR) and (ii) Multilayer Perceptron (MLP) to estimate θ.

LR Model: Our optimisation function is $\theta(\boldsymbol{x}_{s,d}) = \sigma(\text{matmul}(\boldsymbol{x}_{s,d}, W) + b)$ and the loss expression is $loss = \max(0, (\theta(\boldsymbol{x}_{s,d_-}) - \theta(\boldsymbol{x}_{s,d_+}) + 1))$. Here W and b are the learnable parameters that we fit using stochastic gradient descent.

MLP Model: We use a feedforward neural network with one hidden layer. The model equations are $layer1(\boldsymbol{x}_{s,d}) = \tanh(\text{matmul}(\boldsymbol{x}_{s,d}, W_1) + b_1)$ and $\theta(\boldsymbol{x}_{s,d}) = \text{matmul}(layer1(\boldsymbol{x}_{s,d}), W_2) + b_2$ respectively. The loss function is $loss = cost + L_2$ penalty, where the $cost = \sigma(\text{multiply}(a, (\theta(\boldsymbol{x}_{s,d_-}) - \theta(\boldsymbol{x}_{s,d_+}) - b)))$ and $a = \log(1 + \exp(\alpha)), b = \log(1 + \exp(\beta))$. Thus we maintain $a, b > 0$ while α and β are unconstrainted. W_1, b_1, W_2, b_2, α and β are the learnable parameters. The L_2 penalty is calculated over all the learnable parameters. We use stochastic gradient descent.

Feature Construction: Next we discuss how to compute the features $\boldsymbol{x}_{s,d}$.

Pure Autoregressive Features: In this approach, each developer d for a source package s at time t is scored based on autoregressive features. From the changelog, we compute four features—*number of high*, *medium* and *low urgency level* of packages on which the developer has worked, and the *number of bugs* *closed by the developer*. In addition, we introduce a feature which captures the recency—that is, whether the candidate developer worked on this package at time $t-1$.

Inclusion of Network Features: Once again, like bug urgency prediction, we leverage dependency links to improve developer recommendation. Our hypothesis is that developers who have recently contributed to one or more of the in(out)-neighbour packages of a source package should have a greater chance of contributing to the source package itself. This is because, the developers naturally acquire parts of the necessary skill set to contribute to the source package by having already contributed to its closely related packages (in- and out-neighbours) in the recent past. Thus, in addition to the autoregressive features, we add a set of dependency features from previous K distributions – $(t-1)$, $(t-2)$, $(t-3)$ and so on up to $(s, t-K)$. The features are **(i)** $K-1$ binary features telling whether the candidate developer was present in main developer list of $(s, t-i)$ where $i \in [2, K]$, **(ii)** K binary features telling whether the candidate developer was present in the neighbor list of $(s, t-i)$ candidate distribution where $i \in [1, K]$, **(iii)** if the candidate developer is present in the main list of $(s, t-1)$ as well as in at least one of the neighbor list of $(s, t-2)$, $(s, t-3)$, and so on up to $(s, t-K)$, **(iv)** if the candidate developer is present in the neighbor list of $(s, t-1)$ as well as in at least one of the main list of $(s, t-2)$, $(s, t-3)$, and so on up to $(s, t-K)$, and **(v)** if the candidate developer is present in the neighbor list of $(s, t-1)$ as well as in at least one of the neighbor list of $(s, t-2)$, $(s, t-3)$, and so on up to $(s, t-K)$.

7 Experiments and Results

7.1 Bug Urgency Ranking

Experimental Setup: For this experiment, we consider only those source packages whose bug count in any of previous 10 distributions is non zero. We use a train-test split of 5:1 to train and evaluate our model. Let us say we have to predict the bug urgency of all the source packages at time point t. In order to train the model we use the data for all the source packages that appear in the K previous time points. For each time point $(t-1)$ to $(t-K)$ and for every source package s we calculate the autoregressive and dependency features as discussed above; accordingly, the training label for each time point is $|\text{bugs}(s, \cdot)|$ where the $\cdot$ ranges from $(t-1)$ to $(t-K)$. To train the model, we use the random forest regressor[9]. We choose hyperparameters from the following intervals – n_estimators: $[100, 900]$, max_depth: [4, 7], min_samples_split: $[4, 28]$, min_samples_leaf: $[20, 80]$, random_state: $[0, 8]$. We used grid search to find the best parameter combination for both the autoregressive and the dependency approaches.

[9] One may argue that more complex models like point processes could be a possible choice. However note that we only have 20 time points and therefore such complex models cannot be trained sufficiently.

Evaluation: For a given time point t, we rank the source packages based on ground truth $|\text{bugs}(s,t)|$ and the predicted $|\text{bugs}(s,t)|$. We use average ranking method to rank both the score lists. We use Spearman's rank correlation ρ and Kendall's τ for evaluation. We report ρ@25, τ@25 and the ρ, τ for the (quite large) full rank list[10] (see Table 1). We observe that for the most recent time point (i.e., Zesty) the the correlation values are pretty decent ($\rho@25 = 0.582$, $\tau@25 = 0.451$). Use of dependency features bring further benefits ($\rho@25 = 0.60$, $\tau@25 = 0.466$). In fact, for the full rank list also the results using the autoregressive+dependency features are quite good and are significantly different ($p < 0.01$, Mann-Whitney U test) from those using only autoregressive features.

7.2 Developer Recommendation

Upper Bound: We first compute an achievable upper bound using the two policies for creating candidate set as discussed earlier i.e., (i) main list and (ii) main list + dependency network. If the developer of a source package at test distro is present in the candidate developer set then the rank of the developer is set to 1.

Table 1. Spearman's ρ and Kendall's τ for bug urgency ranking—autoregressive only (auto), autoregressive + dependency (+depn). Green cells indicate cases where dependency features bring in additional benefits. ** indicates that the values of ρ and τ for (auto) and (auto, +depn) are significantly different ($p < 0.01$ as per Mann-Whitney U test).

Distribution	ρ@25	τ@25	ρ	τ
	(auto, +depn)	(auto, +depn)	(auto, +depn)	(auto, +depn)
Wily Werewolf	(0.546, 0.546)	(0.407, 0.407)	(0.447, 0.454)**	(0.367, 0.371)**
Yakkety Yak	(0.488, 0.498)	(0.331, 0.331)	(0.260, 0.276)**	(0.218, 0.240)**
Zesty Zapus	(0.582, 0.603)	(0.451, 0.466)	(0.354, 0.380)**	(0.328, 0.351)**

Table 2. Developer recommendation: MRR values comparing our method with different baselines. **: Our results are significantly different from both baselines ($p < 0.001$ for sequence of sets, $p < 0.05$ for majority, Mann-Whitney U test). $^{++}$: Our results are significantly different from majority baseline ($p < 0.01$, Mann-Whitney U test).

Distribution	Autoregressive (auto)				Autoregressive + dependency (auto+depn)		
	Our model	Majority	SeqOfSets	Upper Bound	Our model	Majority	Upper Bound
Wily Werewolf	0.748**	0.736	0.703	0.768	0.763^{++}	0.753	0.844
Yakkety Yak	0.628**	0.607	0.592	0.660	0.642^{++}	0.631	0.740
Zesty Zapus	0.788**	0.773	0.725	0.810	0.794	0.785	0.844

[10] The full rank list has 4K packages on average.

Baselines

Sequence of Sets: In [2], the authors proposed a stochastic model to capture the sequential behaviour of different tasks (such as sending emails, academic collaboration etc.). They proposed two parameters—(i) a correlation parameter (ii) a vector of recency parameters. The correlation parameter measures the chance of repeating the earlier set in future. The recency parameters measure the similarity of a set with the recent one or the oldest one. We directly use their implementation to generate baseline results. Let us choose the test distribution at time point t. We use all the previous time points $(1, t-1)$ for training. For each source package, we fix a correlation probability [2] and perform Monte-Carlo simulation runs to predict a developer in each run. We perform 20 such runs and prepare a ranked list based on the number of occurrences of a developer across these runs (the larger the number of occurrences of a developer across these runs the better is her rank). We perform this experiment for correlation probabilities in the range $[0.1, 0.9]$ in steps of 0.1. We report the results for that correlation probability where the MRR obtained is maximum.

Majority: For each source package, we rank the developers based on the number of times they feature in the last K ($K = 1, 5$, all) distributions (the results are reported for $K = 1$ which turned out to be the best among all choices). In the autoregressive case, for each source package, a developer present the highest number of times in last K distributions receives better rank and so on. In case of the autoregressive + dependency approach, for each source package, we extend our candidate developer set with the developers of its in(out)-neighbors in previous K distributions. Further, we rank the developers of this set based on the number of times they worked on the target source package in last K distributions. Once again, we use the MRR metric to evaluate this approach.

Experimental Setup for Our Method: We use Algorithm 1 to rank the candidate developers using autoregressive and autoregressive + network dependency features. For both the models (i.e., LR and MLP), we try different values of parameters. Through grid search we set the number of epochs to 10 and the learning rate to 0.005. The batch size in our experiment is set to 1. The initial values of α and β are 1 and 0 respectively[11]. We present the results[12] for $K = 5$. For paucity of space we only report the results for the best combination of features and models; in specific, the LR model with autoregressive features and the MLP model with autoregressive + dependency features.

Evaluation: The main results are noted in Table 2. We observe that our methods outperform both the majority and the sequence of sets baseline and are closest to the upper bound. Further, the inclusion of network features always brings additional benefits. For all the three distributions, the results from our model (autoregressive) are better from (a) the sequence of sets baseline ($p < 0.001$, Mann-Whitney U test) and (b) the majority baseline ($p < 0.05$, Mann-Whitney

[11] We also tried other values of α and β but they did not affect the results.

[12] Changes in the value of K does not affect the final results.

U test). Further, for 'Wily' and 'Yakkety', the results from our model (autoregressive + dependency) are better than the majority (+ dependency) baseline ($p < 0.01$, Mann-Whitney U test).

8 Discussion and Conclusion

In this paper we introduced a novel dataset of Ubuntu distributions, motivated by two important software engineering problems: (a) predicting the urgency of a bug and (b) recommending a suitable developer for a package. For both the problems we identify a set of simple autoregressive features which themselves are found to be performing very well. Augmenting these features with the dependency network features brings additional benefits. In future, we would like to investigate further into the dataset to identify if patterns of special relationships exist between developers and bugs and how do these change over time. Discovery of such patterns might allow us to solve the two problems jointly and study other comparable data sets.

Acknowledgement. Soumen Chakrabarti acknowledges support from a Jagadish Bose Fellowship and a Halepete Family Chair. Animesh Mukherjee acknowledges a Humboldt Fellowship and the A K Singh Chair. Pawan Goyal acknowledges support from a Google India AI/ML Research Award.

References

1. Anvik, J.: Automating bug report assignment. In: Proceedings of the 28th International Conference on Software Engineering, pp. 937–940 (2006)
2. Benson, A.R., Kumar, R., Tomkins, A.: Sequences of sets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, pp. 1148–1157. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3219819.3220100
3. Decan, A., Mens, T., Claes, M.: On the topology of package dependency networks: a comparison of three programming language ecosystems. In: Proceedings of the 10th European Conference on Software Architecture Workshops, pp. 1–4 (2016)
4. Janjic, W., Hummel, O., Atkinson, C.: Reuse-oriented code recommendation systems. In: Robillard, M.P., Maalej, W., Walker, R.J., Zimmermann, T. (eds.) Recommendation Systems in Software Engineering, pp. 359–386. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-45135-5_14
5. Kikas, R., Gousios, G., Dumas, M., Pfahl, D.: Structure and evolution of package dependency networks. In: 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), pp. 102–112. IEEE (2017)
6. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**(1), 50–60 (1947)
7. Mao, K., Yang, Y., Wang, Q., Jia, Y., Harman, M.: Developer recommendation for crowdsourced software development tasks. In: 2015 IEEE Symposium on Service-Oriented System Engineering, pp. 347–356. IEEE (2015)
8. Naguib, H., Narayan, N., Brügge, B., Helal, D.: Bug report assignee recommendation using activity profiles. In: 2013 10th Working Conference on Mining Software Repositories (MSR), pp. 22–30. IEEE (2013)

9. de Sousa, O.F., de Menezes, M., Penna, T.J.: Analysis of the package dependency on Debian GNU/Linux. J. Comput. Interdiscip. Sci. **1**(2), 127–133 (2009)
10. Tunio, M.Z., et al.: Impact of personality on task selection in crowdsourcing software development: a sorting approach. IEEE Access **5**, 18287–18294 (2017)
11. Xuan, J., Jiang, H., Zhang, H., Ren, Z.: Developer recommendation on bug commenting: a ranking approach for the developer crowd. Sci. China Inf. Sci. **60**(7), 1–18 (2017). https://doi.org/10.1007/s11432-015-0582-8
12. Yang, Y., Karim, M.R., Saremi, R., Ruhe, G.: Who should take this task? Dynamic decision support for crowd workers. In: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM 2016. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2961111.2962594
13. Ye, B., Wang, Y.: CrowdRec: trust-aware worker recommendation in crowdsourcing environments. In: 2016 IEEE International Conference on Web Services (ICWS), pp. 1–8 (2016)

Mitigating the Position Bias of Transformer Models in Passage Re-ranking

Sebastian Hofstätter[1](✉), Aldo Lipani[2], Sophia Althammer[1],
Markus Zlabinger[1], and Allan Hanbury[1]

[1] TU Wien, Vienna, Austria
{sebastian.hofstatter,sophia.althammer,markus.zlabinger,
allan.hanbury}@tuwien.ac.at
[2] University College London, London, UK
aldo.lipani@ucl.ac.uk

Abstract. Supervised machine learning models and their evaluation strongly depends on the quality of the underlying dataset. When we search for a relevant piece of information it may appear anywhere in a given passage. However, we observe a bias in the position of the correct answer in the text in two popular Question Answering datasets used for passage re-ranking. The excessive favoring of earlier positions inside passages is an unwanted artefact. This leads to three common Transformer-based re-ranking models to ignore relevant parts in unseen passages. More concerningly, as the evaluation set is taken from the same biased distribution, the models overfitting to that bias overestimate their true effectiveness. In this work we analyze position bias on datasets, the contextualized representations, and their effect on retrieval results. We propose a debiasing method for retrieval datasets. Our results show that a model trained on a position-biased dataset exhibits a significant decrease in re-ranking effectiveness when evaluated on a debiased dataset. We demonstrate that by mitigating the position bias, Transformer-based re-ranking models are equally effective on a biased and debiased dataset, as well as more effective in a transfer-learning setting between two differently biased datasets.

1 Introduction

Datasets used to train neural network models are subject to a range of biases, which might constitute unwanted artefacts that should not be incorporated in the trained model [20]. Multiple studies showed that in the ad-hoc retrieval of full documents the text location is of relevant importance, such as the beginning in news articles [7,50] or general web search [23]. In contrast, in this study we specifically probe positional bias in passage collections that are not linked to the previously studied full document relevance distributions. We operate on the assumption, based on the findings of the annotation study of TREC'19 Deep

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 238–253, 2021.
https://doi.org/10.1007/978-3-030-72113-8_16

Learning data [10] by Hofstätter et al. [23], that inside a passage (made up of a few sentences) no word position is supposed to be explicitly favored when matching query and passage sequences.

Transformer-based neural re-ranking models, especially models based on the large-scale pre-trained BERT model [11], have shown a significant improvement in ad-hoc retrieval, where a natural language question is asked by the user and a set of passages is retrieved [35,38]. In this study we evaluate three state-of-the-art Transformer ranking models with varying characteristics: **1)** $\mathbf{BERT_{CAT}}$ [38] using BERT with query and passage concatenation, **2)** $\mathbf{BERT_{DOT}}$ [52] using a dot-product between query and passage BERT classification (CLS) vectors and **3) TK** [22], a lightweight Transformer-Kernel model that does not require pre-training. Each of the three architectures exhibits different strengths and weaknesses, which we describe in Sect. 2.

In the Transformer-architecture, positional information is induced through absolute position information provided by a positional encoding [48]. This positional encoding is added to each non-contextualized representation in a sequence before applying the self-attention. If a bias favoring certain positions in a text exists the Transformer may implicitly incorporate this bias in its word representation as Transformers tend to learn positional information [53]. To our knowledge, the connection between the explicit positional information of the Transformer and positional artefacts in common retrieval collections has not been studied before.

Traditional IR datasets contain relevance judgements for query-document pairs, where a single judgement covers the full document. In contrast to that, QA datasets contain exact location spans of the answer or an answer text that can be partly matched to a position in the document. In our work, we utilize two widely used QA datasets: MS MARCO [3] and SQuAD 2.0 [42]. Both datasets are converted to retrieval collections, by setting paragraphs that were selected to contain the answer as a relevant paragraph for a question. We observe that for the MS MARCO dataset the positions of the mapped answers strongly favor earlier positions in the paragraphs, while the SQuAD 2.0 dataset is more balanced although not completely bias free. The evaluation set is taken from the same distribution, therefore the evaluation is also biased and models overfitting to that bias overestimate their true effectiveness. In the case of MS MARCO this bias is especially concerning as it – because of its size – became the defacto standard collection in the neural re-ranking community, including as base retrieval training for transfer learning [27,55].

We propose to create unbiased versions of the datasets by switching the first and second parts of a passage around a randomly selected position. This approach does not affect the relevance judgements, since they are on a passage level, and allows us to train unbiased re-ranking models as well as to measure the true effectiveness of re-ranking approaches, since relevant matches might now occur in every part of the passage.

We analyze passage term representations to study the position bias induced in Transformer based contextualization and answer ***RQ1*** *How can we measure the* *degree of position bias in the passage representations?* We propose a new metric to measure the mean average term similarity (MATS) per positional delta of all terms in the collection to investigate whether the term representations are independent of the positional encoding or not.

To understand the effects of our debias augmentation in conjunction with Transformer models we further study the following questions:

RQ2 *What effect has the debiasing on the evaluation of Transformers?*

We evaluate the effectiveness of our modifications on the original, as well as the debiased collections. We find that all three models perform better on the original (biased) evaluation, but their effectiveness drops substantially on a debiased evaluation set.

RQ3 *Does a debiased training result in better generalization?*

Training on an unbiased collection shows much more robust results across the evaluated collections and models, which we view as a more accurate indicator for their actual effectiveness.

RQ4 *Do we observe differences in transfer-learning, based on debiased pre-training?*

We demonstrate the usefulness of mitigating bias in the learned representations in the scenario of transfer learning between differently biased collections. We use the larger MS MARCO to pre-train our model variants, before fine-tuning the models on SQuAD 2.0. The bias-mitigated pre-training shows more effective results in the fine-tuning, than starting with a biased pre-training.

The contributions of this work are as follows:

- We measure the positional bias of judgments in two popular Open-QA passage retrieval collections and propose a method to debias the collections;
- We show how three different Transformer-based re-ranking models learn to incorporate the position bias;
- We demonstrate the importance of mitigating the position bias with debiased evaluation sets and the benefit of debiasing in transfer learning between collections.
- We publish the source code of our work at: github.com/sebastian-hofstaetter/transformer-kernel-ranking

2 Background

In this section we first describe the Transformer architecture, followed by the three Transformer-based passage re-ranking models we employ in this study.

2.1 Transformer

The Transformer-layer [48] is a versatile building block for different architectures. In our work we use an encoder structure to encode a sequence and output contextualized representations of this sequence. The Transformer architecture incorporates a natural algorithmic bias on the position of a term in a sequence, because it adds a positional encoding to its input sequence. Vaswani et al. [48] use overlapping sinusoidal-waves per dimension, forming an equidistant relationship among neighbouring terms, whereas Devlin et al. [11] employ a trainable positional embedding for BERT. This positional encoding is important since the Transformer otherwise would be entirely invariant to sequence ordering. However, adding the positional encoding directly to the input means that absolute positional information is retained in the output sequence. Each encoding is unique to a position of the input sequence. Based on the provided training examples, the Transformer may tend to learn position-biased representations.

In this paper we define the Transformer as the sequential use of n Transformer-layers (TLs) as:

$$\begin{aligned} s_{1:m}^{(1)} &= \mathrm{TL}(s_{1:m}) \\ s_{1:m}^{(n)} &= \mathrm{TL}(s_{1:m}^{(n-1)}) \\ \mathrm{TF}(s_{1:m} + e_{1:m}) &= s_{1:m}^{(n)} \end{aligned} \tag{1}$$

where $s_{1:m}$ is the sequence of input embeddings, $e_{1:m}$ is the positional encoding. We call this sequence of recursive applications TF.

2.2 BERT$_{\mathbf{CAT}}$ Ranking Model

First proposed by Nogueira et al. [38] the BERT$_{\mathrm{CAT}}$ approach has become a common way of utilizing the BERT pre-trained Transformer model in a re-ranking scenario [35,55]. It uses the capability of the BERT pre-training approach to compute the relationship of two concatenated sequences, separated by a special SEP token and depending on the BERT version a sequence embedding. The BERT architecture is a simple Transformer model (TF), the effectiveness comes from the masked language and next sentence prediction pre-training. In the BERT$_{\mathrm{CAT}}$ ranking model the query ($q_{1:m}$) and passage ($p_{1:n}$) sequences as well as BERT's special tokens are concatenated (where ; is the concatenation operator) and after the TF computation we select only the first vector of the output sequence (which has been initialized with the special CLS token) and score this pooled representation with a single linear layer (W_s):

$$\mathrm{BERT}_{\mathrm{CAT}}(q_{1:m}, p_{1:n}) = \mathrm{TF}([\mathrm{CLS}; q_{1:m}; \mathrm{SEP}; p_{1:n}])_1 * W_s \tag{2}$$

BERT$_{\mathrm{CAT}}$ is the current state-of-the art in terms of effectiveness, however it requires substantial compute at query time and increases the query latency by seconds [21]. Therefore, we also feature additional models that provide a more balanced efficiency-effectiveness tradeoff.

2.3 BERT$_{\mathbf{DOT}}$ Ranking Model

In contrast to the full-interaction BERT$_{\mathrm{CAT}}$ model, that requires a full online computation of all selected passages, the BERT$_{\mathrm{DOT}}$ model only matches a single

CLS vector of the query with a single CLS vector of a passage [34,52]. This makes it possible to pre-compute contextualized representations for all passages in our index, as well as to employ a vector-based nearest neighbour retrieval approach.

The BERT_{DOT} model, with $\cdot$ as the dot product operator, is formalized by two independent TF computations (and their pooled representations by selecting the first vector output) as follows:

$$\text{BERT}_{\text{DOT}}(q_{1:m}, p_{1:n}) = \text{TF}([\text{CLS}; q_{1:m}])_1 \cdot \text{TF}([\text{CLS}; p_{1:n}])_1 \tag{3}$$

BERT_{DOT} brings strong query time improvements (a few milliseconds latency per query) over BERT_{CAT}, however it still requires the full BERT pre-computation of all indexed passages, which can be very costly depending on the collection size.

2.4 TK Ranking Model

The TK model [22], while also utilizing Transformers, is not based on BERT pre-training, rather it uses shallow Transformers atop word embeddings followed by an explicit term-by-term interaction matrix and scoring with kernel-pooling [51]. In contrast to the BERT approaches TK offers us great control to probe the individual term representations, as it splits the representation learning and their interactions in two distinct parts.

The first part of TK is learning contextualized representations. TK independently contextualizes query ($q_{1:m}$) and passage ($p_{1:n}$) sequences based on pre-trained word embeddings, where the intensity of the contextualization (with TF) is regulated by a gate (α):

$$\begin{aligned} \hat{q}_i &= q_i * \alpha + \text{TF}(q_{1:m})_i * (1 - \alpha) \\ \hat{p}_i &= p_i * \alpha + \text{TF}(p_{1:n})_i * (1 - \alpha) \end{aligned} \tag{4}$$

The two resulting sequences $\hat{q}_{1:m}$ and $\hat{p}_{1:n}$ interact in a match-matrix with a cosine similarity per term pair and each similarity is then activated by a set of RBF-kernels [51]:

$$K_{i,j}^{k} = \exp\left(-\frac{\left(\cos(\hat{q}_i, \hat{p_j}) - \mu_k\right)^2}{2\sigma^2}\right) \tag{5}$$

Kernel-pooling is conceptually a soft-histogram, which counts the number of occurrences of certain similarities. Each kernel focuses on a fixed similarity range with center μ_k and width of σ. Each kernel results in a matrix $K \in \mathbb{R}^{|q| \times |p|}$.

These kernel activations are then summed, first by the passage term dimension j, log-activated, and then the query dimension is summed resulting in a single score per kernel. The final score is calculated by a weighted sum using the linear layer W_s:

$$s = \left(\sum_{i=1}^{|q|} \log\left(\sum_{j=1}^{|p|} K_{i,j}^{k}\right)\right) W_s \tag{6}$$

The kernel-pooling technique is position-independent, as every activation for position j is summed without a weighting them, which allows us to isolate the positional analysis in the Transformer in Sect. 5.

3 Experiment Design

For the first stage indexing and retrieval we use the Anserini toolkit [54] to compute the initial ranking lists with BM25, which we use to generate training and evaluation inputs for the neural models. For our neural re-ranking training and inference we use PyTorch [39] and AllenNLP [15]. We tokenize the text with the fast BlingFire library[1]. As proposed for the MS MARCO dataset [3] we evaluate our neural re-ranking systems using mean reciprocal rank (MRR), normalized discounted cumulative gain (nDCG), and recall (Recall).

For the BERT-based models we use the 6-layer DistilBERT [45] pre-trained weights and the Adam [26] optimizer with a learning rate of $7 * 10^{-6}$. For TK we use pre-trained GloVe [40] word embeddings with 300 dimensions[2] and Adam with a learning rate of 10^{-4} for word embeddings and contextualization layers, 10^{-3} for the kernel-pooling weights.

For the Transformer layers in TK we evaluate 2 layers each with 16 attention heads with size 32 and a feed-forward dimension of 100. For kernel-pooling we set the number of kernels to 11 with the mean values of the Gaussian kernels varying from -1 to $+1$, and standard deviation of 0.1 for all kernels. We use the same sinusoidal positional encodings as Vaswani et al. [48], for the document encodings we shift the start position by 500 to distinguish them from the query encodings.

Table 1. Collection statistics

Collection	# Docs.	# Queries		
		Train	Val.	Test
MS MARCO	8,841,823	502,939	6,980	48,598
SQuAD 2.0	20,239	86,821	5,000	5,928

We train all neural models with a pairwise hinge loss and a batch size of 32. The re-ranking depth for each model instance is tuned on the best mean nDCG@10 of the validation set, as part of an early stopping strategy. For MS MARCO we evaluate a re-ranking depth until 1000 and for SQuAD up to 100.

4 Dataset Analysis and Debiasing

To better understand the neural models, we first need to look at the source of the position bias of the training and evaluation data, specifically the distribution of answer positions in our QA-datasets.

[1] github.com/microsoft/BlingFire.

[2] 42B CommonCrawl: nlp.stanford.edu/projects/glove/.

4.1 Dataset Analysis

The question answering task is strongly linked to ad-hoc information retrieval, as IR provides the first stage of selecting potential candidate passages that contain the natural language answer, that should be presented to a user. In addition to traditional relevance judgements, that cover full documents, the QA datasets also contain short answer strings or exact spans pointing to the answer in a passage.

Using QA datasets to evaluate the retrieval portion of the QA pipeline offers us the unique opportunity of inspecting the answer position, which gives us an insight into the positional importance inside the relevant passages. For SQuAD 2.0 we follow the approach done for MS MARCO [3] and set a passage as relevant to a query if the passage is connected to the answer. We provide an overview of the size of our collections in Table 1, where we observe that MS MARCO is a much larger collection than SQuAD.

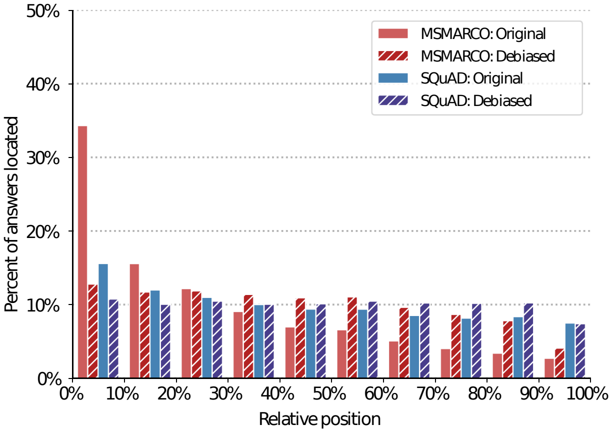


Fig. 1. QA collection in-passage relative answer positions

In Fig. 1 we show the distribution of the QA-answer start positions in their respective relevant passages for the training sets of MS MARCO and SQuAD. To determine the answer positions, we matched the available answer tokens to the passage tokens of the selected passages for both collections and counted all matches. For MS MARCO we omitted answers that could not directly be matched in the passage. In this figure, it is evident that the answer positions in the MS MARCO dataset strongly favor earlier positions in the paragraphs. MS MARCO was created in a retrieval setting, where annotators were given a question and a list of 10 possible paragraphs to judge, which may have favoured passages with answers appearing early in the text. On the other hand SQuAD 2.0, for which annotators were asked to create questions based on a given passage, is relatively unbiased, as the distribution of answer spans in the paragraphs is more uniform.

4.2 Debiasing the Passage Datasets

We have established that MS MARCO answers excessively favor the beginning of a passage, while SQuAD does not. To explicitly study this phenomenon, Hofstätter et al. [23] conducted a fine-grained relevance position study. They found, that if annotators are shown only one query passage pair at a time, annotators select answers uniformly across passages. As we simply cannot re-annotate a collection of the size of MSMARCO with hundreds of thousands of queries, we apply an automatic debiasing method to the existing collections.

For each passage $p_{1:n}$ in the collection we create a *debiased* instance $\tilde{p}_{1:n}$, for which we generate a random number $r \in \{1, \ldots, n\}$, slice the word sequence at the r^{th} index, switch and concatenate the two sub-sequences again:

$$\tilde{p}_{1:n} = [p_{r:n} \, ; \, p_{1:r-1}] \tag{7}$$

As shown in Fig. 1 this approach produces near uniformly distributed relative answer positions for both collections. This approach is minimally invasive as it only breaks the contextualization at a single point per passage, without the need for additional annotations. In a pilot study we also experimented with sentence splitting based rotation, however we found that in the MSMARCO web-page collection too many passages do not contain punctuation and therefore the sentence split approach does not produce uniform answer positions.

Table 2. MATS statistics for TK's contextualized passage vectors. *Lower MATS means less position bias.*

Training	MS MARCO		SQuAD	
	MATS	Std. dev.	MATS	Std. dev.
Original	0.176	0.046	0.056	0.014
Debiased	**0.021**	**0.006**	**0.007**	**0.002**

5 Transformer Bias Analysis

In this section we probe term-wise Transformer representations to determine their bias across positions. Both BERT model variants incorporate their scoring decision mechanism inside the Transformer layers and only use the CLS vector representation, hiding individual term interactions inside the model. The TK model on the other hand utilizes every passage term representation in the cosine match matrix, which allows us to decouple the Transformer layers from the relevance scoring and analyze the passage term representations of a trained model on their own.

We now discuss ***RQ1*** *How can we measure the degree of position bias in the passage representations?* by analyzing the implicit bias of the absolute position of a term in a sequence. If a contextualized vector contains enough information about the original position, then a bias is measurable when we compare different vectors of the same term. We propose to compare the cosine similarity of the contextualized representations r between occurrences of the same term t across different passages computing the average term similarity (ATS) at distance Δa for all terms in the collection $t \in \mathcal{T}$. This is formalized as follows:

$$\begin{aligned} \text{ATS}(\Delta a) &= \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{|C_{t,\Delta a}|} \sum_{(r^t_{a_1}, r^t_{a_2}) \in C_{t,\Delta a}} \cos(r^t_{a_1}, r^t_{a_2}) \\ C_{t,\Delta a} &= \left\{ (r^t_{a_1}, r^t_{a_2}) \middle| \Delta a = |a_1 - a_2|, (t_{a_1}, t_{a_2}) \in \mathcal{C} \right\} \end{aligned} \tag{8}$$

where $r^t_{a_1}$ is the representation of term t at absolute position a_1. The set $C_{t,\Delta a}$ is a set of all couples of representations of term t, which occur in the passages with a distance between their absolute positions of $\Delta a = |a_1 - a_2|$ in the collection $\mathcal{C}$. The mean ATS difference to the first point (MATS) is computed as:

$$\text{MATS} = \frac{1}{\max(\Delta a) - 1} \sum_{i=1}^{\max(\Delta a)} \text{ATS}(0) - \text{ATS}(i) \tag{9}$$

MATS aggregates ATS across all available positions in the passages and allows us to formally compare the different distributions. In Table 2 we show TK's MATS for both collections.

In Fig. 2 we show the ATS for different (Δa) along the x-axis using TK passage term representations on the MS MARCO collection. The shaded area corresponds to the standard deviation. In this plot, an unbiased contextualization would result in a horizontal line, with a uniformly distributed standard deviation of the vectors. A set of contextualized vectors naturally has a standard deviation, as each vector, even for the same term is influenced by different context terms.

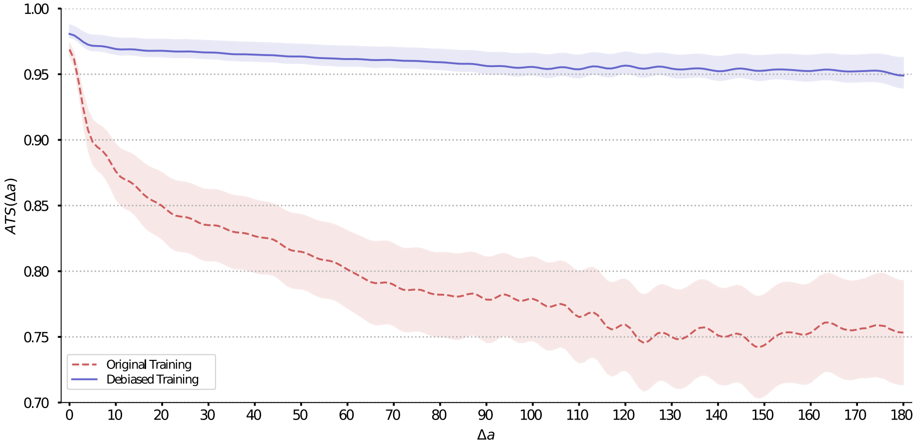


Fig. 2. ATS and standard deviation (y-axis) of same-term occurrences in different passages along positional Δa of each term pair (x-axis) trained and evaluated on the MS MARCO collection with TK passage term representations.

It is evident from observing Fig. 2 and Table 2, that the TK model incurs a strong positional bias, especially for deltas smaller than 20. This shows the influence of the bias in the training data, which conditions the contextualized vectors on their absolute position. Using a debiased training set improves the representations and makes them much less dependent on their position. The SQuAD collection, not pictured in Fig. 2, exhibits a similar pattern, although dampened as the collection is less biased.

6 Retrieval Results

In this section we discuss our effectiveness related research questions with an emphasis on the differences in using the original vs. debiased training and evaluation, including the conclusion we can draw from them:

RQ2 *What effect has the debiasing on the evaluation of Transformers?*
We look at the two collections separately to answer this RQ. In Table 3 we have the results for the heavily-biased MS MARCO collection. We compare each measure by all possible training and evaluation approaches for all three Transformer models. The delta shows the relative difference between the original and debiased evaluation per training type. We can see that across all Transformer models we have a substantial drop in effectiveness when trained on the original training set and evaluated on the debiased set. This shows how the models learn to prioritize the beginning of the passages, and cannot generalize well to the scenario where answers are located in evenly distributed across the passage. The SQuAD results in Table 4 on the other hand offer a different picture with only minor differences between original and debiased evaluation sets. This is to be expected, as we showed in Sect. 4 that the SQuAD collection is almost unbiased in its original form.

RQ3 *Does a debiased training result in better generalization?*
In contrast to the poor original training to debiased test set results on MSMARCO in Table 3, using the debiased training set we observe similar results on the two test sets with little delta across all three models. These debiased training results are better than those using original training to debiased test sets, leading us to the conclusion that these results represent the true generalized effectiveness of the models. For the SQuAD results in Table 4 we make an interesting observation, that some of the debiased trained models outperform those trained on the original training sets when applied to the original test sets.

RQ4 *Do we observe differences in transfer-learning, based on debiased pre-training?*
Finally, we look at a common transfer learning scenario: We utilize the large-scale MSMARCO as first retrieval pre-training and then transfer the trained model to a smaller collection (SQuAD) and train it again. This is especially helpful in production scenarios that require efficient models and do not provide ample training data.

Table 3. MSMARCO re-ranking results of original and debiased training sets (rows) on the original and debiased test sets (columns). Each measure uses a cutoff of 10 and the smallest absolute margin per block is marked in bold.

Model		MSMARCO - test								
		nDCG			MRR			Recall		
	Training	Orig.	Deb.	Δ	Orig.	Deb.	Δ	Orig.	Deb.	Δ
BERT$_{\text{CAT}}$	Original	0.432	0.395	−9.4%	0.372	0.336	−10.7%	0.630	0.594	−6.1%
	Debiased	0.416	0.415	**−0.2%**	0.357	0.355	**−0.6%**	0.617	0.617	**0.0%**
BERT$_{\text{DOT}}$	Original	0.373	0.329	−13.4%	0.316	0.276	−14.5%	0.567	0.509	−11.4%
	Debiased	0.362	0.364	**+0.6%**	0.305	0.307	**+0.7%**	0.555	0.554	**−0.2%**
TK	Original	0.371	0.307	−20.8%	0.312	0.254	−22.8%	0.567	0.484	−17.1%
	Debiased	0.356	0.355	**−0.3%**	0.298	0.296	**−0.7%**	0.551	0.552	**+0.2%**

Table 4. Retrieval effectiveness results of original and debiased SQuAD training sets (rows) on the original and debiased SQuAD test sets (columns). Each measure uses a cutoff of 10 and the smallest absolute margin per block is marked in bold.

Model		SQuAD - Test								
		nDCG			MRR			Recall		
	Training	Orig.	Deb.	Δ	Orig.	Deb.	Δ	Orig.	Deb.	Δ
BERT$_{\text{CAT}}$	Original	0.908	0.902	−0.7%	0.892	0.884	**−0.9%**	0.957	0.956	**−0.1%**
	Debiased	0.910	0.905	**−0.6%**	0.894	0.885	−1.0%	0.959	0.956	−0.3%
BERT$_{\text{DOT}}$	Original	0.780	0.783	+0.4%	0.734	0.738	+0.5%	0.924	0.919	−0.5%
	Debiased	0.784	0.783	**−0.1%**	0.740	0.739	**−0.1%**	0.919	0.919	**0.0%**
TK	Original	0.846	0.840	−0.7%	0.818	0.811	−0.9%	0.933	0.930	−0.3%
	Debiased	0.848	0.844	**−0.5%**	0.820	0.816	**−0.5%**	0.932	0.931	**−0.1%**

In Table 5 we show our transfer learning results. We recall that the original MS MARCO is heavily biased and SQuAD is not. The debiased MS MARCO is closer to the SQuAD answer distribution. In general, using the MS MARCO

Table 5. MS MARCO to SQuAD transfer learning results. Each measure uses a cutoff of 10. Significance is tested between training variants per model with Wilcoxon ($p < 0.05$).

Model			SQuAD original test		
	Train	Sig	nDCG	MRR	Recall
BERT$_{\text{CAT}}$	SQuAD (Original)	*a*	0.908	0.892	0.957
	MS (Original) → SQuAD (Original)	*b*	**0.913**	**0.898**	0.957
	MS (Debiased) → SQuAD (Original)	*c*	0.911	0.896	**0.958**
BERT$_{\text{DOT}}$	SQuAD (Original)	*a*	0.780	0.734	0.924
	MS (Original) → SQuAD (Original)	*b*	0.788^{a}	0.744^{a}	0.922
	MS (Debiased) → SQuAD (Original)	*c*	**0.792**^{ab}	**0.748**^{ab}	**0.927**^{b}
TK	SQuAD (Original)	*a*	0.846	0.818	0.933
	MS (Original) → SQuAD (Original)	*b*	0.854^{a}	0.827^{a}	0.936
	MS (Debiased) → SQuAD (Original)	*c*	**0.857**^{ab}	**0.832**^{ab}	**0.937**

pre-training improves the SQuAD results. For the production scenario models, that enable query independent passage representation caching – BERT_{DOT} and TK – we observe another significant increase in effectiveness on SQuAD using the debiased MS MARCO training. Only BERT_{CAT} does not benefit from the debiased pre-training.

7 Related Work

Biases in Datasets. Recent studies have observed a variety of artefacts (biases) in datasets of several NLP tasks. Gururangan et al. [20] demonstrate that for Natural Language Inference (NLI) datasets it is possible to identify the correct label by only looking at the hypothesis, without observing the premise based on superficial patterns generated while constructing the dataset. This is also confirmed by Poliak et al. [41] and Tsuchiya et al. [47]. McCoy et al. [36] shows that state-of-the-art models follow simple heuristics to identify the correct answer. Glockner et al. [18] show the deficiency of state-of-the-art NLI architecture by testing them in an unbiased dataset. Also QA and Visual QA (VQA) suffer from dataset artefacts. In fact, Jia and Liang [24] show that human-level performance on SQuAD can be achieved by only relying on superficial cues, and Chen et al. [8] show that in NewsQA, 73% of the answers can be predicted by simply identifying the single most relevant sentence. Formal et al. [14] studied the reliance of the ColBERT [25] model on exact term matches in IR.

Another form of bias affecting IR test collections is the pool bias [30,32]. This bias is a side effect of the sampling method used to build these test collections called, the pooling method [29]. This is caused by the presence of non-annotated relevant documents in the collection which makes the evaluation of newly developed retrieval systems less reliable [31,33].

Social biases are another form of bias manifesting in NLP and IR datasets [12,17,44]. In this case these biases are not generated by the way the datasets were constructed but by historical and cultural discriminations manifesting as a prejudice or unfair characterization of the members of a particular group.

Bias Mitigation Methods. The research on the mitigation of these biases has branched out into two directions. One defining methods to mitigate biases when constructing the datasets. The other devising mechanism to make models robust against the presence of bias in datasets. Agrawal et al. [1], Anand et al. [2], and Min et al. [37] develop methods to build unbiased datasets without a variety of biases. Other forms of bias removal consist in learning unbiased representations. Bolukbasi et al. [6] learned unbiased word embeddings to mitigate gender bias. Belinkov et al. [5] propose two probabilistic methods to build models that are more robust to biases and better transfer across datasets. Other methods to develop more robust NLP methods have been developed using adversarial methods [4,9,13,19,28,43]. In the IR setting Gerritse et al. [16] studied and proposed methods to mitigate echo-chamber biases in personalised search.

Modeling Relative Position in Transformers. To overcome this limitation in machine translation tasks, Shaw et al. [46] developed a Transformer with a relation-aware self-attention, which induces the model to learn a relative positional encoding in a translation task. However, we have tested this Transformer-version and observed no improvement over the original version used in this paper. Also in translation tasks, Wang et al. [49] extend the transformer developed by Shaw et al. [46] to model hierarchies based on a dependency tree. We believe that these transformer-versions would benefit from our work, however we leave this to future work.

8 Conclusion

We observed a judgment bias towards the beginning of passages of selected answers in two popular QA datasets used for retrieval. Furthermore, the biased evaluation data hides the existence of this bias in the data. To overcome this problem, we proposed a dataset debiasing method, by switching two parts of a passage split at a random point, as the relevance of word matches in passage retrieval should be position independent.

We showed how the excessive focus on earlier positions in the data propagates through Transformer-based contextualization to form position-biased representations. Our results show that three different Transformer ranking models (BERT_{DOT}, BERT_{CAT}, and TK) trained on the original (biased) MS MARCO collection, substantially lose effectiveness on the debiased version. On the SQuAD collection, acting as an unbiased control dataset, the models do not show this behavior.

We demonstrate that by using a debiased training data transformation, the Transformer models achieve the same performance on biased and debiased datasets, showing the increased generalizability of the models. Finally, we also show that for production-scenario transfer-learning, the debiased pre-training is the most effective strategy. This leads us to the conclusion that going forward, the community should adopt the simple data-transformation for debiasing the MSMARCO pre-training in these transfer-learning scenarios.

References

1. Agrawal, A., Batra, D., Parikh, D., Kembhavi, A.: Don't just assume; look and answer: overcoming priors for visual question answering. In: Proceedings of CVPR (2018)
2. Anand, A., Belilovsky, E., Kastner, K., Larochelle, H., Courville, A.: Blindfold baselines for embodied QA. arXiv preprint arXiv:1811.05013 (2018)
3. Bajaj, P., et al.: MS MARCO: a human generated MAchine Reading COmprehension Dataset. In: Proceedings of NeurIPS (2016)
4. Barrett, M., Kementchedjhieva, Y., Elazar, Y., Elliott, D., Søgaard, A.: Adversarial removal of demographic attributes revisited. In: Proceedings of EMNLP-IJCNLP (2019)

5. Belinkov, Y., Poliak, A., Shieber, S., Van Durme, B., Rush, A.: Don't take the premise for granted: mitigating artifacts in natural language inference. In: Proceedings of ACL (2019)
6. Bolukbasi, T., Chang, K.W., Zou, J.Y., Saligrama, V., Kalai, A.T.: Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Proceedings of NeurIPS (2016)
7. Catena, M., Frieder, O., Muntean, C.I., Nardini, F.M., Perego, R., Tonellotto, N.: Enhanced news retrieval: passages lead the way! In: Proceedings of SIGIR (2019)
8. Chen, D., Bolton, J., Manning, C.D.: A thorough examination of the CNN/daily mail reading comprehension task. In: Proceedings of ACL (2016)
9. Clark, C., Yatskar, M., Zettlemoyer, L.: Don't take the easy way out: ensemble based methods for avoiding known dataset biases. In: Proceedings of EMNLP-IJCNLP (2019)
10. Craswell, N., Mitra, B., Yilmaz, E., Campos, D.: Overview of the TREC 2019 deep learning track. In: TREC (2019)
11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL (2019)
12. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608 (2017)
13. Elazar, Y., Goldberg, Y.: Adversarial removal of demographic attributes from text data. In: Proceedings of EMNLP (2018)
14. Formal, T., Piwowarski, B., Clinchant, S.: A white box analysis of colBERT (2020)
15. Gardner, M., et al.: AllenNLP: a deep semantic natural language processing platform. arXiv preprint arXiv:1803.07640 (2017)
16. Gerritse, E.J., Hasibi, F., de Vries, A.P.: Bias in conversational search: the double-edged sword of the personalized knowledge graph. In: Proceedings of ICTIR (2020)
17. Gezici, G., Lipani, A., Saygin, Y., Yilmaz, E.: Evaluation metrics for measuring bias in search engine results. Inf. Retrieval J. 1–29 (2021). https://doi.org/10.1007/s10791-020-09386-w
18. Glockner, M., Shwartz, V., Goldberg, Y.: Breaking NLI systems with sentences that require simple lexical inferences. In: Proceedings of ACL (2018)
19. Grand, G., Belinkov, Y.: Adversarial regularization for visual question answering: strengths, shortcomings, and side effects. In: Proceedings of the Workshop on Shortcomings in Vision and Language (2019)
20. Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., Smith, N.A.: Annotation artifacts in natural language inference data. In: Proceedings of NAACL (2018)
21. Hofstätter, S., Hanbury, A.: Let's measure run time! Extending the IR replicability infrastructure to include performance aspects. In: Proceedings of OSIRRC (2019)
22. Hofstätter, S., Zlabinger, M., Hanbury, A.: Interpretable & time-budget-constrained contextualization for re-ranking. In: Proceedings of ECAI (2020)
23. Hofstätter, S., Zlabinger, M., Sertkan, M., Schröder, M., Hanbury, A.: Fine-grained relevance annotations for multi-task document ranking and question answering. In: Proceedings of CIKM (2020)
24. Jia, R., Liang, P.: Adversarial examples for evaluating reading comprehension systems. In: Proceedings of EMNLP (2017)
25. Khattab, O., Zaharia, M.: ColBERT: efficient and effective passage search via contextualized late interaction over BERT. In: Proceedings of SIGIR (2020)

26. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
27. Li, C., Yates, A., MacAvaney, S., He, B., Sun, Y.: PARADE: passage representation aggregation for document reranking. arXiv preprint arXiv:2008.09093 (2020)
28. Li, Y., Baldwin, T., Cohn, T.: Towards robust and privacy-preserving text representations. In: Proceedings of ACL (2018)
29. Lipani, A., Losada, D.E., Zuccon, G., Lupu, M.: Fixed-cost pooling strategies. IEEE Trans. Knowl. Data Eng. **33**, 1503–1522 (2019)
30. Lipani, A.: Fairness in information retrieval. In: Proceedings of SIGIR (2016)
31. Lipani, A., Lupu, M., Hanbury, A.: Splitting water: precision and anti-precision to reduce pool bias. In: Proceedings of SIGIR (2015)
32. Lipani, A., Lupu, M., Hanbury, A.: The curious incidence of bias corrections in the pool. In: Proceedings of ECIR (2016)
33. Lipani, A., Lupu, M., Kanoulas, E., Hanbury, A.: The solitude of relevant documents in the pool. In: Proceedings of CIKM (2016)
34. Luan, Y., Eisenstein, J., Toutanova, K., Collins, M.: Sparse, dense, and attentional representations for text retrieval. arXiv preprint arXiv:2005.00181 (2020)
35. MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: CEDR: contextualized embeddings for document ranking. In: Proceedings of SIGIR (2019)
36. McCoy, T., Pavlick, E., Linzen, T.: Right for the wrong reasons: diagnosing syntactic heuristics in natural language inference. In: Proceedings of ACL (2019)
37. Min, S., Wallace, E., Singh, S., Gardner, M., Hajishirzi, H., Zettlemoyer, L.: Compositional questions do not necessitate multi-hop reasoning. In: Proceedings of ACL (2019)
38. Nogueira, R., Cho, K.: Passage re-ranking with BERT. arXiv preprint arXiv:1901.04085 (2019)
39. Paszke, A., et al.: Automatic differentiation in PyTorch. In: Proceedings of NeurIPS-W (2017)
40. Pennington, J., Socher, R., Manning, C.: GloVe: global vectors for word representation. In: Proceedings of EMNLP (2014)
41. Poliak, A., Naradowsky, J., Haldar, A., Rudinger, R., Van Durme, B.: Hypothesis only baselines in natural language inference. In: Proceedings of the CLCS (2018)
42. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of EMNLP (2016)
43. Ramakrishnan, S., Agrawal, A., Lee, S.: Overcoming language priors in visual question answering with adversarial regularization. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Proceedings of NeurIPS (2018)
44. Rekabsaz, N., Schedl, M.: Do neural ranking models intensify gender bias? arXiv preprint arXiv:2005.00372 (2020)
45. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
46. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: Proceedings of NAACL (2018)
47. Tsuchiya, M.: Performance impact caused by hidden bias of training data for recognizing textual entailment. In: Proceedings of LREC (2018)
48. Vaswani, A., et al.: Attention is all you need. In: Proceedings of NIPS (2017)
49. Wang, X., Tu, Z., Wang, L., Shi, S.: Self-attention with structural position representations. In: Proceedings of EMNLP-IJCNLP (2019)
50. Wu, Z., Mao, J., Liu, Y., Zhang, M., Ma, S.: Investigating passage-level relevance and its role in document-level relevance judgment. In: Proceedings of SIGIR (2019)

51. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling. In: Proceedings of SIGIR (2017)
52. Xiong, L., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020)
53. Yang, B., Wang, L., Wong, D.F., Chao, L.S., Tu, Z.: Assessing the ability of self-attention networks to learn word order. In: Proceedings of ACL (2019)
54. Yang, P., Fang, H., Lin, J.: Anserini: enabling the use of Lucene for information retrieval research. In: Proceedings of SIGIR (2017)
55. Yilmaz, Z.A., Yang, W., Zhang, H., Lin, J.: Cross-domain modeling of sentence-level evidence for document retrieval. In: Proceedings of EMNLP-IJCNLP (2019)

Exploding TV Sets and Disappointing Laptops: Suggesting Interesting Content in News Archives Based on Surprise Estimation

Adam Jatowt[1](✉), I-Chen Hung[2], Michael Färber[3],
Ricardo Campos[4](✉), and Masatoshi Yoshikawa[2](✉)

[1] University of Innsbruck, Innsbruck, Austria
adam.jatowt@uibk.ac.at
[2] Kyoto University, Kyoto, Japan
{ichen,yoshikawa}@i.kyoto-u.ac.jp
[3] Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
michael.faerber@kit.edu
[4] Ci2 - Smart Cities Research Center, LIAAD – INESCTEC and Polytechnic Institute of Tomar, Tomar, Portugal
ricardo.campos@ipt.pt

Abstract. Many archival collections have been recently digitized and made available to a wide public. The contained documents however tend to have limited attractiveness for ordinary users, since content may appear obsolete and uninteresting. Archival document collections can become more attractive for users if suitable content can be recommended to them. The purpose of this research is to propose a new research direction of *Archival Content Suggestion* to discover interesting content from long-term document archives that preserve information on society history and heritage. To realize this objective, we propose two unsupervised approaches for automatically discovering interesting sentences from news article archives. Our methods detect interesting content by comparing the information written in the past with one created in the present to make use of a surprise effect. Experiments on New York Times corpus show that our approaches effectively retrieve interesting content.

Keywords: Archival document search · Interestingness · News articles

1 Introduction

Document archives, such as news articles published over past decades, are accumulations of historical records and are important for the humanities and social studies, among others [27]. Accordingly, in recent years, massive digitization efforts of archival documents have been carried out by libraries, national archives,

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 254–269, 2021.
https://doi.org/10.1007/978-3-030-72113-8_17

and numerous other memory institutions. The available data is already considerably large and is continuously growing. For instance, the Chronicling America[1] project has over 5.2 million individual newspaper pages available for viewing and downloading that were published in the USA in the last three centuries. Likewise, Google Books project[2] scanned over 6% of books that were ever published by humanity, many of which are from quite a distant past. In the Web domain, web archives like the Internet Archive[3] are also often used by the general public. Multiple national initiatives [12] have also emerged over the years to crawl national contents. This continuous development of digital document archives allows to learn about historical events and situations directly from primary sources. Yet accessing document archives is different from using a regular search engine, and may lead ordinary users to quickly lose interest or become disappointed. It may be because of the view of history held by some as boring and irrelevant [3,25,33]. This situation calls for research in novel access approaches and retrieval methods that would be adapted to the particular characteristics of archival document collections and could engage user's attention. Such systems should increase archival collections' utility by making them more attractive and interesting to modern users. In this research, we assume in particular that interesting information from the past should contain an element of surprise. Retrieving such content from document archives could surprise and amuse readers as well as evoke their interest, as the contained information would be against the presumed expectations. Note that such information is not easy to be found using a traditional search engine as it requires considerable effort and search skills. Also, although there are websites[4] listing surprising history facts or trivia, they are always manually created.

Although a few studies on identifying content about the unexpected relationships exist, they focus on non-archival data such as Wikipedia [5,36] or current news [20]. Contents in archives have however, particular characteristics due to their age as well as different and often unknown context. In this paper, we focus on extracting sentences from news article archives based on the attributes of content interestingness such as unexpectedness/surprise and importance. We then introduce two unsupervised approaches for discovering interesting content based on these aspects. In particular, the two-layer Mutually Reinforced Random Walk (MRRW) [7] is adapted to capture the novelty and importance in a temporal document collection. The key idea is to rank highly content from the past which was important at that time, yet which is novel or surprising currently. Content importance is modeled by measuring its popularity in the past according to the assumption that popular concepts in the past have more educational value than obscure ones. The second approach involves a topic co-occurrence model used to find surprising and unexpected topic combinations that co-occurred in the past.

[1] https://chroniclingamerica.loc.gov/.

[2] https://books.google.com/.

[3] https://archive.org/.

[4] For example: https://allthatsinteresting.com/interesting-history-facts https://www.thefactsite.com/100-history-facts/ https://parade.com/1099930/marynliles/history-facts/.

Our experiments are performed on the New York Times news corpus [26], which contains documents from 1987 to 2007.

In general, interestingness is a complex concept with little consensus about its definition and scope. It is definitely a challenge to retrieve and recommend attractive content with an objective methodology. Still, this kind of content suggestion should help increase the perceived attractiveness of heritage collections and raise their utility for average users. Successful methods developed for this purpose could be either incorporated as integral components of retrieval mechanisms in archival search engines or could be harnessed to encourage users to start using archives[5].

2 Related Works

Representing Interestingness By Unexpectedness. One of the main problems in finding interesting patterns or data is defining *interestingness* properly. A longtime subject of psychology and cognitive science, the feeling of interestingness was even considered an emotion in the past. Silvia *et al.* [30] and Berlyne *et al.* [4] analyzed interestingness from the viewpoint of cognitive appraisal, which is a personal interpretation of a situation and possible reactions. Within computer science related studies, interestingness was studied in the task of pattern finding in knowledge discovery systems and general databases [13,19,21,31], recommender systems [1] and computational creativity [38]. The Bayesian theory of surprise assumes measuring the difference between posterior and prior beliefs of the observer [2,15]. Based on it, Itti and Baldi [14] developed model that computes expected low-level surprise in video streams which significantly correlates with eye movements of humans watching complex videos.

Geng *et al.* [11] treated interestingness as a broad concept that possibly contains features like reliability, diversity, surprise, and more. Silberschatz *et al.* [28] focused on subjective measures of interestingness, suggesting interesting information should be unexpected and actionable. Unexpectedness was also considered crucial by Padmanabhan *et al.* [23] and Adamopoulos *et al.* [1]. Moreover, the latter introduced serendipity as one of the evaluation measures. Yannakakis *et al.* [40] believed that surprise-focused search maximizes unexpectedness and accordingly proposed a surprise-oriented search algorithm. Tsurel [37] *et al.* assumed that trivia and surprise facts arouse user interest. In line with some of these previous approaches we also model interestingness with the help of the surprise and unexpectedness aspects of information, albeit in our specific case, they arise due to time passage.

Unexpected Relationship Detection. Several studies focused on finding unexpected relationships between data, for example, relationships between entities, which are unexpected. Boldi *et al.* [5] and Tsukuda *et al.* [36] used the

[5] One could imagine a service that automatically detects interesting sentences or headlines for broad topics and publishes them daily on web portals of underlying document archives.

Wikipedia[6] as their underlying knowledge-base to uncover unexpected relations. Tsukuda *et al.* [36] evaluated the unexpectedness of related terms extracted from Wikipedia pages on the basis of relationships of their coordinate terms. Boldi *et al.* [5] focused on finding unexpected links within hyperlinked Wikipedia articles.

Novelty Detection. Interestingess is to some degree related to novelty which should be mentioned here, too. For example, TREC challenge[7], which consists of a set of tracks and tasks, such as TREC Temporal Summarization (TempSum), TREC Knowledge Base Acceleration (KBA), and TREC Novelty Track, has brought about the improvement in the novelty detection for years. Features like *sentence lengths*, *named entities*, and *opinion patterns* were used in Li *et al.* [20] to analyze and improve the novelty detection on the 2002–2004 TREC novelty tracks. Farber *et al.* [10] proposed a new semantic approach to resolve the ambiguities in the languages and extract novel and relevant information from unstructured text documents. For more information, interested readers may refer to the survey on novelty, diversity and serendipity aspects in IR [16] and in recommender systems' evaluation [29].

In general, many of the prior studies developed their methods based on hyperlinked datasets like Wikipedia, which include explicit relationships. Only few tried discovering interesting information from unstructured text. Our research focus is on documents published at different times and subject to change which is inherent in long-term document archives. To the best of our knowledge, the concept of interestingness in archival contents remains largely unexplored.

3 Proposed Approaches

In this section, we describe two novel approaches: *Topic-based Mutually Reinforced Random Walk* and *Topic Pair-based Mutually Reinforced Random Walk*. Before doing that, we first discuss the input data.

3.1 Input Data

In our setting, we assume a sentence to be a retrieval unit. We focus on sentences rather than entire documents for a few reasons. First, we believe that a short but attractive content would have more chance to be read by users than longer text. One of the envisioned applications assumes embedding the automatically extracted content in online archival portals. Doing this based on the entire document may be cumbersome and less flexible. Still, the users could visit the underlying documents from where the interesting sentences were extracted by following added links, especially when headlines are used as is often done in timeline summarization research [24,34], or when snippets are used by regular

[6] https://www.wikipedia.org/.

[7] http://trec.nist.gov/.

search engines. Nevertheless, extending the proposed approaches to returning the entire documents should be relatively easy.

We will make use of two document collections constructed for each input query, D_{past} which represents the set of sentences from a certain time period in the past T_{past} and D_{now} which represents the sentences from the "present" denoted as T_{now} and understood as some recent time span such as the last 6 months or 1 year. Sentences from D_{now} are to be solely used as a reference to support result generation from D_{past}. Our objective is to rank sentences from D_{past} and produce interesting output with the aid of the present collection D_{now}.

3.2 Topic-Based Mutually Reinforced Random Walk

We introduce here our first approach. We generate a two-layered graph G using content from D_{past} and from D_{now} for constructing the layers of the graph. Each node in the graph represents a topic inferred from the respective document collection, while the edge weights represent either similarity or dissimilarity of topics (to be described later). In particular, we run *Latent Dirichlet Allocation* (LDA) to build topic models from the sentences of D_{past} and sentences of D_{now}.

Let us denote the layer in T_{past} as $L_{PP} = \{z_1, z_2, ..., z_i\}$, and the layer in T_{now} as $L_{NN} = \{y_1, y_2, ..., y_j\}$, where z_i and y_j indicate topics from LDA models. Note that the topics in both layers are trained separately on the corresponding datasets, so that the similarities within the two layers will be computed on different topic spaces. We do not mix the datasets when performing the topic modeling in order to determine topics specific to either time period without affecting them by the data from the other time period. $term_{zi}$ and $term_{yj}$ represent the top-scored terms in topic z_i and topic y_j, respectively, according to the determined topic models. We then compute the overlap of the top l terms of topics in order to calculate edge weights. The edge weights within each layer (P(ast) and N(ow)) are computed as follows:

$$Sim^P(z_i, z_j) = \frac{term_{zi} \cap term_{zj}}{l} \tag{1}$$

$$Sim^N(y_k, y_l) = \frac{term_{yk} \cap term_{yl}}{l} \tag{2}$$

while the edge weights between the two layers are calculated as follows:

$$DisSim(z_a, y_b) = 1 - Sim(z_a, y_b) \tag{3}$$

where $Sim(z_a, y_b)$ is calculated similarly to Eqs. 1 and 2, i.e., by measuring term overlap.

We construct such a two-layered graph to find topics that were dominating in the past, yet that are not popular in the present, hence the use of similarity for edge weights within each layer and dissimilarity for edge weights between the layers. Based on this intuition the two-layer *Mutually Reinforced Random Walk* (MRRW) [7] is executed on the graph to assign scores to each topic. MRRW is an algorithm for computing the converged scores of nodes in layered graphs. Given within-layer and between-layer edge weights, the score for each node refers to its importance within the graph computed based on external mutual reinforcement between different layers through the between-layer edges.

The scores of node sets in both layers are reinforced by the following equation:

$$\begin{cases} S_P^{(t+1)} = (1-\alpha)S_P^{(0)} + \alpha \cdot E_{PP}E_{PN}S_N^{(t)} \\ S_N^{(t+1)} = (1-\alpha)S_N^{(0)} + \alpha \cdot E_{NN}E_{NP}S_P^{(t)}. \end{cases} \tag{4}$$

Here $S_P^{(t)}$ and $S_N^{(t)}$ denote the scores of the node set in the past and present layers, respectively, at the t-th iteration. E_{NN}, E_{PP}, E_{NP} and E_{PN} are matrices with the inter- and intra-layers' edge weights. After we apply Eq. 4 to the graph, the score of a node in layer L_{PP} will become higher if the node is more similar to other nodes in this layer and more dissimilar to the nodes in the layer L_{NN}. In this equation, α, which controls the interpolation weight for the propagation part, is set to 0.9 following [7]. The algorithm runs until convergence or until the change of scores becomes very small.

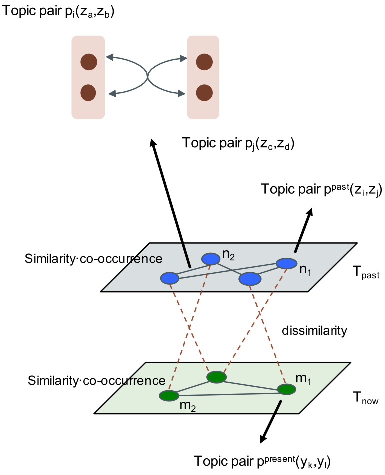


Fig. 1. The overview of the topic pair-based MRRW.

Afterwards, we rank the topics in L_{PP} by their computed scores. As mentioned above, the score of a past topic should be high when this topic is similar to other topics in the past while dissimilar to the topics in the present layer. For each top-ranked topic, we then retrieve the top-n sentences after computing their probability of belonging to that topic.

3.3 Topic Pair-Based Mutually Reinforced Random Walk

Studies in psychology and cognitive science suggest that feeling of unexpectedness and surprise are emotional reactions when people encounter information not conforming to their stereotypical expectations [22]. We hypothesize that a sentence with a rare and uncommon combination of topics would likely be deemed unexpected or surprising. Derezinski *et al.* [9] also view topic diversity as an important element for discovering surprising documents. In this work, instead of measuring the diversity of topic distributions, we propose an approach considering uncommon topic co-occurrences to discover surprising sentences. The underlying intuition is that even if topics are not surprising, their combination could be.

For computation, we again use the two-layered graph, but now the nodes represent topic pairs (a combination of two different topics) based on the set of topics derived from each dataset. Let us denote the layer in T_{past} as $L_{PP} = \{n_1, n_2, ..., n_i\}$, and layer in T_{now} as $L_{NN} = \{m_1, m_2, ..., m_j\}$, where n is a past topic pair $p(z_i, z_j)$ and m denotes a present topic pair $p(y_k, y_l)$ as derived from LDA models. Again, topic models for either time period are trained on its corresponding data, so pair-to-pair similarities within either layer are computed over the topic set corresponding to that layer. We connect any two nodes belonging to the same layer and assign edge weights depending on the similarity and co-occurrence for each topic pair (to be described later). On the other hand, a node pair consisting of nodes from different layers is connected by an edge whose weight represents the nodes' dissimilarity. The concept of *Topic Pair-based MRRW* is visualized in Fig. 1.

When computing the similarity between two nodes (i.e., two topic pairs), we calculate the pair-wise similarity for each possible combination of topics in the two pairs, and use the maximum similarity value as the final edge value. Same as in the above-described Topic-based MRRW, we compute the overlap of the top l topic terms to calculate the similarity and dissimilarity of two topics (Eqs. 2 and 3). We then compute the similarity between two nodes, i.e., two topic pairs in the past $p(z_a, z_b)$ and $p(z_c, z_d)$ as follows:

$$\begin{aligned} Sim^P(p(z_a, z_b), p(z_c, z_d)) = \max\{&Sim^P(z_a, z_c) \cdot Sim^P(z_b, z_d), \\ &Sim^P(z_a, z_d) \cdot Sim^P(z_b, z_c)\} \end{aligned} \tag{5}$$

while the similarity between any two nodes in the present, $p(y_a, y_b)$ and $p(y_c, y_d)$, is calculated by:

$$\begin{aligned} Sim^N(p(y_a, y_b), p(y_c, y_d)) = \max\{&Sim^N(y_a, y_c) \cdot Sim^N(y_b, y_d), \\ &Sim^N(y_a, y_d) \cdot Sim^N(y_b, y_c)\} \end{aligned} \tag{6}$$

Based on the above equations, the edge weights e within each layer are as follows:

$$e^P(n_i, n_j) = Avg\_cooc^P(n_i) \cdot Avg\_cooc^P(n_j) \cdot Sim^P(n_i, n_j) \tag{7}$$

$$e^N(m_i, m_j) = Avg\_cooc^N(m_i) \cdot Avg\_cooc^N(m_j) \cdot Sim^N(m_i, m_j) \tag{8}$$

$Avg\_cooc^P(n_i)$ and $Avg\_cooc^N(m_i)$ are the average co-occurrences of the topics in a given pair in the past and present periods, respectively. They are used here as weights which quantify the importance of topic pairs. The calculation of co-occurrence is done as follows. Sentences in both D_{past} and D_{now} are mapped to a probability distribution over topics to create a sentence-topic matrix, where each row gives a topic distribution for a sentence. The average co-occurrence of the learned topics in each time period is then computed as:

$$Avg\_cooc^{P}(z_i, z_j) = \frac{1}{|D_{past}|} \sum_{d_k \in D_{past}} P(z_i \mid d_k) P(z_j \mid d_k) \tag{9}$$

$$Avg\_cooc^{N}(y_i, y_j) = \frac{1}{|D_{now}|} \sum_{d_k \in D_{now}} P(y_i \mid d_k) P(y_j \mid d_k) \tag{10}$$

where $P(z_i|d_k)$ or $P(y_j|d_k)$ denote the probability of z_i or y_j in d_k, respectively. Finally, edge weights between the different layers are computed in a similar way to Eqs. 6 and 7 as:

$$DisSim(n_a, m_b) = 1 - Sim(n_a, m_b) \tag{11}$$

The final scores are computed by the same equation (Eq. 4) as for MRRW algorithm. After computing final scores of nodes (topic pairs), we rank the topic pairs in T_{past} by their scores, which should be higher if the topic pair is similar to the other topic pairs in the past layer while being dissimilar to the topic pairs in the present layer. For each top ranked topic pair, we then extract top-n sentences after sorting them by their probability of belonging to the corresponding topics.

4 Experimental Settings

4.1 Temporal Document Collection

We use the New York Times (NYT) News collection, which has been frequently utilized in researches of Temporal Information Retrieval [6,17] and alike. The corpus includes news articles published from 1987 to 2007. The documents contain metadata labels such as date, title, category, leading paragraph, full-text, and more. In the experiments, we divide this news archive into two parts: one from Jan. 1987 to Dec. 1989, representing past documents, and the other one containing documents published from Jan. 2005 to Dec. 2007 to represent information of the present. Naturally, the latter part is not exactly representing the "present", and is rather a compromise resulting from the lack of free datasets that would be long enough (e.g., a span of at least three different decades or more) and that, at the same time, would contain also most recent documents. When it comes to the length of time periods our choice results from the need for striking a balance between having the size of data in both the parts of the collection sufficiently large for generating topics and between maintaining a sufficiently long time gap that separates these two dataset parts. We will then process content that is roughly 30 years old as seen from now and that was published during 3 years' long time frame.

In the experiments, we consider five broad categories of concepts inspired by news categories of NYT: *Economy*, *Places*, *Politics*, *Sports*, and *Technology* as broad concepts tend to be often used by ordinary users accessing document archives [8,18,35]. Each category includes 4 general concepts resulting, in total, in 20 different concepts. Table 1 gives the list of categories and their concepts.

Table 1. List of categories and their concepts.

Category	Concept
Economy	Currency, economy, trade, market
Places	Japan, Florida, Los Angeles, New York
Politics	Election, president, nomination, poll
Sports	Basketball, team, olympics, sport
Technology	Machine, computer, plane, technology

4.2 Preprocessing

We first find all sentences that mention the concepts using the Solr[8] search engine. We use only sentences being either the leading paragraph or the title of a document as these are most interpretable and self-contained. To ensure better understandability, we remove sentences shorter than 10 words as well as overly long sentences (longer than 50 words).

Next, we trim sentence contents by removing stopwords and punctuations using NLTK library[9]. Lemmatization is performed to handle inflections and to obtain correct base forms of words. We then use TF-IDF vectors for sentence representation[10]. The number of topics in LDA models has been empirically set to 100 for all the approaches and the number l of top terms was also set to 100.

4.3 Baselines

Besides the two proposed approaches, we also test the following ones:

Random: We return randomly ordered sentences from the pool of candidate sentences from the past documents.

Centroid: This method ranks sentences in D_{past} by their dissimilarity to the centroid vector, which is the average TF-IDF vector of all sentences in D_{now}. It is expected to extract sentences which are less known to current users.

MRRW: This method ranks sentences by simply applying MRRW [7] on the two layers (past and present) composed of sentences treated as nodes.

[8] https://lucene.apache.org/solr/.
[9] https://www.nltk.org/.
[10] We have also experimented with embedding models but they did not perform better.

Topic Co-occurrence: Similarly to the proposed Topic Pair-based MRRW method, we use the concept of surprising topic pairs. However, the calculation is done without building a two-layered graph and running the random walk. To find the co-occurring topics, we use Latent Dirichlet Allocation to build a topic model over the combined sentences from D_{past} and D_{now}. Sentences in both D_{past} and D_{now} are then mapped to a probability distribution over topics $t_i \in T$. As a result, we obtain a sentence-topic matrix, where each row gives a topic distribution for a sentence. We then calculate the average co-occurrence of the learned topics in each time period using similar way as in Eqs. 9 and 10.

Topic pairs that frequently co-occur in D_{past} yet rarely in D_{now} will be ranked high by the following equation:

$$S(t_i, t_j) = \frac{Avg\_cooc^{P}(t_i, t_j) - Avg\_cooc^{N}(t_i, t_j)}{Avg\_cooc^{N}(t_i, t_j) + Avg\_cooc^{P}(t_i, t_j)} \tag{12}$$

The score of a sentence is computed by aggregating the scores of the probability of different topic pairs in the sentence. The top n sentences are then retrieved for each top-ranked topic pair same as in Topic Pair-based MRRW method.

4.4 Data Annotation

We use Figure Eight[11], a popular crowdsourcing platform to evaluate the results. We first pooled the top 15 results for the 20 queried concepts for each of the 6 tested methods[12]. This resulted in an evaluation dataset consisting of 1,800 sentences from the New York Times collection that were published between 1987 and 1989. Judges were then asked to assess the sentences based on their interestingness and surprise, and give scores ranging from 1 to 4. Each sentence in the dataset was scored by five evaluators. The final decision for a sentence to be considered as positive was made based on the average value of judgments. We used the conservative threshold according to which a sentence is deemed positive if its average judgement value is over 2.5.

Table 2. Main results.

	P@1	P@5	P@10	P@15	MRR	MAP
Random	5.00	21.00	18.5	18.33	28.81	28.75
Centroid	10.00	18.00	15.00	16.67	28.94	27.10
Topic co-occurrence	15.00	19.00	19.00	20.33	29.58	26.55
MRRW [7]	25.00	28.00	28.00	30.33	46.42	36.94
Topic-based MRRW	**35.00**	27.00	27.00	27.66	**51.54**	39.87
Topic Pair-based MRRW	15.00	**29.00**	**32.00**	**31.33**	50.04	**39.98**

[11] https://www.figure-eight.com/.

[12] We set n=5 as the number of top sentences returned for every top-ranked topic in *Topic-based MRRW*, and for each top-ranked topic pair in *Topic Pair-based MRRW* method and *Topic co-occurrence* methods.

5 Experimental Results

5.1 Main Results

Table 2 shows the overall results according to the Precision@1, 5, 10, 15, Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP).

We found that both of the proposed approaches perform the best on MRR and MAP when compared to the baselines. For the precision, either *Topic-based MRRW* or *Topic Pair-based MRRW* produces the best results depending on the cut-off level. Out of the two proposed approaches, *Topic Pair-based MRRW* appears to be superior, except for P@1 for which *Topic-based MRRW* produces higher quality output. The third best performing method is MRRW, which indicates that graph-based approaches are effective for our task. The satisfactory performance of both proposed approaches, yet with certain differences, suggests also that it may be worthy to experiment with their combination in the future.

Looking at the performance in terms of MRR and MAP over particular categories as shown in Tables 3 and 4, we can observe that although different methods perform best for different category types, the proposed approaches, especially, *Topic-based MRRW* tend to be most stable. The results of the *Topic-based MRRW* have consistently high interestingness rates across all the concept categories. The results for *MRRW* indicate that it has about 8% to 11% drop when compared to the best performing approach, yet it still outperforms the other baselines by a good margin. On the other hand, *Centroid* method, as the most intuitive and simple one, performed quite similar to the *Random* baseline. Similarly, *Topic co-occurrence* – a direct approach that uses a single shared topic space – is not enough to produce effective results.

The *Technology* category seems to be easiest for the interesting content finding task. Most of the tested methods are able to return many interesting contents in this category. This is likely because technology has changed quite much over the last thirty years, and thus facts and opinions from the past on technology-related news are quite different from the present. Technology is ubiquitous these days and perhaps also more appealing to users.

5.2 Case Studies

We discuss now a few examples of sentences recommended by our approaches. The first sentence that we want to highlight is the following:

> *"Of the 715 apartment fires in Moscow last month, 90 were blamed on exploding television sets, a statistic the Soviet press has viewed as an alarming commentary on soviet technology." (Dec 1987)*

The notion of exploding TV sets in USSR is obviously quite different from our common sense; yet these kinds of unfortunate events were reported several

Table 3. Performance according to different categories by MRR.

	Economy	Places	Politics	Sports	Tech	*Average*
Random	**47.50**	10.49	35.00	23.96	27.08	*28.81*
Centroid	41.67	43.75	10.83	22.62	25.83	*28.94*
Topic co-occurrence	5.20	18.94	8.33	44.58	70.83	*29.58*
MRRW [7]	19.58	**47.92**	25.00	**64.58**	75.00	*46.42*
Topic-based MRRW	40.63	42.36	51.79	58.33	64.58	***51.54***
Topic Pair-based MRRW	43.94	39.40	**52.27**	14.58	**100.00**	*50.04*

Table 4. Performance according to different categories by MAP.

	Economy	Places	Politics	Sports	Tech	*Average*
Random	**38.22**	16.04	**38.57**	23.69	27.25	*28.75*
Centroid	34.38	**39.99**	11.94	17.67	31.50	*27.10*
Topic co-occurrence	6.14	26.67	10.20	31.02	58.71	*26.55*
MRRW [7]	21.93	34.74	19.67	45.37	62.99	*36.94*
Topic-based MRRW	31.07	29.33	32.18	**53.29**	53.47	*39.87*
Topic Pair-based MRRW	34.27	28.40	37.35	15.00	**84.86**	***39.98***

times in 1987[13]. Another example extracted is also rather opposite from what one would claim nowadays:

> *"Laptop computers are great in theory but disappointing in real life." (Oct 1988)*

One could try to explain this example by potentially high expectations put on personal computing tools in the past, coupled with rather low specs of machines at hand and the lack of infrastructure (e.g., wifi spots). Whatever the reasons were, this kind of content might stimulate deliberating about technology evolution and all the "bumps in its evolutionary path" over time. It might serve as an "invitation" for closer reading of the original document or related ones in search for explanation.

Some of the examples from the politics category show certain resemblance to the present day's trade tensions yet the actors are now quite different:

[13] Anecdotally, this particular example triggered recollections of childhood memories of one author. His grandparents owned a USSR-produced TV set and often warned him not to sit close to it when he visited their home. Only now, he could understand that the fears of his relatives were actually not without a substance. On a more general note, exploring news archives offers chances for learning about history, and might sometimes even lead to serendipitous discoveries and recollections as this example demonstrates.

> *"President Reagan is likely to soon lift some of the trade sanctions imposed on Japan seven months ago during a dispute over Japanese dumping of computer chips, the Administration said today." (Nov 1987)*

> *"Prime Minister Yasuhiro Nakasone today accused the Toshiba Machine Company of betraying Japan by selling militarily sensitive technology to the Soviet Union." (Jul 1987)*

We also found opportunities for improvement of our approaches. Take the following two sentences as examples:

> *"Zenith said the new SX laptop could operate for more than three hours on the battery before it needed recharging." (Oct 1989)*

> *"The Houston-based Company Show edits new battery-operated SLT/286 lap-top system, a computer that it said matches the function of desktop computers but comes in a lunch box-sized, 14-pound package." (Nov 1988)*

The news on developments in battery-operated laptops and on battery lifetimes seemed to be frequently reported in the past. However, they do not appear often in the present-day news about laptops. The reason is that battery improvements became rather commonsense nowadays along with the proliferation of producers and, in general, along with the rapid technology progress. Thus they tend not to be special enough to be reported in news articles. Nevertheless, such sentences are returned by our approach (topics popular in the past but not popular now) as our methods do not capture implicit knowledge. Incorporating approaches that use common sense reasoning and analysis as well as extract implicit knowledge could then become advantageous in future research. Another observation based on these examples is that numerical values, such as product specifications (e.g., "14-pound" (or over 6 kg) as in the last example), could be extracted and compared to the currently typical ones for finding striking differences. Also, aspects that are obvious at present but were overly emphasized in the past (e.g., "a lunch box-sized" or "battery-operated" as in the above examples) could be considered. Overall, studying elements of surprise and interestingess in archival news could be opening the door for new ideas that lead to automatic approaches for generating/recommending the content of museums and exhibitions.

6 Conclusions

Making document archives more attractive and popular among ordinary users remains a key and perennial goal of the archival community [32,39]. The attractiveness and, related to it, the level of use of document archives among ordinary users is still moderate and can be improved by applying suitable techniques. To this end, we proposed a novel research problem of finding interesting content from news article archives and we approach this challenging task in a fully unsupervised manner. Our key idea is based on data comparisons across time for capturing information surprising to current users. We note that interestingness may have several aspects according to users' age, culture and other backgrounds. The particular, objective measure of interestingness we used in our methods (i.e., surprise arising due to time passage) naturally cannot exhaustively capture the entire spectrum of interestingness.

In the future, we plan to focus on improving the quality of results. As also discussed in [1], it is important to avoid returning trivial and obvious content (in our case, some returned sentences are novel but unsurprising), or one poorly understandable by users, e.g., due to the lack of necessary context.

Acknowledgments. This work has been partially funded by MEXT JSPS Grant-in-Aid. Ricardo Campos, one of the authors of this paper was financed by the ERDF – European Regional Development Fund through the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 and by National Funds through the Portuguese funding agency, FCT - Fundao para a Cincia e a Tecnologia within project PTDC/CCI-COM/31857/2017 (NORTE-01-0145-FEDER-03185). This funding fits under the research line of the Text2Story project. The first author was employed by Kyoto University when the first version of this paper was created.

References

1. Adamopoulos, P., Tuzhilin, A.: On unexpectedness in recommender systems: or how to better expect the unexpected. ACM TIST **5**(4), 54 (2015)
2. Baldi, P., Itti, L.: Of bits and wows: a Bayesian theory of surprise with applications to attention. Neural Netw. **23**(5), 649–666 (2010)
3. Berk, N.A., Gültekin, F.: The topics that students are curious about in the history lesson. Procedia-Soc. Behav. Sci. **15**, 2785–2791 (2011)
4. Berlyne, D.E.: Conflict, arousal, and curiosity (1960)
5. Boldi, P., Monti, C.: LlamaFur: learning latent category matrix to find unexpected relations in Wikipedia. In: Proceedings of WebScience, pp. 218–222. ACM (2016)
6. Campos, R., Dias, G., Jorge, A.M., Jatowt, A.: Survey of temporal information retrieval and related applications. ACM Comput. Surv. **47**(2), 15:1–15:41 (2014)
7. Chen, Y.N., Metze, F.: Two-layer mutually reinforced random walk for improved multi-party meeting summarization. In: 2012 IEEE Spoken Language Technology Workshop (SLT), pp. 461–466. IEEE (2012)
8. Costa, M., Silva, M.: Understanding the information needs of web archive users. In: The 10th International Web Archiving Workshop (2011)
9. Derezinski, M., Rohanimanesh, K., Hydrie, A.: Discovering surprising documents with context-aware word representations. In: 23rd International Conference on Intelligent User Interfaces, pp. 31–35. ACM (2018)
10. Färber, M.: Semantic Search for Novel Information, vol. 31. IOS Press, Amsterdam (2017)
11. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: a survey. ACM Comput. Surv. (CSUR) **38**(3), 9 (2006)
12. Gomes, D., Cruz, D., Miranda, J., Costa, M., Fontes, S.: Search the past with the Portuguese web archive. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 321–324 (2013)
13. Hidi, S., Baird, W.: Interestingness-a neglected variable in discourse processing. Cogn. Sci. **10**(2), 179–194 (1986)

14. Itti, L., Baldi, P.F.: A principled approach to detecting surprising events in video. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Siego, CA, pp. 631–637, June 2005
15. Itti, L., Baldi, P.: Bayesian surprise attracts human attention. Vision Res. **49**(10), 1295–1306 (2009)
16. Kaminskas, M., Bridge, D.: Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Tran. Interact. Intell. Syst. (TiiS) **7**(1), 1–42 (2016)
17. Kanhabua, N., Anand, A.: Temporal information retrieval. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1235–1238 (2016)
18. Koolen, M., Kamps, J.: Searching cultural heritage data: does structure help expert searchers? In: Adaptivity, Personalization and Fusion of Heterogeneous Information, pp. 152–155. Citeseer (2010)
19. Kuznetsov, S.O., Makhalova, T.: On interestingness measures of formal concepts. Inf. Sci. **442**, 202–219 (2018)
20. Li, X., Croft, W.B.: Improving novelty detection for general topics using sentence level information patterns. In: Proceedings of CIKM, pp. 238–247. ACM (2006)
21. Liu, B., Hsu, W., Mun, L.F., Lee, H.Y.: Finding interesting patterns using user expectations. IEEE Trans. Knowl. Data Eng. **11**(6), 817–832 (1999)
22. Macrae, C.N., Bodenhausen, G.V.: Social cognition: thinking categorically about others. Annu. Rev. Psychol. **51**(1), 93–120 (2000)
23. Padmanabhan, B., Tuzhilin, A.: Unexpectedness as a measure of interestingness in knowledge discovery. Decis. Support Syst. **27**(3), 303–318 (1999)
24. Pasquali, A., Mangaravite, V., Campos, R., Jorge, A.M., Jatowt, A.: Interactive system for automatically generating temporal narratives. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11438, pp. 251–255. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15719-7_34
25. Pessent, E.: Is history irrelevant? Dissent Mag., pp. 1, June 1971
26. Sandhaus, E.: The New York times annotated corpus. Linguist. Data Consortium Philadelphia **6**(12), e26752 (2008)
27. Schwartz, J.M., Cook, T.: Archives, records, and power: the making of modern memory. Arch. Sci. **2**(1–2), 1–19 (2002)
28. Silberschatz, A., Tuzhilin, A.: What makes patterns interesting in knowledge discovery systems. IEEE TKDE **8**(6), 970–974 (1996)
29. Silveira, T., Zhang, M., Lin, X., Liu, Y., Ma, S.: How good your recommender system is? A survey on evaluations in recommendation. Int. J. Mach. Learn. Cybern. **10**(5), 813–831 (2019)
30. Silvia, P.J.: What is interesting? Exploring the appraisal structure of interest. Emotion **5**(1), 89 (2005)
31. Spyropoulou, E., De Bie, T., Boley, M.: Interesting pattern mining in multi-relational data. Data Min. Knowl. Discov. **28**(3), 808–849 (2014)
32. Stiller, J.: A framework for classifying interactions in cultural heritage information systems. Int. J. Heritage Digital Era **1**(1), 141–146 (2012)
33. Strauss, V.: Why so many students hate history - and what to do about it? The Washington Post (2017)
34. Tran, G., Alrifai, M., Herder, E.: Timeline summarization from relevant headlines. In: Hanbury, A., Kazai, G., Rauber, A., Fuhr, N. (eds.) ECIR 2015. LNCS, vol. 9022, pp. 245–256. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16354-3_26

35. Trant, J.: Understanding searches of a contemporary art museum catalogue: a preliminary study. Report, Archives & Museum Informatics (2006)
36. Tsukuda, K., Ohshima, H., Yamamoto, M., Iwasaki, H., Tanaka, K.: Discovering unexpected information on the basis of popularity/unpopularity analysis of coordinate objects and their relationships. In: Proceedings of SAC, pp. 878–885. ACM (2013)
37. Tsurel, D., Pelleg, D., Guy, I., Shahaf, D.: Fun facts: automatic trivia fact extraction from Wikipedia. In: Proceedings of WSDM, pp. 345–354. ACM (2017)
38. Veale, T., Cardoso, A.: Computational Creativity: The Philosophy and Engineering of Autonomously Creative Systems. CSACS, vol. 31. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-43610-4
39. Warwick, C., Terras, M., Huntington, P., Pappa, N.: If you build it will they come? The LAIRAH study: quantifying the use of online resources in the arts and humanities through statistical analysis of user log data. Literary Linguist. Comput. **23**(1), 85–102 (2007)
40. Yannakakis, G.N., Liapis, A.: Searching for surprise. In: Proceedings of the International Conference on Computational Creativity (2016)

Label Definitions Augmented Interaction Model for Legal Charge Prediction

Liangyi Kang[1,2], Jie Liu[1,2(✉)], Lingqiao Liu[3(✉)], and Dan Ye[1,2]

[1] State Key Laboratory of Computer Science, Institute of Software,
Chinese Academy of Sciences, Beijing, China
{kangliangyi15,ljie,yedan}@otcaix.iscas.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China
[3] University of Adelaide, Adelaide, Australia
lingqiao.liu@adelaide.edu.au

Abstract. Charge prediction, determining charges for cases by analyzing the textual fact descriptions, is a fundamental technology in legal information retrieval systems. In practice, the fact descriptions could exhibit a significant intra-class variation due to factors like non-normative use of language by different users, which makes the prediction task very challenging, especially for charge classes with too few samples to cover the expression variation. In this work, we explore to use the charge (label) definitions to alleviate this issue. The key idea is that the expressions in a fact description should have corresponding formal terms in label definitions, and those terms are shared across classes and could account for the diversity in the fact descriptions. Thus, we propose to create auxiliary fact representations from charge definitions to augment fact descriptions representation. Specifically, we design label definitions augmented interaction model, where fact description interacts with the relevant charge definitions and terms in those definitions by a sentence- and word-level attention scheme, to generated auxiliary representations. Experimental results on two datasets show that our model achieves significant improvement than baselines, especially for dataset with few samples.

Keywords: Legal charge prediction · Label definitions · Interaction model · Auxiliary representation · Augmented fact representation

1 Introduction

The task of charge prediction is to determine appropriate charges, such as *theft* or *robbery*, for given cases by analyzing the textual fact descriptions. Automating charge prediction technology could be practically useful for online legal assistant systems, which provide legal consulting for users in a cost-effective way.

In practice, users have different writing habits while inputting the fact of cases. Fact descriptions comprise a substantial amount of diverse non-normative use of language. For example, the cases of robbery in Fig. 1 all involve "theft",

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 270–283, 2021.
https://doi.org/10.1007/978-3-030-72113-8_18

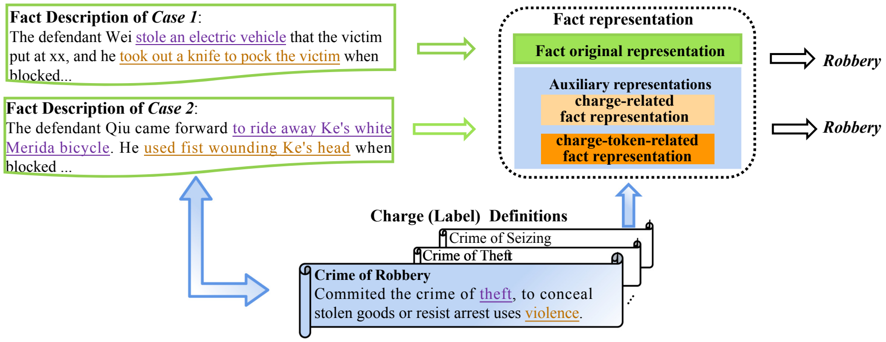

Fact Description of Case 1:
The defendant Wei stole an electric vehicle that the victim put at xx, and he took out a knife to pock the victim when blocked...
Fact Description of Case 2:
The defendant Qiu came forward to ride away Ke's white Merida bicycle. He used fist wounding Ke's head when blocked ...
Fact representation
Fact original representation
Auxiliary representations
charge-related fact representation
charge-token-related fact representation
Robbery
Robbery
Charge (Label) Definitions
Crime of Seizing
Crime of Theft
Crime of Robbery
Commited the crime of theft, to conceal stolen goods or resist arrest uses violence.

Fig. 1. Illustration of our method. Green boxes are two robbery case descriptions and the blue box contains label definitions–charge definitions. The related charges are identified (indicated by the blue double arrow) via sentence-level attention and aggregated to create the auxiliary representation I, charge-related fact representation. Then key words in cases align to terms in identified charge definitions via word-level attention (aligned words are labeled by the same color), which are then formed as the auxiliary representation II, charge-token-related fact representation. The two auxiliary representations combine with original fact representation to predict the label–robbery. (Color figure online)

but the legal term "theft" may be implicitly expressed like *"stole an electric vehicle"* or *"came forward to ride away Ke's white Merida bicycle"*. Consequently, the representation of fact descriptions may exhibit considerable intra-class variation which may lead to prediction failure at the test stage. This could be more pronounced for charge classes with only a few examples since the samples are not sufficient for learning a predictive model robust to expression variation.

To address this issue, we introduce label definitions, the charge definition, to create more robust fact representations for charge prediction. We propose to create auxiliary fact representations from the charge definitions to augment the fact representation. Those auxiliary representations are essentially projections of the fact description in the semantic space of charge definitions. Our motivation is that the expressions in a fact description should have corresponding formal terms in label definitions, and those formal terms can provide an alternative view of the expressions in fact description. Note that many of those formal terms are shared across charge classes and are less diverse. Thus, using elements in charge definitions to re-interpret fact description and generate auxiliary representations could have the potential to account for the diversity in the fact description.

Specifically, we design a label definitions augmented interaction model integrating sentence- and word-level attention to generate two auxiliary fact representations. We identify the related charge definitions through sentence-level attention between fact description and charge definitions, and then aggregate the holistic features of relevant charge definitions to create the first auxiliary representation, named as charge-related fact representation. The relevant charge definitions identified in the course of producing the first auxiliary representation will also serve for creating the second auxiliary representation. To create the second representation, we further consider finer-grained word-level attention between the fact description and related charge definitions. Relevant words from relevant charge definitions are attended and aggregated through a recurrent neural network to generate the second auxiliary representation, named as charge-token-related fact representation. We illustrate our model by an example in Fig. 1. Case 1 and case 2 in Fig. 1 belong to the same class, *robbery*, but with different expressions. With the proposed method, they will be firstly related to the charge definition of *robbery*. Then the statements of *"stole an electric vehicle"* and *"took out a knife to poke the victim"* in case 1, *"came forward to ride away Ke's white Merida bicycle"* and *"used fist wounding Ke's head"* in case 2 will be softly aligned to the terms "theft" and "use violence" in *robbery* definition through interaction. By reinterpreting the fact descriptions through aligned terms, those two cases become more similar. The final charge prediction is based on the original and auxiliary fact representations, and one can expect the prediction made on this fact representation will be more robust.

To investigate the advantage of our method on charge prediction, we conduct experiments on real-world datasets. Experimental results show that our model outperforms baselines, especially on dataset with few samples. We also conduct ablation studies to analyze the effectiveness of each component in our model, and visualize the impact of introducing charge definitions.

2 Related Works

Charge prediction focuses on learning representation of fact descriptions and feeding them into classifiers to make the judgment. At the early stage, [13–15,18] attempt to extract shallow text features from fact descriptions or create hand-crafted features to represent fact descriptions, which are hard to generalize to large datasets due to the diverse expression of fact descriptions. Inspired by the success of deep learning, [8,16,26,27] employ neural models with external information to capture the high-level semantic information. [16] use a separate two-stage scheme to extract the related articles and then attend them attentively to fact representation. [8] design 10 legal attributes to help the few-shot charges prediction. They both need a large amount of feature engineering, either design features or relations between subtasks. LJP [27] and MPBFN [25] model multiple legal subtasks by multi-task learning framework to assist prediction. LegalAtt [2] uses law article to perfect fact representation. However, one article may include more than one charges, which could obscures the fact representation. Instead, we augment fact representation to assist charge prediction by creating auxiliary representation from charge definitions by an interaction model.

Our model is also related to attention and memory in deep learning [1,6,17,19,20,22]. Although researchers propose various neural architectures with memory and attention for NLP problems [7,12,21], they either only consider sentence-level or only word-level alignment between sentences. In contrast, we combine them jointly to form auxiliary representation, where sentence-level interaction identifies relevant charges, and a finer-grained word-level interaction on the top of identified charge definitions makes the generated fact representation more robust.

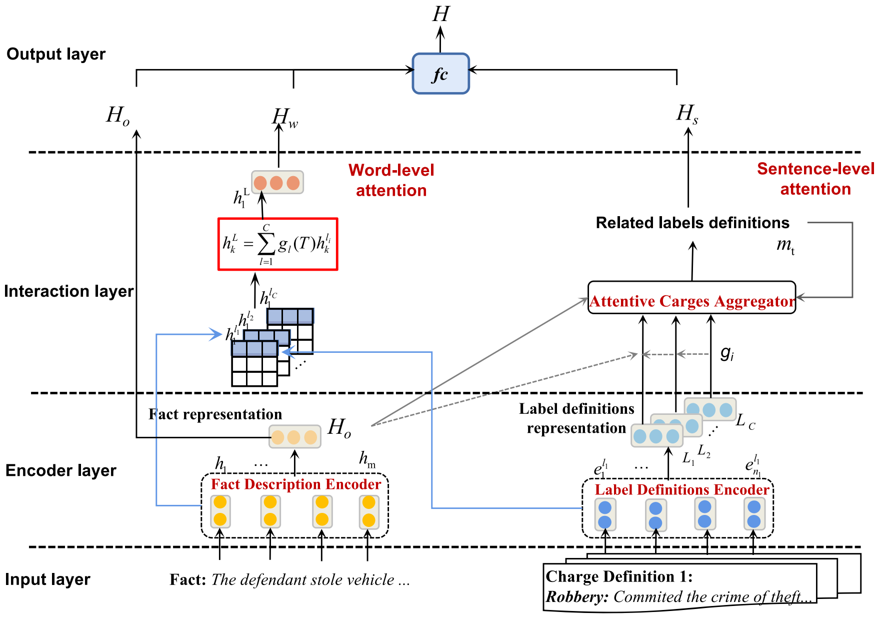


Fig. 2. The architecture of our model. Fact description encoder embeds the fact description into the original fact representation H_o. Sentence-level attention creates auxiliary representation I: attentive charges aggregator is iteratively to identify related charges that are then aggregated to generate H_s. On top of identified charges, word-level attention creates auxiliary representation II: each word in a fact description is represented by the combination of the terms in related charge definitions. The combined intermediate representations are aggregated through a GRU to generate H_w. At last, H_o, H_s and H_w are concatenated to form final fact representation H for prediction.

3 The Proposed Model

3.1 Problem Formulation

Charge prediction is to predict the corresponding charges l for a given fact description d, where fact description d consists of a sequence of words $\{w_1^d, w_2^d, \cdots, w_m^d\}$, and its label is a C dimensional multi-hot vector – a fact description may correspond to one or multiple labels in C classes. The charge definition for the i-th label l_i is a sequence of words $\{w_1^{l_i}, w_2^{l_i}, \cdots, w_{n_i}^{l_i}\}$.

3.2 Framework

To generate a robust fact representation for prediction, we propose a label definitions augmented interaction model integrating sentence- and word-level attention. The architecture is shown in Fig. 2. The final fact representation H is the concatenation of three representations: 1) the original fact representation (H_o), 2) the auxiliary representation I, charge-related fact representation (H_s), 3) the auxiliary representation II, charge-token-related fact representation (H_w).

3.3 Fact Description Encoder

Giving a fact description with a sequence of word embeddings, we use Gated Recurrent Unite [3] to encode contextual information of each word.

$$h_i = GRU(w_i^d, h_{i-1}), \tag{1}$$

where h_i is the hidden state of the GRU at time step i.

For a fact description, the words and consequently those hidden variables do not contribute equally to convey the semantic meaning of a text, and long fact description will involve many less informative words. To suppress the negative impact of the non-informative words, we use attention mechanism to assign each hidden state an importance weight α_i.

$$\alpha_i = \mathit{softmax}(W_2 \mathit{tanh}(W_1 {h_i}^T)), \tag{2}$$

where $\alpha_i \in [0, 1]$ is the weight of h_i and $\sum_i \alpha_i = 1$. W_1 and W_2 are trainable parameters. The holistic representation of original fact description H_o is computed as a weighted sum of those hidden variables:

$$H_o = \sum_{i=1}^{m} \alpha_i h_i. \tag{3}$$

3.4 Charge Definitions Encoder

Each class label l_i is associated with a charge definition. For each charge definition, we use the same CNN [9] to encode the sequence of n words into a sequence of vectors. Since we will deal with a large number of labels, using CNN gives us better training efficiency than using GRUs.

$$e_j^{l_i} = \mathbf{CNN}(w_{j-\frac{s-1}{2}}^{l_i}, \cdots, w_{j+\frac{s-1}{2}}^{l_i}), \tag{4}$$

where the window size of CNN is s. Then we sum up these vectors to create the holistic representation of each charge definition.

$$L_i = \sum_{j=1}^{n_i} e_j^{l_i}. \tag{5}$$

3.5 Two Auxiliary Fact Representations from Charge Definitions

The first auxiliary fact representation is created through the sentence-level attention between the fact description and charge definitions. Its creation process iterates between two steps: identifying related charges and attentively aggregating the holistic representation of related charge definitions. After those iterations, relatedness weights of each charge will be obtained and they will also be used as the basis for creating the second auxiliary fact representation. The second auxiliary fact representation is generated from word-level attention, which aligns terms in charge definitions with the expressions in the fact description and aggregates those terms through a recurrent neural network. We elaborate the creation of those two auxiliary representations as follows.

Auxiliary Representation I: Charge-Related Fact Representation Created via Sentence-Level Attention Related Charges Identification. Identifying related charges is realized by calculating an attention weight for each charge to indicate the relatedness. Specifically, we exploit episodic memory attention mechanism [24] to iteratively calculate the attention weight from the correlation between the charge definitions and fact description and memory m_t, where m_t can be seen as the summary of already identified charges up to the t-th iteration and will be updated at each iteration. With more iterations, the unrelated charges can be filtered out. The memory m_t is initialized with original holistic representation of fact description, that is, $m_0 = H_o$.

Formally, we use following formulas to calculate the attention weight g of each charge definition at the t-th iteration.

$$z_i = [L_i \circ H_o; L_i \circ m_t; |L_i - H_o|; |L_i - m_t|], \tag{6}$$

$$g_i(t) = softmax(W_2^a tanh(W_1^a z_i)), \tag{7}$$

where $\circ$ is the element-wise product, $|.|$ is the element-wise absolute value, and; represents concatenation of the vectors. W_1^a and W_2^a are trainable parameters.

Attentive Charge Aggregator. Once the attention weight of each charge is calculated, we update the memory by performing weighted summation over charge definition representations.

$$m_{t+1} = \sum_{i=1}^{C} g_i(t) L_i. \tag{8}$$

Finally, we concatenate original fact representation with the last memory and the previous memory, and feed them into a fully-connected layer to create charge-related fact representation by using the following equation:

$$H_s = fc([H_o; m_T; m_{T-1}]), \tag{9}$$

where fc denotes the fully connected layer.

Auxiliary Representation II: Charge-Token-Related Fact Representation Created via Word-Level Attention
In the course of creating the above representation, both fact description and charge definitions are represented by holistic feature vectors. In other words, the interaction between fact and charge definitions is only at the sentence level. The second auxiliary representation steps further introducing interaction at the word level. Specifically, for each hidden variable h_k in the fact description, we first compute its matching score towards each word $e_j^{l_i}$ in each charge definition l_i by inner-product. Then $e_j^{l_i}$ is attentively aggregated to an intermediate representation $h_k^{l_i}$:

$$\beta_j = softmax(h_k \cdot {e_j^{l_i}}^T), \tag{10}$$

$$h_k^{l_i} = \sum_{j=1}^{n_i} \beta_j e_j^{l_i}. \tag{11}$$

The above intermediate representation is defined w.r.t. each charge definition l_i. In our method, we further perform a weighted summation over $h_k^{l_i}$ for different charge definition l_i. The weight is the attention weight $g_i(T)$ calculated at the last iteration T in Eq. (7). Using this weight fits our intuition that the terms in the related charges are more relevant to the expressions in the fact description.

$$h_k^L = \sum_{i=1}^{C} g_i(T) h_k^{l_i}. \tag{12}$$

Note that h_k^L can be viewed as a projection of h_k in the space spanned by $e_j^{l_i}$.

After obtaining h_k^L for each word in the fact description, we process the sequence by a new GRU and obtain the last hidden state $\bar{h}_l$:

$$\bar{h}_t = GRU(h_t^L, \bar{h}_{t-1}). \tag{13}$$

We concatenate original and the projected fact representation, and feed them into a fully-connected layer to generate charge-token-related fact representation.

$$H_w = fc([H_o; \bar{h}_l]). \tag{14}$$

3.6 The Output

Finally, we concatenate all the generated representations and feed them into a fully-connected layer to generate the final fact representation H.

$$H = fc([H_o; H_s; H_w]). \tag{15}$$

Since the evaluated tasks are multi-label problem, we input H into a linear classifier layer with sigmoid activation function to predict the probability, p_{il}, of each labels. The loss function for training is as follows:

$$Loss = -\frac{1}{N} \sum_{i=1}^{N} \sum_{l=1}^{C} [y_{il} log(p_{il}) + (1 - y_{il}) log(1 - p_{il})], \tag{16}$$

where N is the number of training data, C is the number of labels. $y_{il} \in \{0, 1\}$ is the original output of l-th class for i-th training sample and p_{il} is the estimated likelihood of the l-th label being true.

Table 1. Statistics of datasets.

Datasets	Training samples	Validation samples	Test samples	Charge classes
CAIL150k	154592	17131	32500	202
CAIL30k	32506	17131	32500	168

4 Experiments

4.1 Datasets

Table 1 shows the statistics of our used datasets. We use publicly available datasets of the Chinese AI and Law challenge (CAIL2018)[1][23]: `CAIL150k` dataset and `CAIL30k` dataset. `CAIL150k` and `CAIL30k` are different scales with 150,000 and 30,000 training samples respectively[2]. It is worth noting that in these two datasets the distribution of charges is quite imbalanced. In `CAIL150k`, the 31% charges in the training set have less than 100 cases, taking up only 1.88% of the total number of cases. In `CAIL30k`, 42% charges have less than 10 cases, taking up only 0.89% of the total number of cases.

As for charge definitions, they are extracted from articles in the Criminal Law of the People's Republic of China. Specifically, in criminal law, except for articles irrelevant to specific charges, each article may include more than one charges, their corresponding charge definitions, and punishment. We merge the charge definitions scattered in multiple articles. A snippet of cases and charge definitions is illustrated in Fig. 1.

Evaluation Metrics. We employ accuracy (Acc.), macro-precision (MP), macro-recall (MR) and macro-F1 (MF1) as evaluation metrics. Macro-precision/recall/F1 are calculated by averaging the precision, recall, and F1 of each class, which are metrics commonly used for multi-label classification tasks. The experimental results on test set use the parameters providing the best validation performance.

4.2 Training Setup

As all the sentences in charge definitions and fact descriptions are written in Chinese without word segmenting, we apply jieba[3] for word cut. We set the maximum length of fact description to 500, charge definitions to 110. We use pre-trained GloVe [5] vectors as our initial word embeddings. In practice, we choose the 64 dimensional embedding vectors trained on `baidubaike`. The iteration time

[1] http://cail.cipsc.org.cn/index.html.

[2] In CAIL2018 dataset, CAIL150k is ./exercise_contest/data_train.json. CAIL30k is ./final_test.json. They share the same validation and test set (./exercise_contest/data_valid.json and data_test.json).

[3] https://github.com/fxsjy/jieba.

T in Eq. (12) is set as 3. Adam [10] is used as the optimizer and the learning rate is initialized as 0.005 and halved in every other epoch. The epoch size is 20.

4.3 Baselines

We compare our model against several text classification models and existing charge prediction methods, where we only consider the methods with no feature engineering. They can be categorized into four categories:

- **Not using charge definitions for classification.** We implement deep learning models, such as multi-layers Convolution Neural Network (**CNN_classify**) [9], Gated Recurrent Unite (**GRU_classify**) [3] and **BERT** [4] for fact representation learning and classification.
- **Matching the fact representation with charge definitions for classification.** We train a **Siamese CNN** [11] to match the representations of fact description and charge definitions to find the best matched labels.
- **Augmenting fact description with charge definitions for classification.** We implement **Fact-Law AN** [16] that uses relevant law articles, selected by SVMs, to serve as a legal basis for encoding the fact description. To demonstrate the advantage of our model in considering sentence- and word-level interaction jointly, we implement improved memory network (**MemNet**) [12] and **GA_Reader** [21], which employ multi-iterative interaction between query and document at sentence- and word-level respectively for question-answer task.
- **Using multi-task learning for classification.** We re-implement existing charge prediction models **TopJudge** [27] and **LegalAtt** [2], which introduce related legal tasks to train a better fact representation in multi-task mode.

4.4 Results

Experimental results on two scale datasets are shown in Table 2. The observations are as followings:

- Generally speaking, models without incorporating charge definitions (**CNN_classify**, **GRU_classify**) perform inferior to their charge-definition-incorporated counterparts. **BERT** works better due to its strong pre-trained model. This observation clearly demonstrates the benefit of introducing label definitions.
- Incorporating charge definitions through matching based approaches (**Siamese CNN**) works to some extent, although their performance is still worse than methods using more sophisticated interaction between fact description and charge definitions, such as **MemNet** and **GA_Reader**.
- Methods that augment fact representation with charge definitions through end-to-end schema (**GA_Reader**, **MemNet** and **Ours**) attain better results than **Fact-Law AN**. In addition, compared with **GA_Reader** and **MemNet**, which perform either sentence- or word-level interaction, our approach achieves better performance through considering sentence- and word-level interaction jointly.

Table 2. The experimental results [%] of baselines and our model on two datasets. Four different types of models are separated by lines and the best scores are highlight in bold font. The results are averaged over 5 runs.

Datasets		CAIL150k				CAIL30k			
Model		Acc.	MP	MR	MF1	Acc.	MP	MR	MF1
i	**CNN_classify**	79.23	70.80	62.27	64.97	52.75	23.64	21.95	20.59
	GRU_classify	77.33	72.45	57.42	61.54	56.14	23.99	22.81	21.51
	BERT	77.83	75.43	65.29	67.45	57.92	32.29	30.11	30.25
ii	**Siamese CNN**	72.98	74.52	64.64	66.55	50.66	32.74	33.74	29.28
iv	**TopJudge** [27]	78.56	78.92	58.46	65.32	25.26	25.78	24.32	25.55
	LegalAtt [2]	70.30	76.43	59.48	65.08	51.55	39.81	24.34	26.92
iii	**Fact-Law AN** [16]	75.61	58.89	52.30	53.62	60.73	28.15	25.16	24.79
	GA_Reader	73.78	74.68	66.59	68.21	54.95	39.29	34.05	33.03
	MemNet	80.18	80.09	67.13	70.78	62.40	32.62	27.54	27.64
	Ours	**81.05**	**82.06**	**68.33**	**72.43**	**67.99**	**46.13**	**36.00**	**37.62**

- Our proposed model outperforms other baselines on two datasets. The improvement is especially significant on the CAIL30k dataset: our method surpasses the second best about 4.5% in MF1. Since the CAIL30k contains more classes with few training samples, the excellent performance of our approach suggests that our auxiliary representations may help to improve the generalization performance for classes with few samples.
- Existing legal models **TopJudge** and **LegalAtt** introduce multiple related tasks and articles for representation training. Although they can improve the performance of charge prediction, **Ours** using charge definitions to relieve the intra-class variance achieves superior performance.

4.5 Ablation Test

We consider several variations of our approach by removing some components of our model to verify the effectiveness of various components in our method. The result is shown in Table 3. As seen, only using fact descriptions without any level auxiliary fact representations (*w/o Hs,Hw*) yields the worst performance, which proves the importance of the use of charge definitions. After adding either the sentence-level (*w/o Hw*) or the word-level auxiliary fact representation (*w/o Hs*), the performance can be significantly improved. We also created a variant of our method without using attention weight g_i of each charge in Eq. (12) in the process of generating charge-token-related fact representation (*w/o Hs,*g_i), which is implemented by setting the attention weight g_i to $\frac{1}{C}$ instead of generated from charge identification part. It can be observed that the performance of *w/o Hs,*g_i declines. This suggests that the two-level interaction is necessary and using them jointly can get the best performance. The little difference between *Ours* and the auxiliary representation only *w/o Ho* shows the importance of original fact representation since it contains original information about the fact description.

Table 3. The experimental results of ablation test of our model on CAIL150k dataset.

Models	Acc.	MP	MR	MF1
Ours	**81.05**	**82.06**	**68.33**	**72.43**
w/o Hs,Hw	77.33	72.45	57.42	61.54
w/o Hw	79.50	78.86	66.18	69.86
w/o Hs	80.62	80.54	66.97	71.28
w/o Hs,g_i	80.54	76.90	64.34	67.98
w/o Ho	80.31	79.12	66.88	70.55

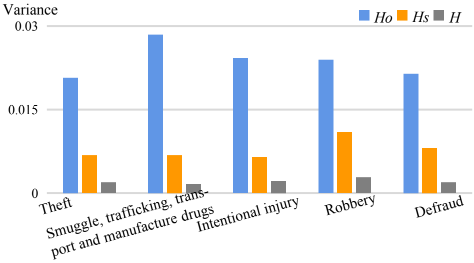


Fig. 3. Intra-class variance of different fact representations of the top-5 frequent classes in CAIL150k dataset. Ho is fact representation only learned from fact description, Hs is the Ho augmented with charge-related fact representation, and H is the Ho augmented with all auxiliary fact representations.

4.6 Intra-class Variance of Different Fact Representations

To investigate whether the fact representation of our method is more stable, we conduct the following experiment: we calculate the variance along each dimension of fact representations from five classes with the most amount of samples, and then use the average variance along all dimensions as an indicator of the intra-class variance of different fact representations. As shown in Fig. 3, fact representation (Ho) only learned from fact description yields the largest intra-class variance. After augmenting fact representation from charge definitions through sentence-level attention (Hs), the intra-class variance declines greatly. Specially, the final fact representation (H) with two auxiliary representations incorporated attains an even lower intra-class variance.

4.7 Case Study

Finally, we select a representative robbery case to give an intuitive illustration of the attention results on the sentence- and word-level interaction. As shown in Fig. 4(a), the case describes that the defendant is convicted of robbery due to stealing property and poking the victim to resist arrest. On the sentence-level interaction, with the increasing of iteration in Eq. (7), our model narrows

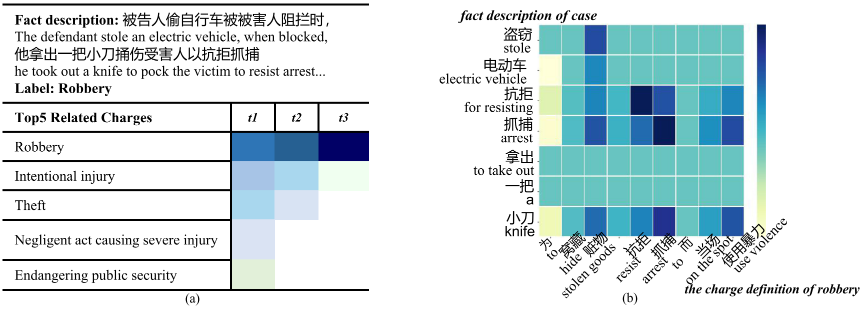


Fig. 4. Attention results of our method for a robbery charge prediction in CAIL150K (in Chinese). The left figure (a) is attention map of sentence-level interaction between fact and charge definitions. t1, t2, and t3 represent the iteration times in Eq. (7). The color darker means the charges are more related to the fact. The right figure (b) is attention map of word-level interaction between fact description and the robbery charge definition. The dark color means a large value.

down the candidate charges and finally identifies the correct related charges. We choose the iteration times as 3 since the performance cannot improve with more iterations. On the word-level interaction, the attention mechanism makes the words in fact description align with the formal terms in charge definitions. Figure 4(b) shows for the words in fact description, which terms are focused on in the charge definition of robbery. The identified keywords in fact description are “electric vehicle”, “resisting arrest” and “a knife”, which correspond to key terms in robbery definition–“stolen goods”, “resist arrest” and “use violence”.

5 Conclusion

In this work, we focus on the task of multi-label charge prediction for given fact descriptions of cases. To address the problem of having a large expression variance in fact descriptions due to informal language use, we introduce charge definitions to create auxiliary representations of the fact descriptions by proposed label definitions augmented interaction model. The experimental results on two datasets show the effectiveness of our model on charge prediction. The significant improvement on the dataset with few training data validate that our method can benefit the small sample training scenario and the two-level auxiliary fact representations can help the model to generalize to the unseen description.

Acknowledgments. This work was supported by National Key R&D Program of China (2018YFC0831302), National Natural Sciences Foundation of China (61972386), and Youth Innovation Promotion Association at Chinese Academy of Sciences.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)

2. Bao, Q., Zan, H., Gong, P., Chen, J., Xiao, Y.: Charge prediction with legal attention. In: Tang, J., Kan, M.-Y., Zhao, D., Li, S., Zan, H. (eds.) NLPCC 2019. LNCS (LNAI), vol. 11838, pp. 447–458. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32233-5_35
3. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
5. Dong, L., Wei, F., Tan, C., Tang, D., Zhou, M., Xu, K.: Adaptive recursive neural network for target-dependent twitter sentiment classification. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (vol. 2: Short papers). vol. 2, pp. 49–54 (2014)
6. Ebesu, T., Shen, B., Fang, Y.: Collaborative memory network for recommendation systems. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 515–524. ACM (2018)
7. Gao, T., Han, X., Liu, Z., Sun, M.: Hybrid attention-based prototypical networks for noisy few-shot relation classification (2019)
8. Hu, Z., Li, X., Tu, C., Liu, Z., Sun, M.: Few-shot charge prediction with discriminative legal attributes. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 487–498 (2018)
9. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
11. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: ICML Deep Learning Workshop, vol. 2 (2015)
12. Kumar, A., et al.: Ask me anything: Dynamic memory networks for natural language processing. In: International Conference on Machine Learning, pp. 1378–1387 (2016)
13. Lin, W.C., Kuo, T.T., Chang, T.J., Yen, C.A., Chen, C.J., Lin, S.D.: Exploiting machine learning models for Chinese legal documents labeling, case classification, and sentencing prediction. In: Processdings of ROCLING, p. 140 (2012)
14. Liu, C.-L., Hsieh, C.-D.: Exploring phrase-based classification of judicial documents for criminal charges in Chinese. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS (LNAI), vol. 4203, pp. 681–690. Springer, Heidelberg (2006). https://doi.org/10.1007/11875604_75
15. Liu, C.-L., Liao, T.-M.: Classifying criminal charges in Chinese for web-based legal services. In: Zhang, Y., Tanaka, K., Yu, J.X., Wang, S., Li, M. (eds.) APWeb 2005. LNCS, vol. 3399, pp. 64–75. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31849-1_8
16. Luo, B., Feng, Y., Xu, J., Zhang, X., Zhao, D.: Learning to predict charges for criminal cases with legal basis. arXiv preprint arXiv:1707.09168 (2017)
17. Sinha, K., Dong, Y., Cheung, J.C.K., Ruths, D.: A hierarchical neural attention-based text classifier. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 817–823 (2018)
18. Sulea, O.M., Zampieri, M., Malmasi, S., Vela, M., Dinu, L.P., van Genabith, J.: Exploring the use of text classification in the legal domain. arXiv preprint arXiv:1710.09306 (2017)
19. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)

20. Wang, S., Mazumder, S., Liu, B., Zhou, M., Chang, Y.: Target-sensitive memory networks for aspect sentiment classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers), pp. 957–967 (2018)
21. Wang, W., Yang, N., Wei, F., Chang, B., Zhou, M.: Gated self-matching networks for reading comprehension and question answering. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers), pp. 189–198 (2017)
22. Weston, J., Chopra, S., Bordes, A.: Memory networks. arXiv preprint arXiv:1410.3916 (2014)
23. Xiao, C., et al.: Cail 2018: A large-scale legal dataset for judgment prediction. arXiv preprint arXiv:1807.02478 (2018)
24. Xiong, C., Merity, S., Socher, R.: Dynamic memory networks for visual and textual question answering. In: International Conference on Machine Learning, pp. 2397–2406 (2016)
25. Yang, W., Jia, W., Zhou, X., Luo, Y.: Legal judgment prediction via multi-perspective bi-feedback network. arXiv preprint arXiv:1905.03969 (2019)
26. Ye, H., Jiang, X., Luo, Z., Chao, W.: Interpretable charge predictions for criminal cases: Learning to generate court views from fact descriptions. arXiv preprint arXiv:1802.08504 (2018)
27. Zhong, H., Zhipeng, G., Tu, C., Xiao, C., Liu, Z., Sun, M.: Legal judgment prediction via topological learning. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3540–3549 (2018)

A Study of Distributed Representations for Figures of Research Articles

Saar Kuzi(✉) and ChengXiang Zhai

University of Illinois at Urbana-Champaign, Urbana, Illinois, USA
{skuzi2,czhai}@illinois.edu

Abstract. Figures of research articles are entities that can be directly used in many application systems to assist researchers, making the representation of figures a problem worth studying. In this paper, we study the effectiveness of distributed representations, learned using deep neural networks, for figures. We learn representations using both text and image data and compare different model architectures and loss functions for the task. Furthermore, to overcome the lack of training data for the task, we propose and study a novel weak supervision approach for learning embedding vectors and show that it is more effective than using some of the pre-trained neural models as suggested by recent works. Experimental results using figures from the ACL Anthology show that distributed representations for research figures can be more effective than the previously studied bag-of-words representations. Yet, combining the two approaches can further improve performance. Finally, the results also show that these representations, while effective in general, can be sensitive to the learning approach used and that using both image data and text and a simple model architecture is the most effective approach.

1 Introduction

Figures are entities in research articles that play an essential role in scientific communications. To accelerate research, it is important to develop tools to assist researchers in accessing and digesting figures. Figure representation is a fundamental problem in all applications involving figures. Different from general images, figures are complex research entities that are associated with various sources of data of various modalities, posing unique novel challenges for representation learning. Thus, the study of how to optimize representation specifically for research figures is crucial. Despite that, this problem has not been well studied in previous works. The dominant approach explored in the existing studies is to represent a figure by its companion text data in an article using the bag-of-words approach [15]. Using this representation of figures has some limitations. First, it does not consider any other types of non-textual features, such as image features. Second, it has limited capability in accommodating the inexact matching of semantically related words.

To address the limitations of the previous work, we study a new view of representation for figures, namely deep neural network-based distributed representations. Learning distributed representations for many real-world entities has

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 284–297, 2021.
https://doi.org/10.1007/978-3-030-72113-8_19

been shown to be very successful in recent years [19,25]. The main idea behind the different approaches in this scope is to learn embeddings of those entities using large data sets where the goal of learning is to capture the complex relations between the entities. For example, learning an embedding representation of words [6,19] has proven to be useful for many text applications. Specifically, word embeddings can effectively address some of the limitations of bag-of-words representations, such as measuring the semantic similarity between words.

In this paper, our goal is to study the effectiveness of distributed representations for figures, exploring the learning of such a representation from multiple views. Specifically, we focus on using both image data and text for learning representations with different model architectures and loss functions to understand how sensitive the embeddings are to the learning approach and the features used.

One technical challenge in learning deep neural network-based representations is that it requires massive amounts of data that is not available for this domain. While word embeddings can be easily learned by leveraging the co-occurrences of words in large amounts of text data, the amount of figure data is quite limited. To overcome this problem, we propose and study two strategies. The first is to leverage massively pre-trained models on general data (e.g., BERT [6]). The second is a novel weak supervision approach that can generate a large amount of training data by leveraging the already existing citation relations between research articles.

We used a collection of figures from the ACL Anthology to empirically study the effectiveness of different representations by their ability to measure the semantic similarity between research figures. We also study the effectiveness of embeddings in the downstream application of recommending figures of interest based on an input (query) figure. The results show that embeddings are generally more effective than bag-of-words, yet combining them is the best performing approach. Another finding is that the pre-trained image/text embeddings have limited effectiveness compared to the weak supervision approach and even the bag-of-words approach. Finally, the results show that the effectiveness of embeddings for figures can be somewhat sensitive to the learning technique. Specifically, the relatively simple model architectures are the most effective ones, text features are more effective than image features, and combining image and text features is the most successful approach.

2 Related Work

There has been growing interest recently in learning vector representations of real-world entities using deep neural networks. This led to the development and study of various embedding models for representing different entities such as words [6,19], sentences [17], and images [22]. Our work can be regarded as the first one to study the effectiveness of embedding-based representations for figures.

Learning embeddings using neural networks often requires massive amounts of data. To address this, there has been an active research direction exploring the use of weak supervision for learning [3,5]. Our work adds to the existing work a new line of application of weak supervision for learning figure embeddings.

There have been several previous works that studied various figure retrieval and mining tasks [2,10,13,16,18]. These previous works mostly relied on the bag-of-words representation of figures. In this work, we explore distributed representations of figures that can benefit a variety of tasks that involve figures.

Previous works have studied the joint embedding of images and text, focusing mostly on images that contain different objects and text that identifies the objects and the interactions between them (e.g., "An apple on a table") [7,8,14,21,23,25]. The main idea in many of these works was to embed image and text to the same space. Learning joint embeddings for image and text aims to find a common representation that can explain both and is thus less appropriate for research figures in which image and text are often two types of complementary information. Thus, in this paper, we learn text- and image-based features separately and combine them using a third model. Using this strategy is sufficient for studying the different aspects of the problem that we are interested in, such as the effectiveness of various architectures for image/text modeling, the effectiveness of image and text feature combination, and the effectiveness of pre-training vs. weak supervision. We thus leave the study on finding the optimal integration of image and text features for figures for future work.

3 Figure Embeddings

Problem Definition: A collection of figures F_D can be generated using a collection of research articles D by extracting the figures from all articles. Each figure can be associated with different types of data of different modalities. For example, a figure can be associated with a caption, the abstract section of its article, an image, and a set of numbers. In this study, as a first step, we focus on learning figure embeddings using only text and image data. Given two figures in the collection, f_i and f_j, the goal is to learn corresponding vectors in a continuous space, f_i and f_j, such that the distance between them in that space is inversely proportional to their semantic similarity. In this paper, we use neural networks to learn these representations of figures.

Textual Representation of Figures: While the image data of a figure is well defined, the textual data for a figure is not readily available. In the general case, the article that contains the figure can be used to extract text that directly describes it (e.g., the figure caption) and text that does not directly describe it, but is related to its topic (e.g., parts of the abstract section). One previous work [15] has explored the effectiveness of using different types of textual data for figure representation to be used for the figure retrieval task. Based on the findings of that work, we generate a textual representation for a figure as follows. We use the caption of the figure, concatenated together with the text in the article that directly describes the figure, for the figure representation. To extract this text, first, the locations in the article where the figure is mentioned are identified. Then, the sentence that directly mentions the figure, one sentence before it, and one sentence after it, are extracted. (In the case of several mentions for the figure, all the text which was extracted is merged.) We use this text as a single textual input which resulted in a good enough performance. In future work, we plan to take into account the sources of those different texts in the learning approach.

Model Architecture: To learn figure embeddings using neural networks, we use the Siamese architecture [4]. According to this architecture, given two figures, the same model is used to generate embeddings for both of them. Then, the dot product between the figure vectors is used as a semantic similarity score. The Siamese model is appropriate for our scenario since the two figures are entities of the same type and we also assume the relationship between them is symmetric. We note that the symmetry assumption may not always hold but is still useful to learn meaningful representations; we thus leave the treatment of asymmetric relationships for future work. The model for our figure embedding approach is composed of three sub-models: (1) An image model that generates visual features. (2) A text model that generates textual features. (3) A fusion model that combines the image and text features. While the image and the text model are both Siamese models, the fusion model is a feed-forward neural network model.

Text Models: To generate textual features, we experimented with three models to explore varying levels of complexity, compare auto-regressive to non-auto-regressive models, and compare pre-training to weak supervision-based training. The first model we used is LSTM [11] that generates features for a text using a recurrent neural network. Specifically, our LSTM-based model contains a word embedding layer (learned from scratch) which is followed by a single LSTM layer, where the weights of the last hidden state of the LSTM layer are used as the textual features. The second model we use is Bi-LSTM [9]. This model is similar to LSTM but has a higher level of complexity since it models dependencies using both directions of the text. As in the case of the LSTM model, we use a word embedding layer which is followed by the Bi-LSTM layer. Additionally, the Bi-LSTM layer generates two sets of features (backward and forward). The two sets of features are concatenated, a dropout layer is added on top of this concatenation, and a final dense layer is added to obtain the textual features. The last model we use is BERT [6] which uses transformers and self-attention mechanisms to learn dependencies in text. This model was shown to achieve state-of-the-art performance in many NLP tasks, where the main approach that was taken is to pre-train the model using a very large amount of text data and then fine-tune the output of the model for the specific task. We experiment with three versions of this model. In the first one, we use a pre-trained model and treat the pooled output as the textual features. In the second version, we add a dropout layer, a dense layer with a Relu activation, and a final dense layer on top of the pooled output. Then, we learn the weights of those dense layers using the Siamese architecture; the output of the final dense layer serves as the textual features. In the third version, we use the same model as in the second one but also fine-tune the last layer of BERT.

Image Models: Previous works on using neural networks for computer vision leveraged massive amounts of data which enabled the learning of complex models with remarkable performance. Another technique that is highly effective for computer vision is transfer learning in which a model is trained using large amounts of data and then is fine-tuned for a specific task. In this work, our goal is to generate effective image features for figures. This is challenging, however, since we do not have available massive amounts of image training data. Furthermore, since images of figures are quite different than images in the massive training data sets (e.g., ImageNet), it is not clear how pre-training will be useful for our scenario. To better understand these issues, we experiment with two models as follows. The first model that we use is a simple Convolutional Neural Network Model (CNN) which is fully trained using the figure image data. The model is composed of two convolutional layers, a max-pooling layer, a dropout layer, a dense layer with Relu activation, and a final dense layer. The second model we use is DenseNet [12] which uses densely connected convolutional networks. This model has higher complexity than the simple CNN and we use it since it was previously shown to be very effective for image representation. We use three versions of this model. In the first one, we use a pre-trained model with ImageNet to generate image features (no fine-tuning). In the second version, we add layers on top of the DenseNet model including a dropout layer, a dense layer with Relu activation, and a final dense layer. We then learn the parameters of the dense layers using the Siamese model. In the third one, we use the same architecture as in the second version but additionally fine-tune the last dense block of the DenseNet model.

Fusion Model: To combine the image and text features, we concatenate them and use a batch normalization layer on top of that. Finally, we use a single dense layer to generate the figure embedding. We take this approach since we are interested in obtaining a single embedding vector for a figure using different types of features.

Loss Function: We assume that each pair of figures, f_i and f_j, is associated with a numeric semantic similarity score $R_{i,j} \in \mathbb{R}$ (larger values of $R_{i,j}$ correspond to greater similarity). A semantic similarity label $L_{i,j} \in \{0, 1\}$ can be generated using $R_{i,j}$ by setting $L_{i,j}$ to 1 when $R_{i,j} > 0$ and setting $L_{i,j}$ to 0 otherwise. We experiment with three loss functions. The first one is the Cross-Entropy loss, computed using the Sigmoid of the dot product between the two vectors and the semantic similarity label, $CE(f_i \cdot f_j, L_{i,j})$. Secondly, we use the Mean Squared Error loss, computed using the dot product between the two vectors and the semantic similarity score, $MSE(f_i \cdot f_j, R_{i,j})$. Finally, we use the triplet hinge loss [22]. The triplet hinge loss is defined for a triplet of figures, comprised of a query figure f_q, a positive figure f_+ (i.e., a related figure), and a negative figure (f_-). This loss is defined as: $max(0, 1 + f_q \cdot f_- - f_q \cdot f_+)$. The main idea is that we want a figure to be closer to a related figure compared to an unrelated figure.

4 Weak Supervision for Figure Embeddings

Since we are dealing with a novel problem in this paper, an important issue that needs to be addressed is how to collect training data. Furthermore, since we are interested in using deep neural networks, there is a need for a large set of training examples. To address this challenge, since log data was not available to us, we propose a novel approach for collecting data for the task using weak supervision.[1] This approach allows us to leverage large amounts of training data that already exist. Specifically, to generate training data, we leverage existing relations between research articles. First, since we know that two articles are related if one is cited by the other, we assume that two figures are semantically similar if they appear in two articles with a citation relation. Secondly, we assume that any two figures that are in the same article are also semantically similar. Although both kinds of relations may be noisy, we expect that most relations are meaningful semantic associations and the learned embedding vectors to be meaningful as in the case of word embeddings where there are also noisy word associations, but they do not significantly affect the results. Comparing the two types of relations, it is reasonable to assume that two figures that appear in the same article are more likely to be more semantically similar than two figures that appear in citing articles and that the latter should be more similar than a random pair of figures. Based on this intuition, we set the semantic similarity score of two figures in citing articles to be lower than the score of two figures in the same paper. Finally, to generate negative examples, we randomly sample pairs of figures from the collection. Formally, given two figures f_i and f_j, extracted form the articles $d(f_i)$ and $d(f_j)$, respectively, and given that $C(d(f_i))$ is the set of articles that cite $d(f_i)$ or are cited by it, the semantic similarity score $R_{i,j}$ is set to 1 if $d(f_i) = d(f_j)$, 0.6 if $d(f_j) \in C(d(f_i))$, and 0 otherwise.

When using this data for training the image model, some modifications need to be made. This is the case since semantically similar figures, as defined by our approach, may have images that are not visually similar. Our goal for the image model is to be able to generate features that can help measure the visual similarity between figures. For this reason, we filter out pairs of figures which are not visually similar enough (we use the Structural Similarity Index (SSIM) [24] with a threshold of 0.5 for filtering out pairs, and a threshold of 0.3 for sampling negative pairs). Finally, we do not make a differentiation between figures in the same paper and figures in citing papers since these relationships may not be indicative of different levels of visual similarity. Taking this approach, a pair of figures will be assigned only with a binary relevance label in the case of the image model (and consequently we do not use the MSE loss).

[1] While there are publicly available large collections of figures [20], they do not provide any relations between figures for the purpose of representation learning.

5 Empirical Study

5.1 Experimental Setup

Collection of Figures: A collection of figures was built using the ACL Anthology (aclweb.org/anthology). 40,367 articles whose copyright belongs to ACL and were published until October 2018 were crawled. Using those articles, a collection of 84,340 figures was created. The PdfFigures toolkit (pdffigures2.allenai.org) was used to extract the figure images. The Grobid toolkit was used to extract the full text from the PDF files of the articles (github.com/kermitt2/grobid).

Data Pre-processing: Text data was Porter stemmed and stopwords were removed (using the INQUERY list). Figures with an associated text, after pre-processing, of less than 5 words were removed. Images of figures were resized to fit a 224 × 224 × 3 matrix and were normalized by a factor of 255.

Training Data: 947,335 pairs of figures in citing articles and 202,944 pairs of figures in the same article were used as related figures. After adding random pairs as negative examples, we ended up having about 2M pairs of figures for training the text network. For the image network, after filtering out images that were not visually similar enough, we ended up with about 300K figure pairs for training. For training the fusion network, since we are interested in figures with both text and image data, we used about 1M pairs after filtering out figures with no image data. In this work, we train all three components of the model (i.e., the image model, text model, and fusion model) separately, due to our limited data. For the evaluation of the different approaches, we only use figures for which both image and text data is available to make it as realistic as possible (57K figures).

Neural Network Implementation: The neural network was implemented using the TensorFlow library. We set the values of the different parameters based on findings in recent works in the text and image domain. All models were trained for 3 epochs using the Adam optimizer with a batch size of 64 and a learning rate of 0.01. The vocabulary size was set to the 1000 most frequent words in the training data. We used only the first 100 words in the text data of a figure (the figure caption was concatenated first) due to BERT's limitation on the input size and the limited effectiveness of LSTM for long sequences. The embedding size was set to 50 for all models which means that the number of hidden layers in LSTM/Bi-LSTM was set to 50 as well as the size of the final dense layer in the other models (our preliminary experiments showed that a larger size of 100 is less effective). The size of the dense layer on top of BERT, DenseNet, and CNN was set to 100. The dropout rate was set to 0.5. The word embedding layer dimension for the LSTM/Bi-LSTM model was set to 100. For BERT, we used a model with 12 layers, 768 hidden units, and 12 attention heads. For DenseNet, we used a 121-layer model. For the CNN model, we used convolutional layers with 32 filters and a kernel size of 3 × 3.

Baselines: Since one of the main research questions that we study is whether embeddings can improve over the currently used bag-of-words representations, we compare our model with two representative baseline methods: tf.idf and LDA. For the LDA baseline [1], we learn a model with 50 topics and use the figure distribution over topics as its representation. The vocabulary used for both models was also restricted to 1000 frequent words.

5.2 Experimental Results

Semantic Similarity Prediction: To evaluate the effectiveness of the different representations in measuring the semantic similarity between figures, a binary classification task was performed. Given two figure vectors, the cosine function was used to get a similarity score which was then transformed into a binary label using a threshold. Since the threshold value can vary depending on the representation type, a validation set was used to set it (selected from $\{0.1, 0.2, ..., 0.9\}$). Three test sets were created for the evaluation. In the first one, denoted "Same", we used 500 pairs of figures that appear in the same article (related figures) and 500 randomly sampled (unrelated) pairs. In the second one, denoted "Citing", we used 500 pairs of figures that appear in citing articles (related figures) and 500 unrelated pairs. Finally, in the third set, denoted "Accuracy", the first two sets were combined. (All selected pairs were removed from the training set.) The sets were balanced such that the accuracy of a random baseline is 0.5. The results are presented in Table 1 for using text and image features separately and in Table 3 for the fusion model. For pre-trained models that were fine-tuned (i.e., BERT and DenseNet), we added the term "(tuned)" when only the dense layers on top of the model were fine-tuned and "(tuned+)" when the dense layers and also part of the model were fine-tuned.

According to the results in Table 1, most text-based and image-based representations perform better than a random baseline. Focusing on the embedding models which use text features only, we can see that for the LSTM/Bi-LSTM model the best performance is achieved for the MSE loss, while for the tuned BERT models there are no large differences between the different loss functions. Overall, based on the results, the best text-based embedding model is LSTM. A possible reason for this might be its relatively small number of parameters and the size of the training data set. Also, it is interesting to see that it outperforms the pre-trained BERT model which might be attributed to the unique vocabulary used in ACL research articles. Comparing the embedding models to the baselines, we can see that LSTM/Bi-LSTM largely outperforms all baselines including tf.idf, LDA, and the pre-trained BERT model. We can also see from the results that tf.idf is the strongest baseline. For this reason, we focus on comparing our embedding approaches mostly to this baseline in the remainder of the evaluation section. Focusing on BERT, we can see that fine-tuning can improve its performance, but resulting in overall effectiveness that is still low. Another finding from the table is that the improvements of the embedding methods over the bag-of-words baselines for the case of citing figures are much larger than the case of figures in the same article. This might be due to the soft matching nature of distributed (dense) representations and their ability to identify more loosely related figures. Moving on to the image features, we can see that most

Table 1. Semantic similarity prediction: text vs. image.

			Accuracy	Same	Citing
Text features		tf.idf	.720	.818	.622
		LDA	.688	.766	.609
		BERT	.525	.522	.527
	CE	LSTM	.740	.776	.704
		Bi-LSTM	.732	.743	.720
		BERT(tuned)	.533	.535	.530
		BERT(tuned+)	.534	.534	.533
	MSE	LSTM	.802	.831	.772
		Bi-LSTM	.791	.811	.770
		BERT(tuned)	.527	.527	.527
		BERT(tuned+)	.527	.528	.525
	Hinge	LSTM	.505	.508	.501
		Bi-LSTM	.500	.500	.500
		BERT(tuned)	.522	.525	.518
		BERT(tuned+)	.534	.537	.530
Image features		DenseNet	.620	.623	.616
	CE	CNN	.500	.500	.500
		DenseNet(tuned)	.635	.641	.629
		DenseNet(tuned+)	.518	.510	.526
	Hinge	CNN	.662	.663	.661
		DenseNet(tuned)	.630	.655	.605
		DenseNet(tuned+)	.500	.500	.499

of them perform better than a random approach and that the best performing model is CNN. Finally, we can see that using fine-tuning for the DenseNet model can improve its performance. Yet, the performance of the fine-tuned DenseNet model is still not as good as that of CNN. Comparing the image to text features we can see that the text features are more effective.

Table 2. Combining tf.idf with text-based embeddings using an "Oracle".

	Accuracy	Same	Citing
tf.idf	.720	.818	.622
LSTM	.802	.831	.772
BERT(tuned+)	.534	.534	.533
LSTM& tf.idf	.914	.941	.886
BERT(tuned+)& tf.idf	.864	.913	.815

In light of the results in Table 1, an important question that comes up is whether embeddings can replace tf.idf for the textual representation of figures. To answer this, we examine the effectiveness of combining the predictions of tf.idf and embeddings using an oracle in Table 2 which serves as an upper-bound for the performance of such combination. We focus on effective models according to Table 1: LSTM trained with MSE and BERT(tuned+) trained with CE. The results show that this combination is of merit, always outperforming the individual models. Even in the case of BERT, which is not very effective according to Table 1, the combination can improve tf.idf substantially. In this paper, we are mainly interested in studying distributed representations and thus leave the study of how to combine the two approaches for future work.

Table 3. Semantic similarity prediction: fusion model.

		Accuracy	Same	Citing
	tf.idf	.720	.818	.622
	LSTM	.802	.831	.772
	BERT(tuned+)	.534	.534	.533
	CNN	.662	.663	.661
	DenseNet(tuned)	.635	.641	.629
CE	LSTM& CNN	.805	.834	.775
	BERT(tuned+)& CNN	.684	.689	.678
	LSTM& DenseNet(tuned)	.643	.681	.604
	BERT(tuned+)& DenseNet(tuned)	.678	.680	.675
MSE	LSTM& CNN	.838	.866	.809
	BERT(tuned+)& CNN	.699	.704	.693
	LSTM& DenseNet(tuned)	.726	.760	.691
	BERT(tuned+)& DenseNet(tuned)	.693	.698	.687

Next, we analyze the performance of representations that combine both image and text data in Table 3. We focus on studying the combination of the most effective image and text features, based on the results in Table 1. Specifically, we use LSTM trained with MSE, BERT(tuned+) with CE, CNN with Hinge loss, and DenseNet(tuned) with CE. We also focus only on MSE and CE due to the very poor performance of the Hinge loss for the textual features. The results show that for the majority of model combinations, using both features largely outperforms the individual components. This finding supports the idea that image and text features are complementary and represent different aspects of the figure. Finally, we can see that the MSE loss is the best performing for all models and that the best performing model is the LSTM&CNN model.

Figure Recommendation: The goal of this task is to recommend figures to the user that are related to a target figure. To address this problem, a standard two-phase approach was used. First, using the target figure, an initial retrieval is performed to get an initial figure set. Then, a re-ranking model is used to obtain the recommended figures. To build a test set of target figures for testing, we first collected all figures that have at least 5 more figures in the same article and also 5 figures in citing articles (to result in $p@5 = 1$ at the best scenario). From this set, 500 figures were selected randomly (400 for testing and 100 for validation); all pairs of figures that contained at least one of the target figures were removed from the training set. The performance of the different models is measured using $p@3$ and $p@5$. Since there are no human relevance judgments available for the task, we assume that a figure is relevant if it appears in the same article as the target figure ("Same"), a citing article ("Citing"), or in either (the main performance measure). We note that while this evaluation is not fully realistic, it can still help us make meaningful comparisons between the different approaches. Statistically significant differences between approaches were measured using the two-tailed paired t-test at a 95% confidence level.

Table 4. Retrieval performance of the recommendation task. All differences with tf.idf are statistically significant.

			Same		Citing	
	$p@3$	$p@5$	$p@3$	$p@5$	$p@3$	$p@5$
tf.idf	.298	.228	.354	.276	.057	.048
LSTM	.044	.032	.058	.047	.014	.016
CNN	.000	.001	.001	.003	.001	.002
LSTM& CNN	.051	.039	.066	.054	.015	.014

Table 5. Figure recommendation performance. Statistically significant differences with tf.idf are marked with an asterisk.

			Same		Citing	
	$p@3$	$p@5$	$p@3$	$p@5$	$p@3$	$p@5$
tf.idf	.298	.228	.354	.276	.057	.048
	Cross Entropy (CE)					
LSTM& CNN	.308	$.241^*$	.368	$.296^*$	.060	$.056^*$
BERT(tuned+)& CNN	.296	.227	.352	.277	.056	.050
LSTM& DenseNet(tuned)	.303	.233	.355	$.289^*$	.053	$.056^*$
BERT(tuned+)& DenseNet(tuned)	.303	.229	.357	.279	.054	.050
	Mean Squared Error (MSE)					
LSTM& CNN	$.320^*$	$.240^*$	$.380^*$	$.299^*$	.060	$.059^*$
BERT(tuned+)& CNN	.296	.226	.353	.277	.057	.052
LSTM& DenseNet(tuned)	$.313^*$	$.235^*$	$.371^*$	$.287^*$	.058	.053
BERT(tuned+)& DenseNet(tuned)	.300	.229	.356	.278	.056	.049

First, we study the effectiveness of the retrieval step in Table 4. The performance of three embedding methods (which use text data, image data, and both), trained using the MSE loss, is compared with that of tf.idf. We can see that the tf.idf approach is the most successful. This result is expected since tf.idf relies mostly on exact keyword matching while embedding-based methods rely more on soft matching. Since we are searching over the entire collection, the embedding model may not be discriminative enough.

The performance of the recommendation task is reported in Table 5. To obtain these results, we first perform retrieval using tf.idf and then re-rank the first 100 figures using the cosine similarity between the figure embeddings. The final score for a figure is defined as a linear interpolation between the tf.idf score and the embedding score. The weight for the tf.idf component and the embedding component in the interpolation is determined using a validation set (selected from $\{0.1, 0.2, ..., 0.9\}$; the weights are set to sum up to 1). We experiment with embedding approaches that use both text and image data with the same setting as in Table 3. According to the results in Table 5, we can see that using embeddings on top of the initial retrieval results (tf.idf based) can largely improve the recommendation performance. Specifically, the embedding approaches outperform the baseline in terms of the overall p@3 and p@5 for the majority of relevant comparisons. Comparing the LSTM model to BERT, we can see that the former is better in the majority of cases. The best embedding model, according to the results, is the LSTM&CNN model with the MSE loss.

Table 6. Figure recommendation example.

LDA graphical representation	
tf.idf	Embeddings
1. The graphical representation of LDA	1. The graphical representation of LDA
2. Graphical models of LDA and DMM	2. Topic model
3. Topic model	3. Graphical representation of strTM
4. Plate notation of our model: MATM	4. Plate notation of our model: MATM
5. LDA plate diagram	5. Graphical representation of (a) BTM, (b) Twitter-BTM

An example target figure with its recommendation list is presented in Table 6. In the table, the caption of the target figure is presented together with the captions of five recommended figures when using either tf.idf or embeddings (LSTM&CNN with MSE); in both cases, tf.idf was used for the initial retrieval. The subject of the figure is the graphical representation of the LDA topic model. Using the tf.idf approach, we get figures that are either equivalent (e.g., “LDA plate diagram”), or diagrams of related models (e.g., “MATM” and “DMM”). When using the embedding approach, we can see that we get more diverse recommendations. This difference can be because using embeddings results in softer matching compared to tf.idf.

6 Conclusions

In this work, we studied the effectiveness of neural network-based figure embeddings. The experimental results showed that figure embeddings outperform the bag-of-words approach in the tasks of semantic similarity prediction and figure recommendation. We also observed that embeddings cannot replace the bag-of-words approach and that combining the two is the best practice. Finally, the results also showed that some learning approaches can be more effective than others. Specifically, using a simple model architecture and combining both image and text features performs the best.

In future work, different methods for combining the different figure features can be studied. Collecting user data to learn more effective representations and improve the evaluation is another possible future direction.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grants No. 1801652 and No. 1937115.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
2. Bockhorst, J.P., Conroy, J.M., Agarwal, S., O'Leary, D.P., Yu, H.: Beyond captions: Linking figures with abstract sentences in biomedical articles. PloS one **7**(7), e39618 (2012)
3. Bordes, A., Weston, J., Usunier, N.: Open question answering with weakly supervised embedding models. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) ECML PKDD 2014. LNCS (LNAI), vol. 8724, pp. 165–180. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44848-9_11
4. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a" siamese" time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
5. Dehghani, M., Zamani, H., Severyn, A., Kamps, J., Croft, W.B.: Neural ranking models with weak supervision. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 65–74. ACM (2017)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Dey, S., Dutta, A., Ghosh, S.K., Valveny, E., Lladós, J., Pal, U.: Learning cross-modal deep embeddings for multi-object image retrieval using text and sketch. arXiv preprint arXiv:1804.10819 (2018)
8. Frome, A., et al.: Devise: a deep visual-semantic embedding model. In: Advances in neural information processing systems, pp. 2121–2129 (2013)
9. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Netw. **18**(5–6), 602–610 (2005)
10. Hearst, M.A., Divoli, A., Guturu, H., Ksikes, A., Nakov, P., Wooldridge, M.A., Ye, J.: Biotext search engine: beyond abstract search. Bioinformatics **23**(16), 2196–2197 (2007)

11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
12. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
13. Kim, D., Ramesh, B.P., Yu, H.: Automatic figure classification in bioscience literature. J. Biomed. Inform. **44**(5), 848–858 (2011)
14. Klein, B., Lev, G., Sadeh, G., Wolf, L.: Associating neural word embeddings with deep image representations using fisher vectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4437–4446 (2015)
15. Kuzi, S., Zhai, C.X.: Figure retrieval from collections of research articles. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11437, pp. 696–710. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15712-8_45
16. Kuzi, S., Zhai, C., Tian, Y., Tang, H.: Figexplorer: a system for retrieval and exploration of figures from collections of research articles. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2133–2136 (2020)
17. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
18. Liu, F., Yu, H.: Learning to rank figures within a biomedical article. PloS one **9**(3), e61567 (2014)
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
20. Siegel, N., Lourie, N., Power, R., Ammar, W.: Extracting scientific figures with distantly supervised neural networks. In: Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries, pp. 223–232 (2018)
21. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: A neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
22. Wang, J., et al.: Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1386–1393 (2014)
23. Wang, L., Li, Y., Lazebnik, S.: Learning deep structure-preserving image-text embeddings. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5005–5013 (2016)
24. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
25. Weston, J., Bengio, S., Usunier, N.: Large scale image annotation: learning to rank with joint word-image embeddings. Mach. Learn. **81**(1), 21–35 (2010)

Answer Sentence Selection Using Local and Global Context in Transformer Models

Ivano Lauriola(✉) and Alessandro Moschitti

Amazon Alexa AI, Manhattan Beach, CA, USA
{lauivano,amosch}@amazon.com

Abstract. An essential task for the design of Question Answering systems is the selection of the sentence containing (or constituting) the answer from documents relevant to the asked question. Previous neural models have experimented with using additional text together with the target sentence to learn a selection function but these methods were not powerful enough to effectively encode contextual information. In this paper, we analyze the role of contextual information for the sentence selection task in Transformer based architectures, leveraging two types of context, local and global. The former describes the paragraph containing the sentence, aiming at solving implicit references, whereas the latter describes the entire document containing the candidate sentence, providing content-based information. The results on three different benchmarks show that the combination of the local and global context in a Transformer model significantly improves the accuracy in Answer Sentence Selection.

Keywords: Question Answering · Answer Sentence Selection · Pre-trained Transformer · Deep learning

1 Introduction

Recent research in Question Answering (QA) mainly addresses two tasks: (i) Answer Sentence Selection (AS2), which, given a question and a set of answer sentence candidates (e.g., retrieved by a search engine), consists in selecting the sentence that correctly answers the question with the highest probability; and (ii) Machine Reading (MR) comprehension [2], which, given a question and a reference text, involves finding an exact text span answering it. AS2 research originated from the TREC competitions [23], which target large databases of unstructured text. It has the advantage of high efficiency, which enables its use in real-world applications, e.g., see the study in [5].

Neural models have significantly contributed to both directions with new techniques [11,13,24]. In particular, recent approaches to neural language models, e.g., ELMO [12], GPT [15], BERT [4], RoBERTa [9], XLNet [3] have led to

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 298–312, 2021.
https://doi.org/10.1007/978-3-030-72113-8_20

Table 1. An example of correct answer sentence requiring larger context to be selected.

Question	**When was Lady Gaga born?**
Prev.	Lady Gaga is an American singer, songwriter, and actress
Target	She was born in 1986
Next	Both of her parents have Italian ancestry, and …

Table 2. Each of the three sentences can be a correct answer. Only the global document information, e.g., the title and the link between document concepts, allows us to select the correct sentence.

Question	**Which role did Bradley Cooper play with Lady Gaga?**
Doc. title	Avengers: endgame - Movie plot
Sentence	Rocket Raccoon was voiced by Bradley Cooper
Doc. title	A star is born - Movie plot
Sentence	Jackson “Jack” Maine (Bradley Cooper), a famous country rock singer …
Doc. title	American sniper - Movie plot
Sentence	Chris Kyle, the leading actor, was played by Bradley Cooper

major advancements in several NLP subfields. These methods capture dependencies between words and their compounds by pre-training neural networks on large amounts of data. Interestingly, the resulting models can be easily applied to different tasks by fine-tuning them on the target training data. The impact of such methods on AS2, also thanks to transfer learning, is impressive. For example, [5] exceeded the state of the art by 50% (relative error reduction) on WikiQA [27] and TREC-QA [23] datasets. Although this result seems hard to improve, we note that most previous work does not exploit contextual information in addition to the candidate sentence with a few exceptions, e.g., [21]. This aspect produces a suboptimal solution as there can be many cases that contain ambiguities, and they cannot be solved without other references or context. Formally, the term *context* refers to additional linguistic information coming from the source of a candidate answer sentence, which can be, for instance, the document containing the sentence, the paragraph, the domain, and so on.

For example, Table 1 shows a simple question asking for the birthdate of *Lady Gaga*. The answer is the middle sentence contained in a paragraph of three sentences. Clearly, an AS2 classifier cannot select the middle sentence with high reliability since the sentence does not reveal that *she* refers to *Lady Gaga*. On the other hand, AS2 is effective as it targets just one sentence at a time: selecting an entire paragraph to be sent to the users, often provides them with too much irrelevant information[1]. A further example is described in Table 2, where the question asks for the role of *Bradley Cooper* in a specific movie. In the same example, we retrieved three sentences belonging to three different documents containing movie plots. Each of the three sentences may reasonably be a correct answer. Also, the title of the movie is not enough to select the right answer and it can be too far from the "local" context window showed in the previous example. However, *"A star is born - Movie plot"* is the only document that contains references to *Lady Gaga*. This related information allows us to recognize the correct answer. The two examples describe two different problems in common QA scenarios. In the first case, the sentence requires a local context to solve the pronoun *she*. Conversely, the candidate requires global information from the whole document to recognize the correct movie in the second example.

It should be noted that (i) previous neural network work, e.g., by [21], used context for AS2 in a hierarchical gated recurrent network but their accuracy is 10–12 points below the state of the art by [5] (as measured on the same exact dataset). Thus, it is not clear if their context is really useful for improving AS2 models. (ii) MR models clearly use a larger context but (a) they are not efficient enough to analyze hundreds of documents for each question, and (b) they target the selection of any subset of the document. This prevents them to be fast and accurate for AS2.

In this paper, we propose to model local and global contexts for AS2 by using multiple sentences and Bag-of-Word (BOW) features in Transformer networks [22]. More specifically, we consider candidates as a triplet (s_{i-1}, s_i, s_{i+1}), where s_i is the target answer sentence and s_{i-1} and s_{i+1} are the preceding and the next sentence of s_i, respectively. We integrate this triplet in Transformer architectures by using one single RoBERTa [9] model encoding the three sentences in three embeddings. Then, we add document-level BOW representation in the classification layer. We tested our models on three different datasets, Google NQ and SQuAD adapted for the AS2 task, as well as the well-known WikiQA, comparing with the very recent state of the art in AS2 [5]. The results clearly show that local and global contexts can improve AS2 models.

2 Related Work

We consider retrieval-based QA systems, which are mainly constituted by (i) a search engine, retrieving top-k documents related to the questions; and (ii) an Answer Sentence Selection (AS2) model, which reranks passages/sentences extracted from the documents. The task of reranking answer sentence candidates provided by a retrieval engine can be modeled with a classifier scoring the candidates.

Recent work has proposed neural networks that apply a series of non-linear transformations to the input question and answer text, represented as

[1] Of course, a solution based on a summarization approach would be optimal but poses complicated challenges, which have prevented to obtain better solutions than AS2 (to our knowledge).

compositions of word or character embeddings; and (ii) then measure the similarity between the obtained representations. Question-to-question and answer-to-answer patterns are typically important to derive if an answer is correct for a question. For example, the CNN by [17] has two separate embedding layers for the question and answer, and a relational embedding, which aims at connecting them. More recent work uses attention mechanism, e.g., Compare-Aggregate [28], inter-weighted alignment networks [19], and pre-trained Transformer models [5].

In particular, the latter has shown to largely outperform any previous approach in AS2: a simply binary classifier is built by adding a linear layer on top of the Transformer architecture, and is fine-tuned with positive and negative answers. The training of such model can be carried out by using a cross-entropy binary loss function. Additionally, the approach was highly boosted using out of domain data, as a first fine-tuning step, followed by a second fine-tuning on the target data. This procedure was referred to as the TANDA model, i.e., transfer the pre-trained models on the task, then adapt it to the target domain.

However, the proposed Transformer methods only focus on the similarity between the question and the candidate sentence pairs, without taking any additional information into account. Contextual information was already introduced in neural networks for solving AS2, e.g., [21], by combining question/answer pairs with context information, selected by applying a similarity between question and document sentences.

MR research has produced state-of-the-art models, e.g., [1,14,25]. By definition, MR is supposed to exploit a larger context than standard AS2 models, as their input is an entire abstract. Transformer models limit the input to 512 tokens, which prevent to encode large documents, e.g., webpages. Thus, we cannot consider them as global models. In contrast, they surely fit the definition of local context. However, as pointed out in the introduction, they are not enough efficient to analyze hundreds of documents for each question, which is a requirement of real-world applications [10,20]. Also, they optimize the selection text sub-sequences, which, is a stronger requirement that does not lead to a better sentence selection model. Indeed, in our experiments, we show that state-of-the-art MR systems used for selecting answers are outperformed by AS2 models.

Differently from previous solutions, our model is built with state-of-the-art Transformer models for AS2. Moreover, our approach is more modular and can be easily extended with additional context definitions. We also improve the results from [21] by a huge margin (+12% on WikiQA and +5% on SQuAD).

3 Transformer Models for Answer Sentence Selecting

AS2 is the task of identifying sentences that contain the answer to a given question. The task can be modeled with a scoring function that outputs a probability of correctness for each question/sentence pair, (q, s_i). Such function can be implemented with a Transformer model as shown in [5]. In the remainder of this section, we formalize the task and describe a state-of-the-art model based on the Transformer.

3.1 AS2 Definition

Let $\mathcal{Q}$ and $\mathcal{S}$ be the sets of questions and sentences, respectively, the AS2 task can be defined as a ranking function $r : \mathcal{Q} \times \mathcal{S} \rightarrow \mathbb{R}$, which assigns a score to each possible question/answer pair, where the higher the score is, the higher the probability of selecting a correct answer candidate is. In other words, we want to learn r, such that for each $q \in \mathcal{Q}$, we select

$$a = \arg \max_{s_i \in \mathcal{S}(q)} r(q, s_i)$$

as the final answer, where $S(q) \subseteq S$ is the set of answer sentence candidates for the input question q. For example, $\mathcal{S}(q)$ can be built by retrieving sentences from text repositories such as the web [5,27]. In this work, we define and develop the ranking function r with Transformer models.

3.2 Selecting Sentences with a Transformer Model

The Transformer is a popular neural network designed to learn language models, e.g., dependencies between words, in a context. Transformer models have recently been shown to produce a remarkable impact on AS2, when used as ranker [5,7,18]. Besides architectural definitions, an important advantage of Transformer models is their ability to be pre-trained on large-scale corpora, using masked language and next sentence prediction tasks [4].

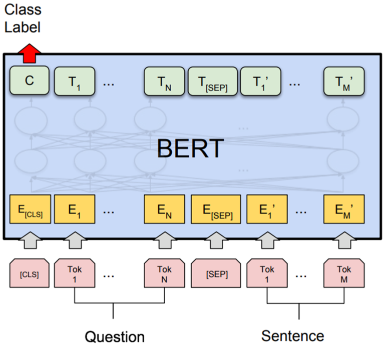


Fig. 1. The q/a pair is codifies as a whole sequence with special delimiters.

More specifically, Fig. 1 shows the approach of using a pre-trained Transformer for AS2. The question and answer candidate pairs are codified as a joint sequence of tokens with specialized delimiters and separators, i.e., *[CLS]* $q^1 \ldots q^n$ *[SEP]* $s^1 \ldots s^m$ *[EOS]*, where x^j defines the j-th token of the sequence x. *[CLS]*, *[SEP]*, and *[EOS]* are special tokens used to mark the beginning of the sequence, the separation between question and candidate answer tokens, and the end of the text, respectively. Several Transformer blocks are applied, and then the representation associated with *[CLS]* is used in a linear fully-connected layer to compute the final score associated with the question/answer pair. The same concepts can be applied to RoBERTa or other pre-trained Transformer models.

4 Contextual Transformer for AS2

To our knowledge, no Transformer model for AS2 uses context, except for the information on the sentences. This is critical as a sentence may contain references to other parts of the text and to external entities (see the example in Table 1). We enhance the standard Transformer model for AS2 with two types of context: local and global. The former aims at resolving the coreferences between the constituents in the candidate answer sentence and its neighborhood sentences (typically part of the paragraph containing the answer). In contrast, the global context introduces information concerning the topics and concepts of the entire document containing the answer sentence.

4.1 Local Context

Given the target answer sentence candidate, s_i, we extend the standard AS2 model using the preceding, s_{i-1}, and the following, s_{i+i}, sentences. The (local) contextual ranker r_{L} takes four elements as input and provides the following answer:

$$a_{\text{L}} = \arg\max_{s_i \in \mathcal{S}(q)} r_{\text{L}}(q, s_{i-1}, s_i, s_{i+1}),$$

where $\mathcal{S}(q)$ is the set of relevant sentences for the question q and r_{L} is our ranking function. To implement r_{L} in the RoBERTa model, the input sequence becomes *[CLS]* q *[SEP]* s_{i-1} *[SEP]* s_i *[SEP]* s_{i+1} *[EOS]*. Additionally, RoBERTa encodes each input word by using three pieces of information: the token, the sentence, and the positional embeddings.

The first is a standard word-embedding. The positional embedding describes a token as a function of its position in the sequence. Finally, the sentence embedding defines a token as a function of the sentence that contains it. The sentence embedding helps the model to distinguish between different input sentences: it can be seen as a particular word embedding of size four, one entry for each element of the input tuple, $(q, s_{i-1}, s_i, s_{i+1})$. This embedding plays a crucial role in our model to learn that the instance label is exclusively associated with the middle sentence. According to the canonical procedure, the three embeddings are then summed to produce the final representation of the sentences to be fed as input to the Transformer. This process is described in Fig. 2 (see dashed squares). When the preceding sentence s_{i-1} is not available, we consider an empty sequence in our input encoding, that is, *[CLS]* q *[SEP] [SEP]* s_i *[SEP]* s_{i+1} *[EOS]*. In this case, the model is still able to recognize the different parts of the input thanks to the sentence embedding and the two consecutive separators. The same strategy holds when the following sentence s_{i+1} is missing. Note that the local context is not limited in co-ref resolution as it also encodes semantic information from the whole sentences.

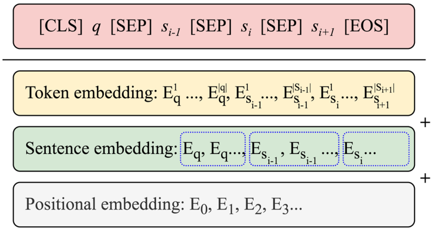

[CLS] q [SEP] s_{i-1} [SEP] s_i [SEP] s_{i+1} [EOS]
Token embedding: E_q^1 ..., $E_q^{|q|}$, $E_{s_{i-1}}^1$ ..., $E_{s_{i-1}}^{|S_{i-1}|}$, $E_{s_i}^1$ ..., $E_{s_{i+1}}^{|S_{i+1}|}$
+
Sentence embedding: E_q, E_q..., $E_{s_{i-1}}$, $E_{s_{i-1}}$ ..., E_{s_i}...
+
Positional embedding: E_0, E_1, E_2, E_3...

Fig. 2. BERT/RoBERTa input sequences.

4.2 Global Context

The local context models the information related to the paragraph containing the candidate answer, and it is helpful to solve coreference problems. However, the local context is small and does not include other important information. Global document-based features can provide additional information to the local context, e.g., the main document topic, which can be used to select the correct sentence. Our global context describes the document content rather than the structure of the paragraph containing the answer. The global ranker r_{G} is defined as

$$a_{\text{G}} = \arg \max_{s_i \in \mathcal{S}(q)} r_{\text{G}}(q, s_i, d(s_i)),$$

where $d(s_i)$ is the document containing s_i. There are several ways to take global information into account. We concatenate a *bag-of-words* (BOW) based feature vector to the CLS representation developed in the last Transformer layer. Specifically, given a candidate answer s_i, we firstly compute the representation associated with the CLS token at the last layer, v_{CLS}. Then, we extract BOW features from $d(s_i)$. The BOW vector v_{BOW} contains the frequency of each input word from the document. It should be noted that the BOW vector contains 50265 components as we considered the same vocabulary used by RoBERTa model. Hence, the direct concatenation of v_{CLS} and v_{BOW} is not adequate: it may suffer from scaling issues as v_{CLS} consists of only 768 components. To solve this problem, we apply a random projection over v_{BOW}, that is, $\widetilde{v}_{BOW} = v_{BOW}^{\top}\mathbf{W}$. $\mathbf{W} \in \mathbb{R}^{50265 \times 768}$ is the random projection matrix. Finally, we normalize the two vectors and concatenate them. The classification is then performed using the RoBERTa's classification head.

4.3 Combined Context

Local and global contexts contain different information, thus their combination can provide a better model. The complete architecture using global and local contexts, here named DUAL-CTX, is depicted in Fig. 3. A RoBERTa model receives the question and the candidate sentence with local context encoded as described in Sect. 4.1. The output of the Transformer is then combined with the global representation by using the strategy introduced in Sect. 4.2. The architecture is modular and extensible, local and global feature extraction modules can be easily exchanged. This flexibility can lead to the definition of several models. However, our main objective is to show the benefits of global and local contexts in the AS2 task. The exhaustive evaluation of all different context combinations and strategies is beyond our scope.

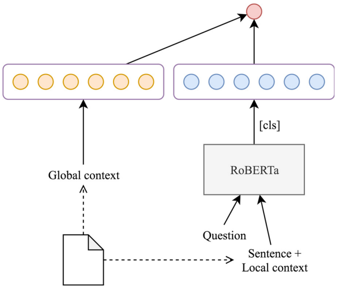


Fig. 3. Model combining global and local contexts.

5 Empirical Assessment

We carried out comparative experiments to evaluate the local and global contexts and their combination.

5.1 Corpora

We used two AS2 corpora, ASNQ, and WikiQA, to empirically assess the proposed contextual architecture. Additionally, to better collocate our research in a broader QA work, we tested our model on SQuAD, which is a standard MR dataset, adapted for AS2.

ASNQ, Answer Sentence Natural Question [5] is a large-scale open-domain corpus for AS2. The corpus is built by transforming the recently proposed Natural Question (NQ) dataset [8] corpus from MR into AS2. In short, the corpus consists of 57,242 distinct natural questions for training and 2,672 for development. For each question, candidate answers have been extracted from a single Wikipedia page. The original NQ defines a long answer (typically a paragraph) and a short answer inside the associated page, whereas the ASNQ splits the document into sentences, whose binary label is 1 if the sentence contains the short answer, 0 otherwise. The corpus contains 21,307,630 question/answer pairs, with an average of 356 answer candidates per question.

Table 3. Questions (Q) and question/answer (QA) pairs available for training.

Corpus	**Q**	**QA pairs**
ASNQ	59914	21,307,630
WikiQA	2,117	20,374
SQuAD	87,355	448,108

WikiQA [27] is an open-domain corpus containing queries sampled from Bing logs. Based on the user clicks, the questions have been associated with a Wikipedia page (only the summaries were used). We used the clean setting for which only questions having at least one good and one wrong answer are considered. The resulting corpus consists of 2,118 training, 126 development, and 243 test questions, with about 10 candidate answers per question on average. We merged the dev. and test sets as they are too small to derive reliable results from each of them individually. Overall, we have 2,117 questions and 20,374 question/answer pairs.

SQuAD 1.1, Stanford Question Answering Dataset [16], is a large-scale corpus consisting of questions crowdsourced on a set of 20,000 Wikipedia articles. The dataset was designed for MR. We transformed it into a corpus for AS2 task, by applying the same procedure described by [5]. In short, we split each input paragraph into sentences and labeled those containing the annotated answers as correct candidates, and all the others as negative candidates. After this preprocessing, our corpus contains 87,355 questions and 448,108 question/answer pairs. Please note that the results presented in this paper are not directly comparable to the SQuAD leaderboard[2].

The main characteristics of the datasets are briefly described in Table 3.

[2] https://rajpurkar.github.io/SQuAD-explorer/.

5.2 Models

We implemented our methods with RoBERTa pre-trained models, using the shared checkpoint [26]. We fine-tuned the checkpoint on our data by using (i) the Adam optimizer set with the warmup linear scheduler and a learning rate peak of $1e-6$; (ii) the binary cross-entropy loss; (iii) a batch size of 64 examples on a single GPU to train on WikiQA and SQuAD; and (iii) a batch size of 512 examples distributed on 8 GPUs to train on the ASNQ corpus (which is much larger). We used the official dev. set to derive the results, thus we set the hyper-parameters, i.e., learning rate, scheduler, and batch size, on a small portion of the training set (as our dev. set). We train and test our models on SQuAD and WikiQA four times and take the average results to account for their variability.

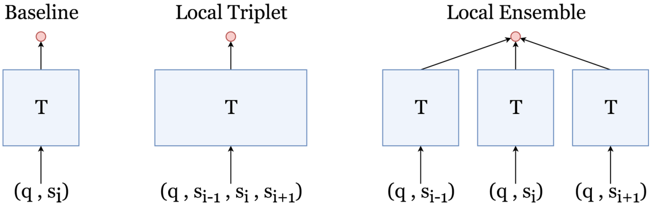


Fig. 4. Our three approaches to encode local context: a simple Transformer with question/answer pairs (left), the contextual multi-sentence architecture (center), and the ensemble of Transformers (right).

Finally, we also used the models generated with TANDA (transfer and adapt) approach [5] for WikiQA. The authors apply a first fine-tuning on ASNQ and then a second fine-tuning on the target data. TANDA is the current state of the art, 7–10 points better than any other approach on WikiQA. Our models based on local context are depicted in Fig. 4, and described below:

- Transformer: the Transformer model for AS2 introduced in Sect. 3.2. It receives the question/answer pair as input without any context.
- Local Triplet (Loc_t): the proposed Transformer-based method described in Sect. 4.1, which relies on three different sentences, i.e., the previous, the target, and the next;
- Local Ensemble (Loc_e): an ensemble of three Transformer models encoding the three pairs, q/s_{i-1}, q/s_i, and q/s_{i+1} and a final linear layer fed with the concatenation of the *[CLS]* embeddings of the three models. The latter do not share their weights except those from *[CLS]*. The ensemble is the most expensive approach.

The baseline models for encoding global context are:

- Global BOW (Glob_b): the global context described in Sect. 4.2 consisting of a simple Transformer model with a (compressed) BOW feature set on the top;

- Global Embedding (Glob_e): a document embedding constituted by the average of the embeddings derived from all document sentences. We extract the sentence embedding using RoBERTa fine-tuned on ASNQ. We concatenate the average with the *[CLS]* representation output by the AS2 Transformer model.

We set the max sequence length of the input text to 128 tokens for Loc_t, Glob_b/e, and each branch of Loc_e, whereas the contextual architecture Loc_t uses sequences up to 256 tokens, which cover a larger input.

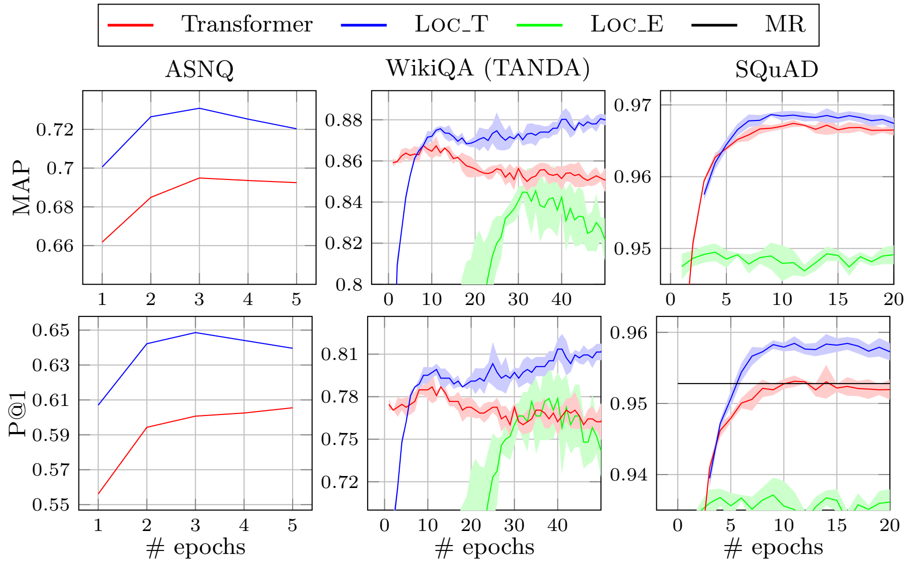


Fig. 5. Local context results (including standard deviation) computed on the dev. sets.

5.3 Results

We tested different context models on three different datasets using the state-of-the-art model in AS2 as our baseline, i.e., the transformer model made available in [5]. The latter improves all previous AS2 models 7–10 points, on WikiQA and TREC-QA datasets.

Local Context. Figure 5 shows the Mean Average Precision (MAP) and the Precision at 1 (P@1) for each epoch for the Transformer, Loc_t, and Loc_e models. The plots show two main results: first, the superior accuracy of Loc_t is evident on all corpora, demonstrating that the local context has a positive impact on the AS2 model accuracy. Additionally, the performance of Loc_e method shows that the mere use of more information is not sufficient: its arrangement into the model is fundamental. Indeed, the simple aggregation of the three summarized context vectors seems not able to capture sentence dependencies: disarranged information produces noise, with a consequent drop in performance.

Next, we used an MR Transformer [26] to implement a sentence selector model. Our MR approach achieves 0.881 of F1 score on MR task (showing competitive results on the SQuAD leaderboard with respect to single models). Then, we simply select the sentence from which the MR extracts the answer span to solve the AS2 task on SQuAD: the model achieves a P@1 of 0.952. Figure 5 shows that such model (straight line) is comparable to our baseline (single Transformer models), whereas Loc_T achieves better performance, 0.96. Although this is a loose comparison, it suggests that our approach may be applied to develop new MR methods. Table 4 illustrates interesting examples of answers correctly selected by Loc_T but misclassified by the baseline (which does not exploit any context). For example, the baseline could not reliably link *the show* to *The Glades*: this prevented to select the correct s_i as the top answer. In contrast, Loc_T contains such name in s_{i-1}.

Table 4. Input examples from WikiQA and SQuAD.

q	**What happened to "The Glades" tv series?**
s_{i-1}	The Glades was renewed by A& E for a third season on October 18, 2011, which aired from June 3 to August 12, 2012.
s_i	The show has been renewed for a fourth season.
q	**What field of computer science is primarily concerned with determining the likelihood of whether or not a problem can ultimately be solved using algorithms?**
s_i	Closely related fields in theoretical computer science are analysis of algorithms and computability theory.
s_{i+1}	A key distinction between analysis of algorithms and computational complexity theory is that the former..., Whereas the latter asks a more general question about all possible algorithms that could be used to solve the same problem

Global Context. Figure 6 shows the MAP and the P@1 achieved by the simple Transformer and the two global models, i.e., Glob_B and Glob_E. We also report the results of the combined model, which includes local and global contexts. Finally, we evaluated the models when applied to WikiQA without the TANDA approach, showing their behavior in a scenario, where there is no large and general data for the first fine-tuning step of TANDA.

The figure shows that both global methods, i.e., BOW and document embedding, improve the standard model both on WikiQA and SQuAD. We did not apply Glob_B and Glob_E to ASNQ as the training has a very large computational cost. This means that we cannot apply TANDA to WikiQA with such context. In any case, the global context produces an increase of accuracy on

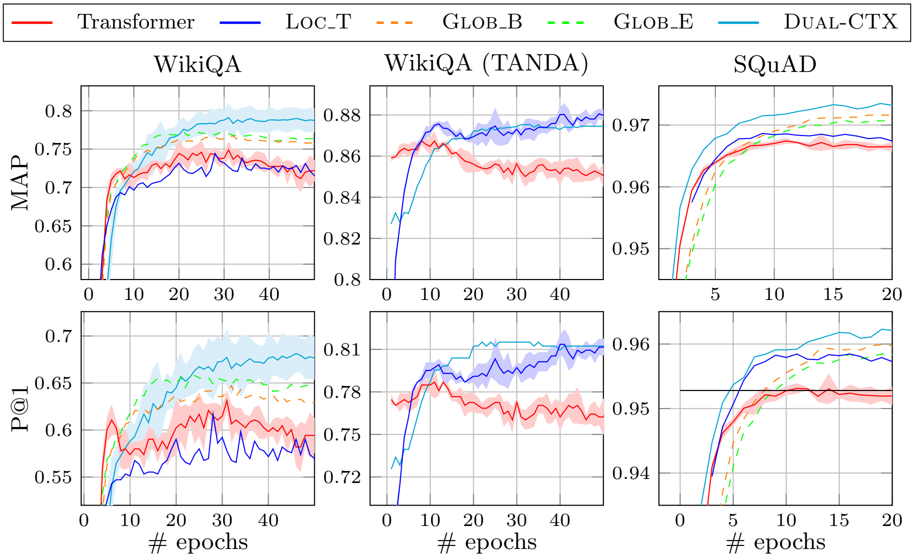


Fig. 6. Global context - empirical results computed on the development sets. The standard deviation is not always exposed to improve the readability.

WikiQA and SQuAD (w/o TANDA). Concerning the combined model, Dual-CTX improves the overall performance on WikiQA (w/o TANDA) and SQuAD. It does not improve the MAP of Loc_T on WikiQA when TANDA is used, but P@1 receives a significant boost. This result provides evidence that global and local features describe different (and potentially orthogonal) information.

It should be noted that we used BOW in the Dual-CTX rather than the document embedding for computational reasons. The BOW representation can be efficiently computed, and it does not require dedicated hardware. Conversely, the document embedding requires the application of a RoBERTa model to each sentence composing the document. Moreover, the BOW representation can be highly improved, for instance, by learning the projection matrix. This is an interesting research line we would like to explore in the future.

6 Conclusion

AS2 is an important IR task, which provides an effective and efficient solution for the design of automated QA systems. Previous state-of-the-art models for AS2 only considered the question and the answer sentence candidate, without taking the context into account, and, to our knowledge, previous work did not use a context beyond the target sentence with Transformer models.

In this paper, we define two types of context, local and global. The former tries to solve implicit references in a candidate sentence, and it consists of the previous and successive sentences of a candidate answer. Conversely, the global context injects document related information, such as the main content and topics. We proposed Transformer-based architectures that leverage the different contexts for AS2. Our empirical assessment shows that our proposed approach remarkably improves over the TANDA model, which is the state of the art for AS2, on three different AS2 datasets, i.e., ASNQ, WikiQA, and SQuAD 1.1 adapted for AS2. It should be stressed that we used the model made available by the TANDA's authors, thus our results are perfectly comparable with their model. We also release our contextualized checkpoints and the SQuAD adaption for AS2[3]. In addition to some follow up in [6], interesting future extensions of our work regard the extraction of features from the entire rank of documents retrieved for a question. Clearly, learning to rank features can also improve the selection of answer sentences.

References

1. Alberti, C., Lee, K., Collins, M.: A BERT baseline for the natural questions. arXiv preprint arXiv:1901.08634 (2019)
2. Chen, D., Fisch, A., Weston, J., Bordes, A.: Reading Wikipedia to answer open-domain questions. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1870–1879. Association for Computational Linguistics, Vancouver (2017). https://doi.org/10.18653/v1/P17-1171
3. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., Salakhutdinov, R.: Transformer-XL: attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2978–2988. Association for Computational Linguistics, Florence (July 2019). https://doi.org/10.18653/v1/P19-1285
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis (June 2019). https://doi.org/10.18653/v1/N19-1423
5. Garg, S., Vu, T., Moschitti, A.: TANDA: transfer and adapt pre-trained transformer models for answer sentence selection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 7780–7788 (2020)
6. Han, R., Soldaini, L., Moschitti, A.: Modeling context in answer sentence selection systems on a latency budget. In: Proceedings of The 16th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics (2021)
7. Kumar, S., Mehta, K., Rasiwasia, N., et al.: Improving answer selection and answer triggering using hard negatives. In: EMNLP-IJCNLP (2019)
8. Kwiatkowski, T., et al.: Natural questions: a benchmark for question answering research. Trans. Assoc. Comput. Linguist. **7**, 453–466 (2019)
9. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach, CoRR abs/1907.11692 (2019). http://arxiv.org/abs/1907.11692

[3] https://github.com/alexa/wqa-contextual-qa.

10. Matsubara, Y., Vu, T., Moschitti, A.: Reranking for efficient transformer-based answer selection. In: Huang, J., et al. (eds.) Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020, Virtual Event, China, 25–30 July 2020, pp. 1577–1580. ACM (2020). https://doi.org/10.1145/3397271.3401266
11. Nogueira, R., Cho, K.: Passage re-ranking with BERT, CoRR abs/1901.04085 (2019). http://arxiv.org/abs/1901.04085
12. Peters, M.E., et al.: Deep contextualized word representations, CoRR abs/1802.05365 (2018). http://arxiv.org/abs/1802.05365
13. Qiao, Y., Xiong, C., Liu, Z., Liu, Z.: Understanding the behaviors of BERT in ranking, CoRR abs/1904.07531 (2019). http://arxiv.org/abs/1904.07531
14. Qu, C., Yang, L., Qiu, M., Croft, W.B., Zhang, Y., Iyyer, M.: Bert with history answer embedding for conversational question answering. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1133–1136 (2019)
15. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2018). https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf
16. Rajpurkar, P., Jia, R., Liang, P.: Know what you don't know: unanswerable questions for squad. arXiv preprint arXiv:1806.03822 (2018)
17. Severyn, A., Moschitti, A.: Learning to rank short text pairs with convolutional deep neural networks. In: SIGIR. ACM (2015)
18. Shao, T., Guo, Y., Chen, H., Hao, Z.: Transformer-based neural network for answer selection in question answering (2019)
19. Shen, G., Yang, Y., Deng, Z.H.: Inter-weighted alignment network for sentence pair modeling. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1179–1189. Association for Computational Linguistics, Copenhagen, Denmark (September 2017). https://doi.org/10.18653/v1/D17-1122. https://www.aclweb.org/anthology/D17-1122
20. Soldaini, L., Moschitti, A.: The cascade transformer: an application for efficient answer sentence selection. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5697–5708. Association for Computational Linguistics (July 2020). https://doi.org/10.18653/v1/2020.acl-main.504. https://www.aclweb.org/anthology/2020.acl-main.504
21. Tan, C., et al.: Context-aware answer sentence selection with hierarchical gated recurrent neural networks. IEEE/ACM Trans. Audio Speech Lang. Proc. **26**, 540–549 (2017)
22. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
23. Wang, M., Smith, N.A., Mitamura, T.: What is the Jeopardy model? A quasi-synchronous grammar for QA. In: EMNLP-CoNLL, pp. 22–32. Association for Computational Linguistics, Prague (June 2007). https://www.aclweb.org/anthology/D07-1003
24. Wang, S., Jiang, J.: A compare-aggregate model for matching text sequences, CoRR abs/1611.01747 (2016). http://arxiv.org/abs/1611.01747
25. Wang, Z., Ng, P., Ma, X., Nallapati, R., Xiang, B.: Multi-passage BERT: a globally normalized bert model for open-domain question answering. arXiv preprint arXiv:1908.08167 (2019)
26. Wolf, T., et al.: Transformers: state-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019)

27. Yang, Y., Yih, W., Meek, C.: WikiQA: a challenge dataset for open-domain question answering. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2013–2018. Association for Computational Linguistics, Lisbon (September 2015). https://doi.org/10.18653/v1/D15-1237. https://www.aclweb.org/anthology/D15-1237
28. Yoon, S., Dernoncourt, F., Kim, D.S., Bui, T., Jung, K.: A compare-aggregate model with latent clustering for answer selection, CoRR abs/1905.12897 (2019). http://arxiv.org/abs/1905.12897

An Argument Extraction Decoder in Open Information Extraction

Yucheng Li[1], Yan Yang[1(✉)], Qinmin Hu[2], Chengcai Chen[3], and Liang He[1]

[1] East China Normal University, Shanghai, China
yanyang@cs.ecnu.edu.cn
[2] Ryerson University, Toronto, Canada
[3] Xiaoi Robot Technology Co., Ltd., Shanghai, China

Abstract. In this paper, we present a feature fusion decoder for argument extraction in Open Information Extraction (Open IE), where we challenge argument extraction as a predicate-dependent task. Therefore, we create a predicate-specific embedding layer to allow the argument extraction module fully shares the predicate information and the contextualized information of the given sentence, after using a pre-trained BERT model to achieve the predicates. After that, we propose a decoder in argument extraction that leverages both token features and span features to extract arguments with two steps as **argument boundary identification** by token features and **argument role labeling** by span features. Experimental results show that the proposed decoder significantly enhances the extraction performance. Our approach establishes a new state-of-the-art result on two benchmarks as OIE2016 and Re-OIE2016.

Keywords: Open Information Extraction · Argument extraction · Span extraction · Decoder

1 Introduction

Open Information Extraction (Open IE) has been widely used in many downstream tasks [12] such as word embedding learning [16], document summarization [8] and question answering [7], which aims to generate structured tuples consisting of predicate and their arguments that represents assertions in a given sentence. An example of Open IE is shown in Table 1.

Previous neural Open IE systems usually treat Open IE as a pipeline task, including the following two independent sub-tasks as: (1) extracting predicates first, and (2) extracting corresponding arguments later [18,20]. However, researchers observe that the extraction of predicate and arguments are tightly interwoven [2,9], which means these two sub-tasks are not independent in realistic. This motivates us to consider them to be dependent/joint rather than independently sequential. Our work focuses on argument extraction and regard it as a predicate-dependent task.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 313–326, 2021.
https://doi.org/10.1007/978-3-030-72113-8_21

Moreover, the existing Open IE approaches leverage different level features in argument extraction. For example, Stanovsky et al. [18] uses *token features* to extract arguments by the sequence tagging method with customized BIO tags. Zhan et al. [20] adopts a span-based approach that enumerates all possible candidate spans and scores them with labels via *span features*. However, Ouchi et al. [13] presents that sequence tagging approaches generate span boundary more accurately than span-based method, yet span-based methods produce label prediction more precisely than sequence tagging models. This motivates us to jointly utilize token features and span features in argument extraction, which is our proposed decoder.

Table 1. An example sentence and respective Open IE extractions. The extractions consists of a predicate phrase (underlined) and a list of arguments. The proposed decoder is applied in argument extraction, which extracts argument in two steps: identifying argument boundary and labeling the argument role.

Example sentence	
Costco has missed the trend this summer	
Predicate extraction	
(Costco$_O$ has$_{\textbf{B-pred}}$ missed$_{\textbf{I-pred}}$ the$_O$ trend$_O$ this$_O$ summer$_O$.$_O$)	
Argument extraction	
Argument boundary	(Costco$_{\mathbf{B}}$ has$_O$ missed$_O$ the$_{\mathbf{B}}$ trend$_{\mathbf{I}}$ this$_{\mathbf{B}}$ summer$_{\mathbf{I}}$.$_O$)
Argument role labeling	(Costco, **A0**), (the trend, **A1**), (this summer, **A2**)
Output	
(A0: Costco; has missed; A1: the trend; A2: this summer)	

In this paper, we propose a decoder of multi-level feature fusion for argument extraction in an Open IE framework. The framework is introduced in Fig. 1. First, we train a BERT-based model for predicate extraction and then create a predicate-specific embedding layer as the input for the argument extraction module. Unlike the pipeline approaches, the predicate-specific embedding layer allows argument extraction to share useful features from predicate extraction. Second, we offer a decoder for argument extraction as our unique contribution, which jointly leverages both token features and span features. Our decoder extract arguments in two steps: identifying the boundary of arguments with token features and labeling the role of arguments with span features, where the features are fused in the overall decoding.

In order to better understand our Open IE framework and the proposed decoder shown in Fig. 1, we show the extraction process of our approach in Table 1. Note that the ultimate goal is to extract structured tuples consisting of a predicate and a list of arguments. We have an example sentence as "Costco has missed the trend this summer." in Table 1. At first, we feed this sentence to the pre-trained BERT model in Fig. 1 to extract its predicates as "has$_{\textbf{B-pred}}$", "missed$_{\textbf{I-pred}}$". After that, the argument extraction model extracts the argument boundary for the sentence with the BIO tags, followed by the role labels of "A0" for "Costco", "A1" for "the trend" and "A2" for "this summer".

We conduct experiments on two Open IE benchmarks. The experimental results on OIE2016 show that our method outperforms the state-of-the-art (SpanOIE [20]) by about 4.7 F1 points.

2 The Open IE Framework with Our Proposed Decoder

The framework in Fig. 1 mainly consists of three parts: (1) predicate extraction; (2) predicate-specific embedding for LSTM; and (3) argument extraction with a decoder.

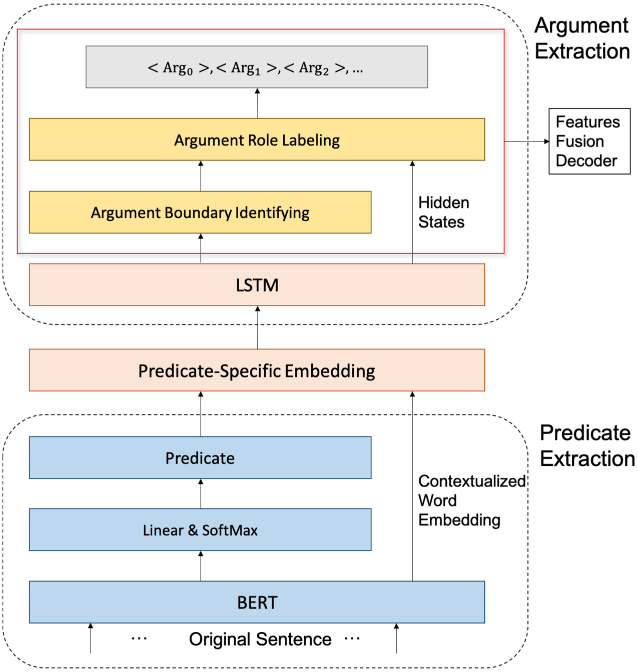

Argument Extraction
$< \mathrm{Arg}_0 >, < \mathrm{Arg}_1 >, < \mathrm{Arg}_2 >, \ldots$
Argument Role Labeling
Features Fusion Decoder
Argument Boundary Identifying
Hidden States
LSTM
Predicate-Specific Embedding
Predicate Extraction
Predicate
Contextualized Word Embedding
Linear & SoftMax
BERT
··· Original Sentence ···

Fig. 1. The architecture of our Open IE framework, including a predicate extraction module, a predicate-specific embedding layer for argument generation purpose, and a argument extraction module where a decoder is proposed as our major contribution.

2.1 Predicate Extractions

As shown in Fig. 1, we add a linear softmax layer on top of the BERT encoder to extract predicate. Specifically, the linear softmax layer works on the final produced contextualized word representations. Therefore, given a sentence $S = (w_1, w_2, \cdots, w_n)$, the BERT model [5] produced a list of contextualized word embeddings $(h_1, \cdots, h_i, \cdots, h_n)$ where each h_i represents the i-th input token w_i. Then, the predicate extraction process predicts a list of BIO labels to identify the predicate. The label distribution of the i-th token for predicate extraction is computed as follows:

$$P_i^{PE} = \mathrm{softmax}(W_1 h_i + b_1)$$

where W_1 and b_1 is the trainable weight matrix and the bias for predicate extraction, h_i is the contextualized token embedding of the i-th token.

2.2 The Predicate-Specific Embedding Layer

First of all, this predicate-specific embedding layer works for the argument extraction layer. In particular, the layer is the input of the LSTM encoder.

The motivation of this embedding layer lies in: (1) we extract arguments based on the generated predicate to make predicate extraction and argument extraction be dependent; (2) the representation of predicate is applied on the contextualized word embedding using the attention mechanism theory; and (3) the predicate-specific embedding allows LSTM to directly access predicate information, since we concatenate the representation of predicate on it.

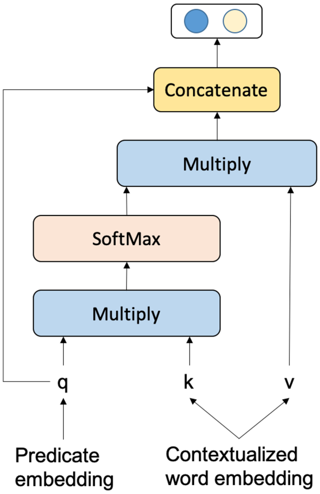


Fig. 2. The predicate-specific embedding layer as the input of the LSTM encoder.

Figure 2 presents the flowchart of the predicate-specific embedding layer, in which the output is the input of the LSTM encoder. Formally, we define the input of the LSTM network as:

$$h_i' = h_p \oplus (\alpha_i \cdot h_i)$$

where h'_i is the the i-th input vector of LSTM, h_p represents the contextualized embedding of the predicate, the $\oplus$ is the concatenation operator, and α_i is the weight assigned to the token embedding h_i.

We utilize selective attention [11] to weight each token embedding based on the predicate representation, in order to extract arguments based on the extracted predicates. Selective attention learns to identify the tokens that are highly related to the extracted predicate rather than treating each token representation equally. The weight α_i is obtained for each token as follows:

$$\alpha_i = \frac{exp(h_p \cdot h_i)}{\sum_{i=1}^{n} exp(h_p \cdot h_i)}$$

The hidden states produced by LSTM are formulated as follow:

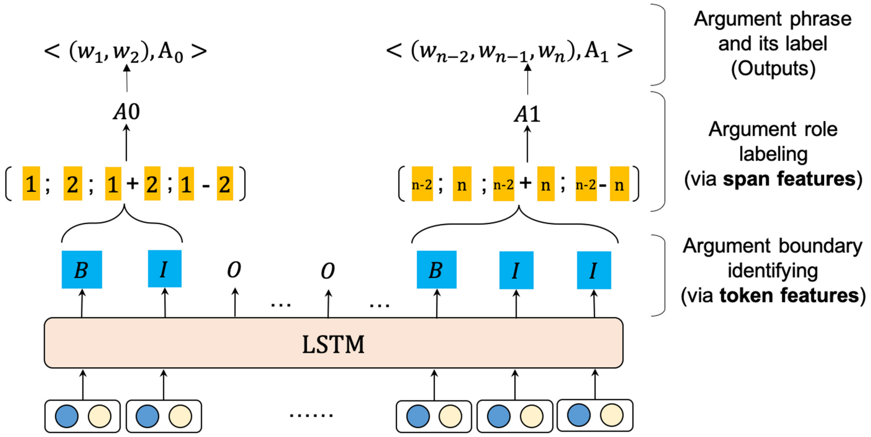
$$l_i = \mathrm{LSTM}(h'_i, l_{i-1})$$

where l_i is the i-th output hidden states and h'_i is the i-th input vector.

2.3 Argument Extraction with the Proposed Decoder

As shown in Fig. 1, the argument extraction module consists of an LSTM encoder and our proposed decoder. The proposed decoder fuses multi-level features to extract arguments.

Our decoding process divides the argument extraction into two steps in Fig. 3: (1) identifying the argument boundary with BIO tagging; and (2) labeling the role of arguments with span classification.



$< (w_1, w_2), A_0 >$
$< (w_{n-2}, w_{n-1}, w_n), A_1 >$
A0
A1
[1 ; 2 ; 1 + 2 ; 1 - 2]
[n-2 ; n ; n-2 + n ; n-2 - n]
B
I
O
O
B
I
I
...
...
LSTM
......
Argument phrase and its label (Outputs)
Argument role labeling (via **span features**)
Argument boundary identifying (via **token features**)

Fig. 3. The decoder of multi-level features fusion extracts argument in two steps.

Identifying the Argument Boundary. To identify the argument boundary, we adopt BIO tags that indicate the start and the end of the argument phrase.

Comparing with previous work [18] that uses customized BIO tagging (i.e., B-A0, I-A0, B-A1, B-A1, B-A2, ...) to extract argument directly, the BIO tags in our method as 'B', 'I' and 'O', are only to identify the boundary. Specifically, we apply a linear layer plus a softmax function on top of the LSTM network that produces labels for each word. Formally, the output distribution of the i-th token for the argument boundary labeling is as follows:

$$P_i^{AB} = \text{softmax}(W_2 l_i + b_2)$$

where l_i is the hidden state of the i-th token produced by the LSTM network, W_2 and b_2 are the trainable parameter matrix and the bias.

Labeling the Role of Argument. We take the span-level features to predict the role of argument. Formally, the span features are constructed as follows:

$$f_{span}(s_{i:j}) = l_i \oplus l_j \oplus (l_i + l_j) \oplus (l_i - l_j)$$

where $s_{i:j}$ is the argument span identified by the BIO tags, which starts at i and ends at j, l_i and l_j are representations of the start token and the end token produced by the LSTM network, $\oplus$ indicates the concatenation operation.

The span features are then fed into a linear layer to obtain the scores of different labels for each span.

$$\texttt{Score}(y|s_{i:j}) = \text{softmax}(W_3 f_{span}(s_{i:j}) + b_3)$$

where W_3 and b_3 is the trainable parameter matrix and the bias, y is the role label. For each span $s_{i:j}$, we select the label with the highest score as its final results:

$$\arg\min_y \texttt{Score}(y|s_{i:j}), y \in [A_0, A_1, A_2, A_3]$$

2.4 Training

To train our Open IE framework, we jointly minimize three loss functions. For each training sample S, the loss function are formulated as follows:

$$L = -\left[\sum_S \log P_i^{PE}(Y_{pred}) + \sum_S \log P_i^{AB}(Y_{argu\_bound}) + \sum_S \log(\texttt{Score}(\hat{y}|s_{i:j}))\right]$$

where Y_{pred} is the gold label of predicated extraction, Y_{argu_bound} is the gold label of argument boundary identification, and $\hat{y}$ is the gold argument role of span $S_{i:j}$.

Note that we use *teacher forcing* in the training process of the argument boundary identification and the argument role labeling. For detail information about *teacher forcing*, we refer readers to [19].

3 Experiments

3.1 Data

We use the training dataset processed by [20], which uses all the sentences that are fewer than 40 words in Wikipedia dump 20180101 and extract corresponding n-ary information tuples by an exited Open IE system OpenIE 4 [12]. The extraction of OpenIE 4 is used as training data in many neural Open IE systems [3,10,20] due to its reasonable computational cost and generation quality. Different from [20], to reduce the noise, we only keep the tuples with a confidence score higher than 0.9. Finally, there are a total of 2,175,294 (sentence, tuple) pairs in our training dataset.

For the test data, we test our model on two Open IE benchmark datasets, `OIE2016` [15] and `Re-OIE2016` [20]. `OIE2016` is a widely used test dataset for Open IE that automatically transferred from QA-SRL. We use a subset of `OIE2016` that contains 600 sentences with 1,730 extractions[1]. We also leverage the `Re-OIE2016` benchmark proposed in [20]. `Re-OIE2016` was relabeled on the basic of `OIE2016` manually to reduce incorrect tuples in `OIE2016` that contains 595 sentences with 1,506 extractions.

3.2 Settings

We take the pre-trained BERT model [5] as our base sentence encoder. The BERT model we use is `bert-base-cased` pre-trained on BooksCorpus, which consists of 12 transformer layers, 12 attention heads, and 768 dimensional states. We employ a one-layer LSTM network with the hidden state size of 1536 as our second encoder.

For hyper-parameters, we use a similar setting reported in BERT. We set the learning rate to $5e{-5}$ and use a linear learning rate decay schedule with warm-up over $2e{-3}$ of the training updates for our optimizer. We also set the dropout rate to 0.1 for the Transformer blocks and 0.2 for the classifier. We split the training dataset into eight partitions and random sample instances to train our model. This reduces the size of epochs, resulting in less training time. We set the batch size to 64 and trained our model for four epochs.

3.3 Baselines

We compare our method with the rule-based Open IE systems, including ClauseIE [4] and OpenIE 4 [12]. We also compare our approach with the state-of-the-art neural Open IE systems, including RNN OIE [18], SpanOIE [20], Seq2Seq OIE [3], and IMoJIE [10]. The RNN OIE is a sequence labeling Open IE system and SpanOIE is a span-prediction Open IE system. Both of them are pipeline methods. The seq2seq OIE and IMoJIE model are sequence-generation based Open IE system, which can only produce binary extraction (subject-verb-object

[1] This subset is also used as test data in [18,20].

tuples) instead of n-ary extractions. The IMoJIE model adopts BERT as a basic encoder. Note that all neural Open IE models, except RNN OIE, are trained on the same training set as our method[2]. RNN OIE is trained on a dataset transferred from QAMR [18]. We test baselines by directly evaluating their extractions of the test set of OIE2016, which is published in [15] or in the related published papers.

3.4 Metrics

We evaluate all approaches based on three popular metrics. First, *precision-recall (PR) curve* is widely used in evaluating the Open IE systems' performance at different extraction confidence thresholds. Second, we compute the *area under the PR curve (PR-AUC)* to get an overall measurement of the overall system performance. Finally, for each system, we report a single *F1-score* using a confidence threshold optimized on the development set. Note that, since we do not implement the confidence scoring function in our work, we set the confidence score of all extracted results to 1. Therefore, our PR curve will be a straight line. We also treat all extractions as confident results in the evaluation of F1 (i.e., confidence threshold for our approach is set to 1). Additionally, to verify the robustness of our method, the results used in Sect. 4 is the average performance of 5 runs of our model.

4 Discussion and Analysis

4.1 Overall Analysis

We use the scripts published in [15,20] to evaluate the precision and recall of the baseline models over the OIE2016 and Re-OIE2016 datasets[3]. The results are shown in Table 2.

We find that our approach outperforms all baselines significantly. Our approach achieves the best AUC score of 0.551 over OIE2016, which exceeds Span OIE by 6.2%, and it gains the best AUC score of 0.703 over Re-OIE2016, which exceeds IMoJIE by 1.1%. Compared with RNN OIE (the sequence labeling model) and SpanOIE (the span prediction model), our approach shows progress on precision and recall, which demonstrates the effectiveness of joint learning and feature fusion. What's more, although the performance of IMoJIE on Re-OIE2016 is remarkably strong, we find that our method achieves a better recall score. More important, our approach tends to find more complete predicates, which leads to a higher recall for argument extraction. When compared with the

[2] The only difference is the confidence score for training data chosen by different baselines, please check Sect. 3.1 for details.

[3] Note that results reported in [15] contradicts our results. That is because the author changed the matching function of evaluation scripts. While this changes the absolute performance numbers of the different systems, it does not change the relative performance of any of the tested systems.

Table 2. The Area under P-R Curve (AUC) and f1-score of Open IE systems over the OIE2016 and Re-OIE2016 datasets.

Systems	OIE2016		Re-OIE2016	
	AUC	F1	AUC	F1
ClausIE	0.364	58.01	0.464	64.17
OpenIE 4	0.408	58.83	0.509	68.32
IMoJIE	0.409	58.40	0.692	**79.90**
RNN OIE	0.462	68.55	–	–
Seq2Seq OIE	0.473	–	–	–
SpanOIE	0.489	68.65	0.659	78.50
Our approach	**0.554**	**73.93**	**0.708**	79.23

rule-based methods, our approach obtains better performance than Open IE 4 on both AUC and F-1 score, which shows that our model is capable of learning from good extractions.

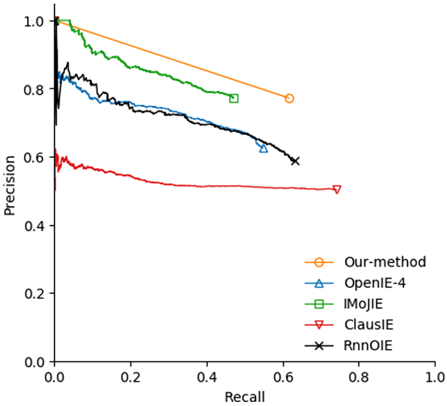


Fig. 4. The precision-recall curve on the OIE2016 test dataset. Since the extraction results of seq2seq OIE and SpanOIE are not published, we do not draw pr curve for these models.

The results of the PR curve on OIE 2016 are shown in Fig. 4. The results show that the PR curve of our method is consistently above other baselines. We find that the improvement of our method over other baselines comes from the following two aspects: (1) our method can find more predicates than other methods, which lead to a higher recall; and (2) our method is more accurate in finding argument owing to the precise argument role labeling.

4.2 Analysis of the Joint Modeling

To further investigate the joint learning of the two-sub tasks of Open IE and our predicate-specific embedding layer, we compare our method (**Joint**) with the pipeline approach (**Pipeline**), which employs two independent labeling models for predicate extraction and argument extraction. The experimental results are shown in Table 3.

Table 3. Comparison of pipeline method and our multi-task learning method. Tested on OIE2016.

	Pipeline			Joint (our method)		
	Precision	Recall	F1	Precision	Recall	F1
Predicate extraction	0.840	0.928	**0.882**	0.816	0.920	0.864
Argument extraction	0.710	0.677	0.693	0.748	0.681	**0.713**
Overall	0.726	0.701	0.713	0.774	0.708	**0.739**

In Table 3, we find that the pipeline model achieves the best F1 score as 0.882 in term of predicate extraction. Our joint model achieves a comparable F1 score as 0.859 in predicate extraction. We draw this conclusion that predicate extraction gains little benefits from the argument extraction process. The reason we analyze is because predicate extraction is relatively straightforward to learn.

As for argument extraction, we see the joint model outperforms the pipeline method. We say that argument extraction is highly related to the predicate extraction process, and the argument extraction process can be better predicted by sharing useful features with the predicate extraction procedure via our predicate-specific embedding layer. The other point is that the increase of performance mainly comes from the rise of precision. The recall is relatively consistent.

4.3 Analysis of Feature Fusion

We evaluate the decoder of multi-level features fusion here. We compare our method with the customized BIO tagging approach [18] that leverage token features only (**w/o span features**). We test the proposed decoder in two factors: the argument boundary identification and the argument role labeling. We regard the argument boundary as correct if it matched the gold annotation regardless of its role label, and we evaluate argument role labeling for whose boundaries match the gold annotation. The results are shown in Table 4.

As shown in Table 4, our approach outperforms baselines in both argument boundary identification and argument role labeling. Since we apply the simplified BIO tags to identify the boundaries of arguments, our model has a smaller output space than the customized BIO tags. This may be the reason for our better boundary identifying performance. Moreover, the span features contribute a lot

Table 4. Comparison of custom BIO method (without span features) and our method. Tested on OIE2016.

	W/o span features			Our method		
	Precision	Recall	F1	Precision	Recall	F1
Argument boundary	0.737	0.690	0.712	0.748	0.698	**0.722**
Argument role labeling	0.788	0.802	0.795	0.809	0.796	**0.802**
Overall	0.758	0.701	0.728	0.774	0.709	**0.739**

in argument role labeling. That shows the span-level features are more suitable than token-level features to make role labeling prediction.

We also present extraction examples in Table 5 to show the benefits of using multi-level features. According to the results, RNN OIE can deal with inputs with normal word order. However, it is confused by the input with inverted word order that regards the object as the subject. That may be because the token feature is dominated by the position information, which makes it difficult to predict the correct argument role when the inputs' word order is inverted. With the usage of span features, our approach shows the potential to reveal the semantic dependencies among subjects and predicates that lead to correct extractions.

Table 5. Example sentences and respective extractions of RNN OIE and our method. The first sentence has a normal word order (i.e., Subject-Verb-Object (SVO) order). The second sentence has an inverted word order (i.e., OVS order).

Original sentence	Elon Musk said "SpaceX will send human to the Mars in recent 10 years"
RNN OIE	(A0: Elon Musk; said; A1: "SpaceX will send human to the Mars in recent 10 years")
Our method	(A0: Elon Musk; said; A1: "SpaceX will send human to the Mars in recent 10 years")
Original sentence	"SpaceX will send human to the Mars in recent 10 years", said Elon Musk
RNN OIE	(A0: "SpaceX will send human to the Mars in recent 10 years"; said; A1: Elon Musk)
Our method	(A0: Elon Musk; said; A1: "SpaceX will send human to the Mars in recent 10 years")

4.4 Error Analysis

We randomly sample 50 sentences from OIE2016 test set and analyze errors in extractions produced by our approach. We find several common problems that take the main part of errors.

- Redundant extraction: Although our method rarely generates repetitive tuples (comparing with generation-based Open IE systems like IMoJIE), it still suffers from the redundant problem. Nearly 52% of all errors result from the irrelevant words in the extracted tuples.
- Incomplete extractions: Incomplete extractions (i.e., missing subject or object) contribute nearly 58% of the recall error. We find that it is mainly owing to the error from the argument boundary identification procedure.
- Extractions with nominalized predicates: extractions with a noun or nominal predicates are hard to extract, and it makes up 34% of all recall errors. We speculate that enhancing training instances with noun predicates can reduce this problem.

5 Related Work

Open IE was first introduced to extend traditional information extraction, such that all of the propositions asserted by a given sentence are extracted. Most Open IE systems aim to extract binary propositions (i.e., subject-verb-object tuples) or n-ary relations (i.e., arguments and predicate). Some systems also made efforts to extract in other formats, such as nested propositions.

Traditional Open IE methods use hand-crafted patterns to extract predicate-argument structures through syntactic constraints. ReVerb [6] extracts Open IE propositions from part of speech tags, OLLIE [14], ClauseIE [4], and PropS [17] post-process dependency trees. Open IE4 [12] extracts tuples from semantic role labeling structures. Recently, neural Open IE approaches have gained great success. [18] developed RNN OIE based on a BiLSTM labeler and BIO tagging scheme, which was the first supervised model for Open IE. There were also attempts to perform Open IE in a generation setting. [3] built Seq2seq OIE that adopted a neural sequence to sequence framework with copy mechanism to generate binary extractions. To solve the stuttering problem of Seq2seq OIE, IMoJIE [10] used sequential decoding of tuples conditioned on previous tuples by adding generated extraction to the encoder. [20] introduced a span prediction model for Open IE, which exploits span-level features to extract argument phrases.

Previous studies over Open IE suffered from the lack of labeled Open IE datasets for training and evaluation. Recently, [18] created a large Open IE training corpus, which is derived from Question-Answer Meaning Representation. [3] constructed a large but noisy annotated corpus by using Open IE4 to perform extractions on Wikipedia. [20] also created a large annotated corpus similar to [3] but replaced binary extractions with n-ary extractions. To enhance the quality of training dataset, [10] proposed a novel technique to combine multiple Open IE dataset to a comprehensive dataset in a completely unsupervised manner. For the evaluation benchmark, [15] created the first large Open IE corpus `OIE2016`, which is widely used as a test set for evaluation, by automatically translating from QA-SRL. [1] made public a crowd-sourced dataset, CaRB, with novel evaluation rules that penalize overlong extractions. [20] relabeled the dataset `OIE2016` manually to reduce the noise and published a new benchmark `Re-OIE2016`.

6 Conclusions and Future Work

In this paper, we present a feature fusion decoder for argument extraction in Open IE, which extracts arguments in two steps and leverages multi-level features. Our system achieves state-of-the-art results on two Open IE benchmarks as OIE2016 and Re-OIE2016. Additionally, we perform extensive analysis and find that argument extraction depending on predicates enhances the overall Open IE performance and span features help our model to label argument role more accurately.

For future research, we will further enhance the performance of Open IE and investigate more complex extraction results, such as nested tuples.

References

1. Bhardwaj, S., Aggarwal, S., Mausam, M.: CaRB: a crowdsourced benchmark for Open IE. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 6262–6267. Association for Computational Linguistics, Hong Kong, China (November 2019). https://doi.org/10.18653/v1/D19-1651. https://www.aclweb.org/anthology/D19-1651
2. Chen, D., Li, Y., Lei, K., Shen, Y.: Relabel the noise: joint extraction of entities and relations via cooperative multiagents. arXiv preprint arXiv:2004.09930 (2020)
3. Cui, L., Wei, F., Zhou, M.: Neural open information extraction. arXiv preprint arXiv:1805.04270 (2018)
4. Del Corro, L., Gemulla, R.: ClausIE: clause-based open information extraction. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 355–366 (2013)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Fader, A., Soderland, S., Etzioni, O.: Identifying relations for open information extraction. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1535–1545. Association for Computational Linguistics (2011)
7. Fader, A., Zettlemoyer, L., Etzioni, O.: Open question answering over curated and extracted knowledge bases. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1156–1165 (2014)
8. Fan, A., Gardent, C., Braud, C., Bordes, A.: Using local knowledge graph construction to scale seq2seq models to multi-document inputs. arXiv preprint arXiv:1910.08435 (2019)
9. He, R., Wang, J., Guo, F., Han, Y.: Transs-driven joint learning architecture for implicit discourse relation recognition. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 139–148 (2020)
10. Kolluru, K., Aggarwal, S., Rathore, V., Mausam, Chakrabarti, S.: IMoJIE: iterative memory-based joint open information extraction (2020)
11. Lin, Y., Shen, S., Liu, Z., Luan, H., Sun, M.: Neural relation extraction with selective attention over instances. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2124–2133. Association for Computational Linguistics, Berlin (August 2016). https://doi.org/10.18653/v1/P16-1200. https://www.aclweb.org/anthology/P16-1200

12. Mausam, M.: Open information extraction systems and downstream applications. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, pp. 4074–4077 (2016)
13. Ouchi, H., Shindo, H., Matsumoto, Y.: A span selection model for semantic role labeling. arXiv preprint arXiv:1810.02245 (2018)
14. Schmitz, M., Bart, R., Soderland, S., Etzioni, O., et al.: Open language learning for information extraction. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 523–534. Association for Computational Linguistics (2012)
15. Stanovsky, G., Dagan, I.: Creating a large benchmark for open information extraction. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2300–2305 (2016)
16. Stanovsky, G., Dagan, I., et al.: Open IE as an intermediate structure for semantic tasks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pp. 303–308 (2015)
17. Stanovsky, G., Ficler, J., Dagan, I., Goldberg, Y.: Getting more out of syntax with props. arXiv preprint arXiv:1603.01648 (2016)
18. Stanovsky, G., Michael, J., Zettlemoyer, L., Dagan, I.: Supervised open information extraction. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 885–895 (2018)
19. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Comput. **1**(2), 270–280 (1989)
20. Zhan, J., Zhao, H.: Span model for open information extraction on accurate corpus (2020)

Using the Hammer only on Nails: A Hybrid Method for Representation-Based Evidence Retrieval for Question Answering

Zhengzhong Liang[1](✉), Yiyun Zhao[2], and Mihai Surdeanu[1]

[1] Department of Computer Science, University of Arizona, Tucson, AZ 85721, USA
zhengzhongliang@email.arizona.edu, msurdeanu@email.arizona.edu

[2] Department of Linguistics, University of Arizona, Tucson, AZ 85719, USA
yiyunzhao@email.arizona.edu

Abstract. Evidence retrieval is a key component of explainable question answering (QA). We argue that, despite recent progress, transformer network-based approaches such as universal sentence encoder (USE-QA) do not always outperform traditional information retrieval (IR) methods such as BM25 for evidence retrieval for QA. We introduce a lexical probing task that validates this observation: we demonstrate that neural IR methods have the capacity to capture lexical differences between questions and answers, but miss obvious lexical overlap signal. Learning from this probing analysis, we introduce a hybrid approach for representation-based evidence retrieval that combines the advantages of both IR directions. Our approach uses a routing classifier that learns when to direct incoming questions to BM25 vs. USE-QA for evidence retrieval using very simple statistics, which can be efficiently extracted from the top candidate evidence sentences produced by a BM25 model. We demonstrate that this hybrid evidence retrieval generally performs better than either individual retrieval strategy on three QA datasets: OpenBookQA, ReQA SQuAD, and ReQA NQ. Furthermore, we show that the proposed routing strategy is considerably faster than neural methods, with a runtime that is up to 5 times faster than USE-QA.

Keywords: Neural information retrieval · Representation-based · BM25

1 Introduction

Open-domain question answering (QA) systems traditionally have three components: evidence retrieval, evidence reranking, and answer classification/extraction. In evidence retrieval, the model retrieves a smaller set of possibly useful evidence texts from a large knowledge base (KB), which are then reranked

This work was supported by the DARPA, grant number HR00111990011.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 327–341, 2021.
https://doi.org/10.1007/978-3-030-72113-8_22

by the following component to push the most useful information to the top. Traditional directions use word-overlap based models for evidence retrieval such as tf-idf and BM25. However, this can potentially cause the missing of useful information due to the "lexical chasm" [2] between the question and the answer. A potential remedy for this is to use neural networks for evidence retrieval, such as transformer network-based contextualized embedding methods [7,28,29].

Focusing on this evidence retrieval stage of a QA system, we argue that, for this component, transformer networks should not always be preferred over standard information retrieval (IR) methods. First, due to their reliance on continuous representations, transformer methods do not take direct advantage of obvious lexical evidence. This is a drawback in long-text retrieval, which tends to be affected less by the lexical chasm problem than short-text retrieval. Second, transformer-based methods are expensive to run, which makes them a less than ideal choice for end-user NLP applications with temporal constraints.

In this paper we introduce a *hybrid* approach for evidence retrieval for question answering.[1] Our approach uses a routing classifier that routes an incoming question to either an IR method or a supervised transformer method for evidence retrieval, using solely shallow statistics sampled from the knowledge base of explanatory texts for each question. This strategy has two benefits: first, evidence retrieval performance improves overall because each question is handled by the appropriate retrieval method. Second, this method reduces computational overhead because for a considerable number of questions it does not use the more expensive neural component. In particular, our contributions are:

(1) We design and conduct a series of supervised lexical probing tasks on two QA datasets, which are trained to predict the terms in the query and the gold evidence text from the entire vocabulary, using as input either the tf-idf vector of the query, or the neural embedding of the same query. The comparison of the two probes indicates that the probe trained from the tf-idf vector of the query tends to predict terms that exist in the original query (thus emphasizing lexical overlap), whereas the probe trained on top of the query's neural embedding predicts more terms in the evidence text that do not exist in the query (thus bridging the lexical chasm). This validates our hypothesis that different retrieval strategies should be used in different scenarios.

(2) Learning from this observation, we propose a hybrid retrieval method, which routes queries to either an information retrieval method (BM25 [24]) or a transformer-based one (USE-QA [28]). We show that this routing decision is learnable from simple statistics that can be efficiently extracted from the top documents retrieved by an IR method.

(3) We show that using this hybrid strategy generally improves evidence retrieval performance in three QA datasets: OpenBookQA [16], ReQA SQUAD, and ReQA Natural Questions (NQ) [1]. The hybrid approach performs significantly better than either individual model on ReQA SQUAD

[1] (Code is available at: https://github.com/clulab/releases/tree/master/ecir2021-hyrbid-retrieval).

and NQ, with improvements in the mean reciprocal rank (MRR) of the correct evidence sentence ranging from 1% to 7.4% (depending on the dataset). On OpenBookQA the difference between the hybrid method and USE-QA is not statistically significant.

(4) Our analysis indicates that the hybrid method is significantly faster than neural IR methods. For example, the hybrid method is 2.2 times faster than USE-QA in OpenBookQA, and 5.2 times faster in ReQA SQUAD.

2 Related Work

Neural IR methods provide an exciting potential direction to mitigate the lexical chasm in QA [6]. Neural IR approaches can be broadly divided into two categories: representation-based and interaction-based [8]. Representation-based neural IR directions *pre-encode* the query and the document into a continuous representation learned using a subsample of the data, and use a shallow method to compute relevance scores at runtime (e.g., dot product) [12,15]. Representation-based neural IR methods have low runtime overhead because all documents can be pre-computed as vectors, so that at test time the embeddings of the documents do not have to be recomputed for each query (i.e., the neural model is run for N_q times at test time, where N_q is the number of queries).

Interaction-based methods learn a query-specific representation of the documents *at runtime* [11,19,22]. Usually the query and the candidate document are concatenated and processed by a neural model jointly, so that complex interactions of the terms in the query and the document can be better captured. However, this requires running the neural model for $N_q \cdot N_d$ times at test time (where N_q is the number queries and N_d is the number of docs). Therefore interaction-based methods are not suitable for large-scale first stage retrieval and are usually used for second-stage retrieval (reranking). In this paper we focus on the representation-based method in the first stage retrieval.

Empirical evidence has shown that neural IR methods perform better in short-text retrieval, where the word-overlap-based IR methods are more likely to suffer from the lexical chasm problem. However, not much work has been done to show why neural IR methods are able to reduce the lexical chasm problem [8], partly because it is hard to explain the meaning of neural embeddings. Recently, probing tasks have been widely used to help understand the properties of neural networks [5,9,10]. In probing tasks, a shallow model is placed on top of the large neural model, and the shallow model is trained to show some properties of the large model. For example, in [10], the authors show that some syntactic information is encoded in the embeddings of the intermediate layers of BERT. Inspired by this, we design and conduct a series of lexical probing tasks to compare the abilities of traditional IR methods and neural IR methods to predict the terms that are indicative of lexical chasm, i.e., they exist in the evidence sentences but not in the original query.

Although they do not rely on explicit word overlaps, neural IR methods do not always outperform traditional IR. For instance, it has been shown that neural IR models usually work better on short text retrieval [4], and when training data is abundant [8], but not in other situations [13].

Table 1. Statistics of the three datasets used throughout this paper, including the number of queries in the train/dev/test set, the number of candidate documents, and the average number of tokens per query/document.

Dataset	N train	N dev	N test	N doc	Avg. Q len.	Avg. D len.
OBQA	4,957	500	500	1,326	13.71	9.49
ReSQ	87,599	11,426	N/A	101,957	10.38	160.62
ReNQ	N/A	N/A	74,097	239,013	9.09	146.16

Table 2. Examples of queries, answer sentences, and contexts in the three datasets.

Dataset	Query	Answer sentence	Context
ReSQ	To whom did the Virgin Mary allegedly appear in 1858 in Lourdes France?	It is a replica of the grotto at Lourdes, France where the Virgin Mary reputedly appeared to Saint Bernadette Soubirous in 1858	... a Marian place of prayer and reflection. It is a replica of the grotto at Lourdes, France where the Virgin Mary reputedly appeared to Saint Bernadette Soubirous in 1858. At the end of the main drive ...
ReNQ	Who sings the song i don't care i love it	In its chorus, Icona Pop and Charli XCX shout in unison "I don't care / I love it"	... breaking up with an older boyfriend. In its chorus, Icona Pop and Charli XCX shout in unison "I don't care/I love it". Critics compared the song's breakup ...
OBQA	Tadpoles start their lives as Water animals	Tadpole changes into a frog	N/A

Efforts have been made to use traditional IR for evidence retrieval and neural IR for evidence reranking [3,18,21,27]. However, *always* relying on traditional IR for evidence retrieval may miss useful evidence that does not have large lexical overlap with the query.

Motivated by these works, in this paper we propose a hybrid evidence retrieval direction for first-stage retrieval, in which we learn *when* to use traditional IR vs. neural IR. As our results show, this yields a more accurate retrieval component that also has a lower runtime overhead than neural methods.

3 Datasets and Evaluation Measures

We conduct our probing analyses and retrieval experiments on three QA-related retrieval datasets. One of these datasets comes from the science domain; the other two are open domain. More statistics and examples of these datasets are shown in Table 1 and 2, and we describe them below.

OpenBookQA: The OpenBookQA dataset [16] (abbreviated to *OBQA* from now on) addresses a multiple-choice QA task in the science domain. Each correct answer is jointly annotated with one key evidence sentence (or *justification*) that supports its correctness. The justification comes from a knowledge base of 1326 sentences. In this paper, we construct a corpus of 1326 documents from these sentences. Further, for each question, we concatenate the question and the correct answer choice to form the *query*, and retrieve the gold justification (or *target document*) for that query from the corpus of 1326 documents.

ReQA SQuAD: The ReQA SQuAD dataset [1] (abbreviated to *ReSQ*) is a sentence-level retrieval dataset converted from the SQuAD reading comprehension dataset [23]. In the original SQuAD reading comprehension task, the answers to questions must be extracted from sentences in a set of provided paragraphs. The ReQA SQuAD dataset uses the questions in SQuAD as the queries, and converts all paragraphs to single sentences. The goal of this retrieval task is to retrieve the sentence that contains the correct answer from all the sentences generated from all the paragraphs. Since some answer sentences are meaningless without the surrounding context, each candidate sentence is accompanied by its original paragraph as the context.

ReQA NQ: The ReQA NQ dataset [1] (abbreviated to *ReNQ*) is similarly converted from another reading comprehension task – Natural Questions [14] – following the same process as ReQA SQuAD. Similarly, each query is a question and each target document is a sentence/context pair, where the context is the paragraph that contains the gold justification.

Table 3. Results of the probing tasks on two datasets. We report mean average precision (MAP) (higher is better) and perplexity (PPL) (lower is better) scores for the gold terms to be predicted. We report separate scores for terms in the query, and terms that occur *only* in the justification fact (mean and stdev across 5 random seeds).

Dataset	Task	Query MAP	Query PPL	Fact MAP	Fact PPL
OBQA	USE-QA embd, gold label	0.306 ±0.01	1.709 ±0.02	0.154 ±0.01	1.188 ±0.01
	tf-idf embd, gold label	0.458 ±0.01	1.558 ±0.01	0.098 ±0.00	1.334 ±0.01
	Random embd, gold label	0.053 ±0.02	3.640 ±0.10	0.031 ±0.01	3.294 ±1.63
	USE-QA embd, rand label	0.085 ±0.00	1.974 ±0.02	0.043 ±0.00	1.557 ±0.02
ReSQ	USE-QA embd, gold label	0.139 ±0.01	1.944 ±0.01	0.147 ±0.00	2.134 ±0.00
	tf-idf embd, gold label	0.142 ±0.04	1.828 ±0.00	0.127 ±0.01	2.043 ±0.00
	Random embd, gold label	0.091 ±0.00	7.419 ±3.50	0.093 ±0.00	3.478 ±0.15
	USE-QA embd, rand label	0.122 ±0.02	1.909 ±0.00	0.124 ±0.00	2.013 ±0.00

4 Understanding the Behavior of Neural IR Through Lexical Probing

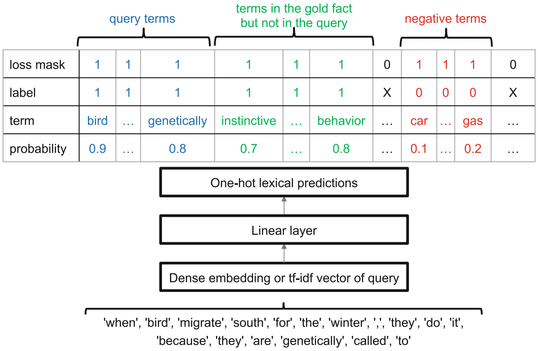
Our key hypothesis is that neural IR methods are better at modeling the lexical chasm between queries and evidence sentences than traditional IR, whereas traditional IR captures explicit lexical overlap better. We design a lexical probe and control tasks to investigate this.

4.1 Task Overview

Figure 1 summarizes our lexical probe, with an example from OBQA.

Probe Input: The probe starts by generating a representation of the input query. This representation is either: (a) the tf-idf vector of the query, generated using *scikit-learn* [20], or (b) the query embedding generated by USE-QA [28].

Linear Layer: This vector is fed to a linear layer, with input size N_d and output size N_v, to predict the terms (i.e., unique words) in the query *and* in the gold fact. N_v is the size of vocabulary $\mathcal{V}$, where $\mathcal{V}$ is the set of all terms in the dataset. Each number in the output is the predicted probability of that particular term being in the query/gold fact. Note that the input embedding/vector is not changed during the training of the probe task. Thus, if the neural embedding contains meaningful information about the gold fact, it should perform better than tf-idf on predicting the terms that are only in the gold fact.


query terms
terms in the gold fact but not in the query
negative terms
One-hot lexical predictions
Linear layer
Dense embedding or tf-idf vector of query
'when', 'bird', 'migrate', 'south', 'for', 'the', 'winter', ',', 'they', 'do', 'it', 'because', 'they', 'are', 'genetically', 'called', 'to'

loss mask	1	1	1	1	1	1	0	1	1	1	0
label	1	1	1	1	1	1	X	0	0	0	X
term	bird	…	genetically	instinctive	…	behavior	…	car	…	gas	…
probability	0.9	…	0.8	0.7	…	0.8	…	0.1	…	0.2	…

Fig. 1. Probe task overview: the linear probe is trained to predict the terms in the query and in the gold fact from the entire vocabulary, given either the input embedding or the tf-idf vector of the query. This probe investigates the capability of the query representation to predict both lexical overlap (i.e., terms from the query), as well as its ability to bridge the lexical chasm between queries and supporting facts (i.e., predict terms that exist in the fact and not in the query). A loss mask is used to make sure the loss is only computed on certain terms during training.

Training Label and Loss: For each query q_i, we use $\mathcal{P}_i$ to indicate the set of all terms in $\{q_i, \text{gold\_fact}(q_i)\}$. The training label is a one-hot vector of size N_v, where the values for the terms in $\mathcal{P}_i$ are 1, and the rest of the entries are 0. However, since the terms in $\mathcal{P}_i$ are considerably fewer than the whole vocabulary, there will be many more 0s than 1s in this label vector, causing label imbalance. Therefore, we construct a set of negative terms $\mathcal{N}_i$, which contains terms that are randomly sampled from the vocabulary $\mathcal{V}$ but not in $\mathcal{P}_i$. The size of $\mathcal{N}_i$ also equals to the size of $\mathcal{P}_i$. The loss is only computed on the terms in $\mathcal{P}_i \cup \mathcal{N}_i$ instead of the whole vocabulary $\mathcal{V}$. The total loss of each query q_i is summarized by: $L = -\sum_{j \in \mathcal{P}_i \cup \mathcal{N}_i} [y_j \log \hat{y_j} + (1 - y_j) \log(1 - \hat{y_j})]$, where $y_j \in \{0, 1\}$ is the label, and $\hat{y_j} \in (0, 1)$ is the predicted probability of the corresponding term.

4.2 Control Tasks

We designed two control tasks to check whether the information necessary for prediction is contained in the query representation and not in the linear layer [9]:

Random Embedding (rand embd): This probe replaces the neural embedding with a randomly-generated embedding. If the query representation encodes useful information, this probe should perform much worse than the one using the neural representation.

Rand Label (rand label): In this experiment we randomly replace the target terms in both training and testing. For example, we replace the terms to be predicted for query i with terms from a randomly-selected query j: $\{q_j, \text{gold\_fact}(q_j)\}$. This is to examine whether it is possible for the linear probe to learn non-sensical associations between random (embedding, target terms) pairs.

4.3 Probe Results

Table 3 lists the results of these probing tasks. We draw several observations:

(1) With minor exceptions, the two actual probes perform better than the two control tasks. This confirms that there is indeed signal that is encoded in the query representations, and this is what the linear probe classifier exploits.

(2) The probe that relies on the neural query representation obtains higher fact MAP (and lower fact PPL) than the probe that uses the tf-idf representation. This indicates that the neural representation does indeed contain information that helps bridge queries, answers, and supporting facts. On the other hand, the tf-idf probe has higher query MAP (and lower query PPL) than the neural probe. This confirms that the traditional IR representation is better at capturing explicit lexical overlap with the query than the neural one. All in all, these observations suggest that these two retrieval directions are better at different things.

5 Hybrid Retrieval Approach

5.1 Individual IR Models

The hybrid approach proposed builds from (and is compared against) the following individual retrieval models. Note that these approaches were chosen because they had the best performance on these datasets. For example, USE-QA consistently performed better than BERT.

BM25: We use the Lucene 6.4.0[2] Java implementation of BM25 [24] as the "traditional" IR method. For OBQA, each document is one sentence in the knowledge base corpus (1326 sentences in total). In ReSQ and ReNQ, each document is constructed by concatenating the candidate answer sentence and its context (so that each candidate answer sentence appears twice in the document).

BERT: For this method we fine-tune a pretrained BERT-base model [7,26]. We use the BERT retriever in the representation-based manner: the query q_i and the document d_j are encoded using the [CLS] embedding of BERT as h_i^q and h_j^d. Then the relevance score of q_i and d_j are obtained by $Rel(q_i, f_j) = h_i^q \cdot h_j^d$. For ReSQ and ReNQ, each document is composed of the candidate answer sentence and its context. We concatenate them and separate them with the [SEP] token. Therefore, the input of each document is "[CLS] candidate answer sentence [SEP] context sentences [SEP]".

USE-QA: The USE-QA retriever [28] has separate encoders for the query and document. The query encoder is a transformer-based model, producing a 512-dimension embedding as the query representation. The document encoder has a transformer-based model to encode the answer sentence and a Convolutional Neural Network (CNN)-based model to encode the context. A single 512-dimension embedding is produced as the document representation. Finally, the relevance score is computed as the dot product of the query embedding and the document embedding. USE-QA is pre-trained on large scale retrieval tasks and, as used in [1], we do not fine-tune it in the retrieval tasks.

5.2 Are Neural IR Methods Generally Better Than Traditional IR?

The probe introduced in Sect. 4 indicates that neural and traditional IR methods have different behaviors. But what impact does that have in practice, with respect to overall performance? To answer this question, we performed a comparison that aims to understand if transformer-based retrieval methods are better overall than traditional IR. Due to space limitations, we discuss here results from the best individual models in each class: BM25 for traditional IR, and USE-QA for neural IR (We observed similar behavior from tf-idf and BERT.). We use two datasets: one domain-specific (OBQA) and one open-domain (ReSQ).

[2] https://lucene.apache.org.

Figure 2 summarizes this comparison between BM25 and USE-QA on the dev partitions of OBQA and on a 10,000-query subset of ReSQ training partition. Here, we consider a model better than the other when it yields a better ranking for the correct justification. We draw two observations from this analysis:

(1) No approach is consistently better than the other. Overall, BM25 is at least as good as USE-QA in 293 queries out of 500 queries in OBQA dev set, and 7603 queries out of 10000 randomly sampled queries in ReSQ train set. This is further motivation for a hybrid approach.

(2) Importantly, there is immediate signal to differentiate between the two situations. When BM25 performs better than or similarly to USE-QA, the top BM25 score (after the softmax normalization) tends to be in the 0.8 to 1 range. In contrast, when there is little lexical overlap between question and justification indicated by low BM25 scores, e.g., below 0.2, USE-QA performs considerably better. This supports the intuition that USE-QA can capture lexical differences between question and justification when present.

5.3 Hybrid Retrieval Model

Motivated by the previous observations, we propose a hybrid evidence retrieval method that uses a routing classifier to direct an incoming question to either the BM25 retriever or the USE-QA retriever based on simple statistics that can be extracted efficiently. The key intuition behind our hybrid strategy is that we can estimate the optimal retrieval method based on the top answers retrieved by traditional IR. In particular, if these answers receive a high score from the traditional IR method, it indicates that the current scenario is driven by lexical overlap, and traditional IR is likely to do better; the opposite is true otherwise.

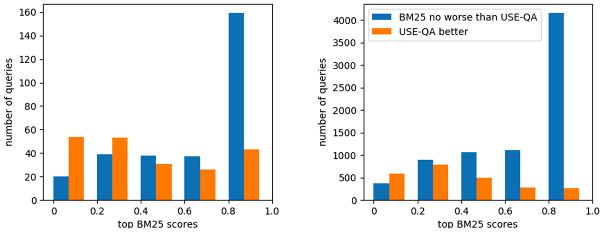


Fig. 2. Histograms of queries in OBQA dev (left) and a randomly sampled subset of ReSQ train (right) where BM25 is no worse than USE-QA (blue) or where USE-QA is better (orange). The x axis is the top BM25 score after a softmax is applied to the BM25 scores of the top 64 sentences. (Color figure online)

We propose two variants for the routing classifier:

Hybrid (1-param): This classifier uses a single parameter: a threshold on the normalized BM25 score of top document retrieved by the BM25 method.[3] If the top 1 score is higher than this threshold, the classifier routes the question to BM25; otherwise it sends it to USE-QA. This is a simple implementation of the intuition above – if the top normalized BM25 score is high, then it is likely that there is a candidate document that has a large lexical overlap with the query, and which is probably a correct justification. On the other hand, if the top normalized BM25 score is low, it is either because: (a) there is no document that has a large lexical overlap with the query, or (b) because there are multiple candidate documents that have high BM25 scores (and they are squished during normalization). In either of these scenarios BM25 is unlikely to identify the gold document, and, therefore, USE-QA should be selected. The value of this threshold is determined by performing a grid search on the dev partition, with the threshold ranging from 0 to 1 with an interval of 0.1.

Hybrid (BM25): This classifier is a generalization of the above. That is, instead of relying solely on the top retrieved document, this classifier extracts features from the top k. In particular, for each query, we construct a feature vector f and use a logistic regression (LR) classifier that takes f to predict whether to use BM25 or USE-QA. The i^{th} feature of f is computed as $f_i = mean(S[0:2^i])$, where S are the top BM25 scores ranked in the descending order (after softmax normalization). In this paper we use i up to 6 (i.e., use up to top 64 BM25 scores). For example, feature 2 averages the BM25 scores of the top 4 documents retrieved by the traditional IR method. This strategy allows the classifier to take advantage of more documents when needed, but also focus on the top result(s) when they are sufficiently predictive.

Note that all *Hybrid* approaches choose *either* one of the individual models. USE-QA is *not* used as a reranking method on top of BM25, because USE-QA is applied to all documents instead of the top documents retrieved by BM25.

6 Results

In this section, we empirically evaluate the proposed evidence retrieval methods. We use the mean reciprocal rank (MRR) score [25] of the correct evidence sentence (or target document) in the test dataset as the evaluation measure.

Since ReSQ only provides training and developments partitions, we randomly sample 10,000 queries from the training data and use them for development, and use the original development set of ReSQ as test. ReNQ does not provide training/development/test partitions; for this dataset we use 5-fold cross-validation, sampling 10,000 queries from one fold as the development data in each split, and using the remaining folds as test.

[3] We normalize this score by applying a softmax layer to the BM25 scores of the top k ($k = 64$ in this paper) documents.

For all datasets, USE-QA is used without fine-tuning as proposed in [1]. For the BERT retriever, we fine-tune it on the training data of OBQA and ReSQ. For all hybrid retrievers, we tune their routing classifiers on the respective development partitions. For ReSQ, we further divide the development set into 5 splits (2,000 queries in each split) and tune 5 routing classifiers and evaluate them separately on the full test set to make sure the results are robust to different development sets.

6.1 Individual Vs Hybrid Retrievers

Table 4 shows the MRR scores of the individual retrieval methods compared to the hybrid ones, on the three datasets. We draw several observations from this:

(1) Most hybrid strategies outperform the individual retrieval methods, as well as the naive strategy that simply sums up the scores of two individual models, and uses the sum for ranking. *Hybrid (1-param)* and *Hybrid (BM25)* are statistically significantly better than BM25 and USE-QA on ReSQ and ReNQ under a bootstrap resampling significance analysis (10,000 samples, p-value $< 10^{-5}$). On OBQA, *Hybrid (1-param)* and *Hybrid (BM25)* are statistically significantly better than BM25 under the same bootstrap resampling significance analysis, but there is no significant difference between the hybrid methods and USE-QA. This demonstrates that transformer-based and IR-based methods capture complementary information, and the distinction of when to use one vs. another is learnable. Table 5 lists several runtime statistics of our best classifier, *Hybrid (BM25)*, which support this observation. The first two rows indicate that the routing classifier uses both individual retrievers, with around 60% (OBQA) or 86% (ReSQ) of questions being routed to BM25. The next four rows indicate that, on average, the hybrid approach improves over both individual methods especially on ReSQ and ReNQ.

Table 4. Mean reciprocal rank (MRR) scores of the retrieval methods investigated on the three QA datasets. The *BM25 + USE-QA* method sums up the scores produced by BM25 and USE-QA, and uses that score for ranking. * indicates that *Hybrid (BM25)* is statistically significantly better than both USE-QA and BM25 (bootstrap resampling with 10,000 iterations; p-value $< 10^{-5}$). *Hybrid-NN (BM25)* uses approximate Nearest Neighbor for USE-QA in the hybrid method with 20 search trees. The *Ceiling* method always selects the best individual model (USE-QA or BM25) for each query by their ranking of the gold justification.

	BM25	BERT	USE-QA	BM25 + USE-QA	Hybrid (1-param)	Hybrid (BM25)	Hybrid-NN (BM25)	Ceiling
OBQA	0.522	0.557	0.610	0.550	**0.611**	0.596	N/A	0.69
ReSQ	0.645	0.260	0.520	0.647	0.656	**0.657***	0.656*	0.71
ReNQ	0.293	N/A	0.223	0.290	0.301	**0.303***	0.298*	0.39

Table 5. Routing statistics for the routing classifier that trains a logistic regression model using features extracted from the top 64 BM25 documents.

	OBQA	ReSQ	ReNQ
n samples routed to BM25	306	49, 270	260, 640
n samples routed to useQA	194	7, 860	59, 845
Samples w/improved rankings vs. BM25	129	4, 095	33, 078
Samples w/worse rankings vs. BM25	47	3, 257	25, 562
Samples w/improved rankings vs. useQA	61	21, 488	146, 020
Samples w/worse rankings vs. useQA	93	8, 640	84, 839

(2) While *Hybrid (BM25)* outperforms the simpler *Hybrid (1-param)*, the difference is not statistically significant.[4] This further suggests that simpler approaches work in this case. The routing decision can be approximated with a single parameter (a threshold on the BM25 score), applied to a single justification that is efficiently extracted by IR.

6.2 Runtime Analysis

A further advantage of our hybrid approach is improved runtime over neural methods, because a considerable number of queries are routed to a traditional, fast IR engine. To investigate this, we measure the processing time per query using BM25, USE-QA and various hybrid retrievers and calculate the total time usage of these retrieval methods. The processing time per query is measured as:

(1) For BM25, we measure the time of parsing the query, searching the top k ($k = 1400$ for OBQA, and 2000 for ReSQ and ReNQ) documents, and sorting the retrieved documents by the BM25 scores.

(2) For USE-QA, we measure the query processing time (including query tokenization and the embedding generation of USE-QA[5]), searching the top k

Table 6. Runtime comparison of BM25, USE-QA and hybrid retrievers on the corresponding test partitions. All times are the total times in seconds on all test queries.

	BM25	USE-QA	BM25 + USE-QA	Hybrid (1-param)	Hybrid (BM25)	Hybrid-NN (BM25)
OBQA	1.38	19.74	21.23	20.85	8.95	N/A
ReSQ	179.56	3241.73	3476.99	922.28	625.74	593.82
ReNQ	1547.56	26722.63	28929.47	9513.97	6565.21	3696.11

[4] Bootstrap resampling with 10,000 samples, p-value < 0.13.

[5] The batch size is set to 1 when generating the embedding, for a fair comparison with BM25, and because in a real use case the queries may not arrive in batch.

(1326 for OBQA and 2000 for ReSQ and ReNQ) documents, and sorting them by the scores. We run this experiment using Tensorflow on Google Colab with GPU.

(3) For hybrid models, the processing time of each query is the sum of: (1) the BM25 processing time (2) the runtime of the routing classifier and (3) the processing time of USE-QA if USE-QA is selected for that query.

Table 6 shows the total processing time of all queries using different retrieval methods. The table indicates that USE-QA is more than 15 times slower than BM25 on all datasets. Further, the hybrid approach reduces that gap while still allowing for the benefits of the neural IR when needed: *Hybrid (BM25)* is 2.2 times faster than USE-QA in OBQA, and 5.2 times faster in ReSQ. Our hybrid method is also significantly faster than *BM25 + USE-QA*, which uses both BM25 and the neural retriever on every query [17].

7 Conclusion

We argue that transformer network-based approaches do not always outperform IR methods for evidence retrieval for QA. We validate this observation with an empirical analysis, and with a lexical probing task where two probes were trained to predict words in the gold evidence texts. The first probe, trained on the tf-idf vector of the query, tends to predict words that exist in the original query (thus emphasizing lexical overlap), whereas the second probe, trained on top of the query's neural embedding, predicts more words in the evidence text that do not exist in the query (bridging the lexical differences between these texts).

Learning from this analysis, we introduced a routing classifier that learns when to direct incoming questions to traditional or neural IR methods for evidence retrieval. The routing classifier is trained using very simple statistics, which can be extracted from the top candidate evidence sentences produced by traditional IR. We showed that this hybrid evidence retrieval generally performs better than either individual retrieval strategy on three QA datasets. Further, we showed that this routing classifier can be approximated with nearly the same performance with a 1-parameter model (a threshold over the IR score of the top evidence sentence retrieved by BM25), which simplifies real-world applications of our approach. Lastly, we show that our routing classifier is considerably faster than USE-QA, with runtime improvements of up to 5 times.

References

1. Ahmad, A., Constant, N., Yang, Y., Cer, D.: Reqa: an evaluation for end-to-end answer retrieval models. In: Proceedings of the 2nd Workshop on Machine Reading for Question Answering, pp. 137–146 (2019)
2. Berger, A., Caruana, R., Cohn, D., Freitag, D., Mittal, V.: Bridging the lexical chasm: statistical approaches to answer-finding. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 192–199. ACM (2000)

3. Chen, R.C., Spina, D., Croft, W.B., Sanderson, M., Scholer, F.: Harnessing semantics for answer sentence retrieval. In: Proceedings of the Eighth Workshop on Exploiting Semantic Annotations in Information Retrieval, pp. 21–27. ACM (2015)
4. Cohen, D., Ai, Q., Croft, W.B.: Adaptability of neural networks on varying granularity ir tasks. arXiv preprint arXiv:1606.07565 (2016)
5. Conneau, A., Kruszewski, G., Lample, G., Barrault, L., Baroni, M.: What you can cram into a single $ \ &!# vector: probing sentence embeddings for linguistic properties. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2126–2136 (2018)
6. Dehghani, M., Zamani, H., Severyn, A., Kamps, J., Croft, W.B.: Neural ranking models with weak supervision. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 65–74. ACM (2017)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)
8. Guo, J., et al.: A deep look into neural ranking models for information retrieval. Inf. Process. Manag. **57**(6), 102067 (2020)
9. Hewitt, J., Liang, P.: Designing and interpreting probes with control tasks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2733–2743 (2019)
10. Hewitt, J., Manning, C.D.: A structural probe for finding syntax in word representations. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4129–4138 (2019)
11. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Advances in Neural Information Processing Systems, pp. 2042–2050 (2014)
12. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 2333–2338. ACM (2013)
13. Iyyer, M., Manjunatha, V., Boyd-Graber, J., Daumé III, H.: Deep unordered composition rivals syntactic methods for text classification. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). vol. 1, pp. 1681–1691 (2015)
14. Kwiatkowski, T., et al.: Natural questions: a benchmark for question answering research. Trans. Assoc. Comput. Linguist. **7**, 453–466 (2019)
15. Lee, K., Chang, M.W., Toutanova, K.: Latent retrieval for weakly supervised open domain question answering. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 6086–6096 (2019)
16. Mihaylov, T., Clark, P., Khot, T., Sabharwal, A.: Can a suit of armor conduct electricity? a new dataset for open book question answering. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2381–2391 (2018)

17. Mitra, B., Diaz, F., Craswell, N.: Learning to match using local and distributed representations of text for web search. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1291–1299 (2017)
18. Nie, Y., Wang, S., Bansal, M.: Revealing the importance of semantic retrieval for machine reading at scale. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2553–2566 (2019)
19. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)
20. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
21. Pirtoaca, G.S., Rebedea, T., Ruseti, S.: Answering questions by learning to rank-learning to rank by answering questions. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2531–2540 (2019)
22. Qiao, Y., Xiong, C., Liu, Z., Liu, Z.: Understanding the behaviors of bert in ranking. arXiv preprint arXiv:1904.07531 (2019)
23. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250 (2016)
24. Robertson, S., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. Found. Trends Inf. Retr. **3**(4), 333–389 (2009). https://doi.org/10.1561/1500000019
25. Voorhees, E.: The TREC-8 question answering track report. In: Proceedings of the 8th Text Retrieval Conference, pp. 77–82 (1999)
26. Wolf, T., et al.: Huggingface's transformers: state-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019)
27. Yang, W., et al.: End-to-end open-domain question answering with bertserini. NAACL HLT **2019**, 72 (2019)
28. Yang, Y., et al.: Multilingual universal sentence encoder for semantic retrieval. arXiv preprint arXiv:1907.04307 (2019)
29. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems, pp. 5753–5763 (2019)

Evaluating Multilingual Text Encoders for Unsupervised Cross-Lingual Retrieval

Robert Litschko[1(✉)], Ivan Vulić[2], Simone Paolo Ponzetto[1], and Goran Glavaš[1]

[1] Data and Web Science Group, University of Mannheim, Mannheim, Germany
{litschko,simone,goran}@informatik.uni-mannheim.de
[2] Language Technology Lab, University of Cambridge, Cambridge, UK
iv250@cam.ac.uk

Abstract. Pretrained multilingual text encoders based on neural Transformer architectures, such as multilingual BERT (mBERT) and XLM, have achieved strong performance on a myriad of language understanding tasks. Consequently, they have been adopted as a go-to paradigm for multilingual and cross-lingual representation learning and transfer, rendering cross-lingual word embeddings (CLWEs) effectively obsolete. However, questions remain to which extent this finding generalizes 1) to unsupervised settings and 2) for ad-hoc cross-lingual IR (CLIR) tasks. Therefore, in this work we present a systematic empirical study focused on the suitability of the state-of-the-art multilingual encoders for cross-lingual document and sentence retrieval tasks across a large number of language pairs. In contrast to supervised language understanding, our results indicate that for unsupervised document-level CLIR – a setup with no relevance judgments for IR-specific fine-tuning – pretrained encoders fail to significantly outperform models based on CLWEs. For sentence-level CLIR, we demonstrate that state-of-the-art performance can be achieved. However, the peak performance is not met using the general-purpose multilingual text encoders 'off-the-shelf', but rather relying on their variants that have been further specialized for sentence understanding tasks.

Keywords: Cross-lingual IR · Multilingual text encoders · Unsupervised IR

1 Introduction

Cross-lingual information retrieval (CLIR) systems respond to queries in a source language by retrieving relevant documents in another, target language. Their success is typically hindered by data scarcity: they operate in challenging low-resource settings without sufficient labeled training data, i.e., human relevance judgments, to build supervised models (e.g., neural matching models for pairwise retrieval [22,53]). This motivates the need for robust, resource-lean and unsupervised CLIR approaches.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 342–358, 2021.
https://doi.org/10.1007/978-3-030-72113-8_23

In previous work, Litschko et al. [27] have shown that language transfer through cross-lingual embedding spaces (CLWEs) can be used to yield state-of-the-art performance in a range of unsupervised ad-hoc CLIR setups. This approach uses very weak supervision (i.e., only a bilingual dictionary spanning 1K-5K word translation pairs), or even no supervision at all, in order to learn a mapping that aligns two monolingual word embedding spaces [19,45]. Put simply, this enables casting CLIR tasks as 'monolingual tasks in the shared (CLWE) space': at retrieval time both queries and documents are represented as simple aggregates of their constituent CLWEs. However, this approach, by limitations of static CLWEs, cannot capture and handle polysemy in the underlying text representations. *Contextual text representation models* alleviate this issue [28]. They encode occurrences of the same word differently depending on its surrounding context.

Such contextual representations are obtained via large models pretrained on large text collections through general objectives such as (masked) language modeling [16,30]. Multilingual text encoders pretrained on 100+ languages, such as mBERT [16] or XLM [14], have become a *de facto* standard for multilingual representation learning and cross-lingual transfer in natural language processing (NLP). These models demonstrate state-of-the-art performance in a wide range of supervised language understanding and language generation tasks [26,36], especially in zero-shot settings: a typical *modus operandi* is fine-tuning a pretrained multilingual encoder with task-specific data of a source language (typically English) and then using it directly in a target language.

It is unclear, however, whether these general-purpose multilingual text encoders can be used directly for ad-hoc CLIR without any additional supervision (i.e., relevance judgments). Further, can they outperform unsupervised CLIR approaches based on static CLWEs [27]? How do they perform depending on the (properties of the) language pair at hand? How can we encode useful semantic information using these models, and do different "encoding variants" (see later Sect. 3) yield different retrieval results? Are there performance differences in unsupervised sentence-level versus document-level CLIR tasks? Finally, can we boost performance by relying on sentence encoders that are specialized towards dealing with sentence-level understanding in particular? In order to address these questions, we present a systematic empirical study and profile the suitability of state-of-the-art pretrained multilingual encoders for different CLIR tasks and diverse language pairs. We evaluate two state-of-the-art general-purpose pretrained multilingual encoders, mBERT [16] and XLM [14] with a range of encoding variants, and also compare them to CLIR approaches based on static CLWEs, and specialized multilingual sentence encoders. Our key contributions can be summarized as follows:

(1) We empirically validate that, without any task-specific fine-tuning, multilingual encoders such as mBERT and XLM fail to outperform CLIR approaches based on static CLWEs. Their performance also crucially depends on how one encodes semantic information with the models (e.g., treating them as sentence/document encoders directly versus averaging over constituent words and/or subwords). We also show that there is no “one-size-fits-all” approach, and the results are task- and language-pair-dependent.

(2) We provide a first large-scale comparative evaluation of state-of-the art pretrained multilingual encoders on unsupervised document-level CLIR. We also empirically show that encoder models specialized for sentence-level understanding substantially outperform general-purpose models (mBERT and XLM) on sentence-level CLIR tasks.

2 Related Work

Self-supervised Pretraining and Transfer Learning. Recently, research on universal sentence representations and transfer learning has gained much traction. InferSent [13] transfers the encoder of a model trained on natural language inference to other tasks, while USE [8] extends this idea to a multi-task learning setting. More recent work explores self-supervised neural Transformer-based [44] models based on (causal or masked) language modeling (LM) objectives such as BERT [16], RoBERTa [30], GPT [5,37], and XLM [14].[1] Results on benchmarks such as GLUE [47] and SentEval [12] indicate that these models can yield impressive (sometimes human-level) performance in supervised Natural Language Understanding (NLU) and Generation (NLG) tasks. These models have become *de facto* standard and omnipresent text representation models in NLP. In supervised monolingual IR, self-supervised LMs have been employed as contextualized word encoders [32], or fine-tuned as pointwise and pairwise rankers [33].

Multilingual Text Encoders based on the (masked) LM objectives have also been massively adopted in multilingual and cross-lingual NLP and IR applications. A multilingual extension of BERT (mBERT) is trained with a shared subword vocabulary on a single multilingual corpus obtained as concatenation of large monolingual data in 104 languages. The XLM model [14] extends this idea and proposes natively cross-lingual LM pretraining, combining causal language modeling (CLM) and translation language modeling (TLM).[2] Strong performance of these models in supervised settings is confirmed across a range of tasks on multilingual benchmarks such as XGLUE [26] and XNLI [15]. However, recent work [6,39] has indicated that these general-purpose models do not yield strong results when used as out-of-the-box text encoders in an unsupervised transfer learning setup. We further investigate these preliminaries, and confirm this finding also for unsupervised ad-hoc CLIR tasks.

[1] Note that self-supervised learning can come in different flavors depending on the training objective [10], but language modeling objectives still seem to be the most popular choice.

[2] In CLM, the model is trained to predict the probability of a word given the previous words in a sentence. TLM is a cross-lingual variant of standard masked LM (MLM), with the core difference that the model is given pairs of parallel sentences and allowed to attend to the aligned sentence when reconstructing a word in the current sentence.

Multilingual text encoders have already found applications in document-level CLIR. Jiang et al. [22] use mBERT as a matching model by feeding pairs of English queries and foreign language documents. MacAvaney et al. [31] use mBERT in a zero-shot setting, where they train a retrieval model on top of mBERT on English relevance data and apply it on a different language. However, prior work has not investigated unsupervised CLIR setups, and a systematic comparative study focused on the suitability of the multilingual text encoders for diverse ad-hoc CLIR tasks and language pairs is still lacking.

Specialized Multilingual Sentence Encoders. An extensive body of work focuses on inducing multilingual encoders that capture sentence meaning. In [2], the multilingual encoder of a sequence-to-sequence model is shared across languages and optimized to be language-agnostic, whereas Guo et al. [20] rely on a dual Transformer-based encoder architectures instead (with tied/shared parameters) to represent parallel sentences. Rather than optimizing for translation performance directly, their approach minimizes the cosine distance between parallel sentences. A ranking softmax loss is used to classify the correct (i.e., aligned) sentence in the other language from negative samples (i.e., non-aligned sentences). In [50], this approach is extended by using a bidirectional dual encoder and adding an additive margin softmax function, which serves to push away non-translation-pairs in the shared embedding space. The dual-encoder approach is now widely adopted [18,20,39,51,56], and yields state-of-the-art multilingual sentence encoders which excel in sentence-level NLU tasks.

Other recent approaches propose input space normalization, and re-aligning mBERT and XLM with parallel data [6,56], or using a teacher-student framework where a student model is trained to imitate the output of the teacher network while preserving high similarity of translation pairs [39]. In [51], authors combine multi-task learning with a translation bridging task to train a universal sentence encoder. We benchmark a series of representative sentence encoders; their brief descriptions are provided in Sect. 3.3.

CLIR Evaluation and Application. The cross-lingual ability of mBERT and XLM has been investigated by probing and analyzing their internals [23], as well as in terms of downstream performance [34,49]. In CLIR, these models as well as dedicated multilingual sentence encoders have been evaluated on tasks such as cross-lingual question-answer retrieval [51], bitext mining [58,59], and semantic textual similarity (STS) [21,25]. Yet, the models have been primarily evaluated on sentence-level retrieval, while classic ad-hoc (unsupervised) document-level CLIR has not been in focus. Further, previous work has not provided a large-scale comparative study across diverse language pairs and with different model variants, nor has tried to understand and analyze the differences between sentence-level and document-level tasks. In this work, we aim to fill these gaps.

3 Multilingual Text Encoders

We provide an overview of all multilingual models in our evaluation. We discuss general-purpose multilingual text encoders (Sect. 3.2), as well as specialized multilingual sentence encoders in Sect. 3.3. For completeness, we first briefly describe static CLWEs (Sect. 3.1).

3.1 CLIR with (Static) Cross-Lingual Word Embeddings

We assume a query Q_{L_1} issued in a source language L_1, and a document collection of N documents D_{i,L_2}, $i = 1,\ldots,N$ in a target language L_2. Let $d = \{t_1, t_2, \ldots, t_{|D|}\} \in D$ be a document with $|D|$ terms t_i. CLIR with static CLWEs represents queries and documents as vectors $\overrightarrow{Q}, \overrightarrow{D} \in \mathbb{R}^d$ in a d-dimensional shared embedding space [27,46]. Each term is represented independently with a pre-computed static embedding vector $\overrightarrow{t_i} = emb\,(t_i)$. There exist a range of methods for inducing shared embedding spaces with different levels of supervision, such as parallel sentences, comparable documents, small bilingual dictionaries, or even methods without any supervision [41]. Given the shared CLWE space, both query and document representations are obtained as aggregations of their term embeddings. We follow Litschko et al. [27] and represent documents as the weighted sum of their terms' vectors, where each term's weight corresponds to its inverse document frequency (idf) : $\overrightarrow{d} = \sum_{i=1}^{N_d} idf(t_i^d) \cdot \overrightarrow{t_i^d}$. During retrieval documents are ranked according to the cosine similarity to the query.

3.2 Multilingual (Transformer-Based) Language Models: mBERT and XLM

Massively multilingual pretrained neural language models such as mBERT and XLM can be used as a dynamic embedding layer to produce contextualized word representations, since they share a common input space on the subword level (e.g. word-pieces, byte-pair-encodings) across all languages. Let us assume that a term (i.e., a word-level token) is tokenized into a sequence of K subword tokens ($K \geq 1$; for simplicity, we assume that the subwords are word-pieces (wp)): $t_i = \left\{wp_{i,k}\right\}_{k=1}^{K}$. The multilingual encoder then produces contextualized subword embeddings for the term's K constituent subwords $\overrightarrow{wp_{i,k}}$, $k = 1,\ldots,K$, and we can aggregate these subword embeddings to obtain the representation of the term t_i: $\overrightarrow{t_i} = \psi\left(\{\overrightarrow{wp_{i,k}}\}_{k=1}^{K}\right)$, where the function $\psi()$ is the aggregation function over the K constituent subword embeddings. Once these term embeddings $\overrightarrow{t_i}$ are obtained, we follow the same CLIR setup as with CLWEs in Sect. 3.1.

Static Word Embeddings from Multilingual Transformers. We first use multilingual transformers (mBERT and XLM) in two different ways to induce static word embedding spaces for all languages. In a simpler variant, we feed terms into the encoders *in isolation* (**ISO**), that is, without providing any surrounding context for the terms. This effectively constructs a static word embedding table similar to what is done in Sect. 3.1, and allows the CLIR model (Sect. 3.1) to operate at a non-contextual word level. An empirical CLIR comparison between ISO and CLIR operating on CLWEs [27] then effectively quantifies how well multilingual encoders (mBERT and XLM) encode word-level representations.

In a more elaborate variant we do leverage the contexts in which the terms appear, constructing *average-over-contexts* embeddings (**AOC**). For each term t we collect a set of sentences $s_i \in \mathcal{S}_t$ in which it occurs. We use the full set of Wikipedia sentences $\mathcal{S}$ to sample sets of contexts $\mathcal{S}_t$ for vocabulary terms. For a given sentence s_i let j denote the position of t's first occurrence. We then transform s_i with mBERT or XLM as the encoder, $enc(s_i)$, and extract the contextualized embedding of t via *mean-pooling*, i.e., by averaging embeddings of its constituent subwords, $\psi\left(\{\overrightarrow{wp_{j,k}}\}_{k=1}^{K}\right) = 1/K \cdot \sum_{k=1}^{K} \overrightarrow{wp_{j,k}}$. For each vocabulary term, we obtain $N_t = min(|\mathcal{S}_t|, \tau)$ contextualized vectors, with $|\mathcal{S}_t|$ as the number of Wikipedia sentences containing t and τ as the maximal number of sentence samples for a term. The final static embedding of t is then simply the average over the N_t contextualized vectors.

The obtained static AOC and ISO embeddings, despite being induced with multilingual encoders, however, did not appear to be well-aligned across languages [6,29]. We evaluated the static ISO and AOC embeddings induced for different languages with multilingual encoders (mBERT and XLM), on the bilingual lexicon induction (BLI) task [19]. We observed poor BLI performance, suggesting that further projection-based alignment of respective monolingual ISO and AOC spaces is required. To this end, we use the standard Procrustes method [1,43] to align the embedding spaces of two languages, with bilingual dictionaries from [19] as the supervision guiding the alignment. Concretely, for each language pair in our experiments we project the AOC (ISO) embeddings of the source language to the AOC (ISO) space of the target language.

Direct Text Embedding with Multilingual Transformers. In both AOC and ISO, we use the multilingual (contextual) encoders to obtain the static embeddings for word types (i.e., terms): we can then leverage in exactly the same ad-hoc retrieval setup (Sect. 3.1) in which CLWEs had previously been evaluated [27]. In an arguably more straightforward approach, we also use pretrained multilingual Transformers (i.e., mBERT or XLM) to directly encode the whole input text (**SEMB**). We encode the input text by averaging the contextualized representations of all terms in the text (we again compute the weighted average, where the terms' IDF scores are used as weights, see Sect. 3.1). For SEMB, we take the contextualized representation of each term t_i to be the contextualized representation of its first subword token, i.e., $\overrightarrow{t_i} = \psi\left(\{\overrightarrow{wp_{i,k}}\}_{k=1}^{K}\right) = \overrightarrow{wp_{i,1}}$.[3]

3.3 Specialized Multilingual Sentence Encoders

Off-the-shelf multilingual Transformers (mBERT and XLM) have been shown to yield sub-par performance in unsupervised text similarity tasks; therefore, in order to be successful in semantic text (sentences or paragraph) comparisons,

[3] In our initial experiments taking the vector of the first term's subword consistently outperformed averaging vectors of all its subwords.

they first need to be fine-tuned on text matching (typically sentence matching) datasets [6,39,57]. Such encoders *specialized for semantic similarity* are supposed to encode sentence meaning more accurately, supporting tasks that require unsupervised (ad-hoc) semantic text matching. In contrast to mBERT and XLM, which contextualize (sub)word representations, these models directly produce a semantic embedding of the input text. We provide a brief overview of the models included in our comparative evaluation.

Language Agnostic SEntence Representations (LASER) [2] adopts a standard sequence-to-sequence architecture typical for neural machine translation (MT). It is trained on 223M parallel sentences covering 93 languages. The encoder is a multi-layered bidirectional LSTM and the decoder is a single-layer unidirectional LSTM. The 1024-dimensional sentence embedding is produced by max-pooling over the outputs of encoder's last layer. The decoder then takes the sentence embedding as additional input as each decoding step. The decoder-to-encoder attention and language identifiers on the encoder side are deliberately omitted, so that all relevant information gets 'crammed' into the fixed-sized sentence embedding produced by the encoder. In our experiments, we directly use the output of the encoder to represent both queries and documents.

Multilingual Universal Sentence Encoder (m-USE) is a general purpose sentence embedding model for transfer learning and semantic text retrieval tasks [51]. It relies on a standard dual-encoder neural framework [9,52] with shared weights, trained in a multi-task setting with an additional translation bridging task. For more details, we refer the reader to the original work. There are two pretrained m-USE instances available – we opt for the 3-layer Transformer encoder with average-pooling.

Language-Agnostic BERT Sentence Embeddings (LaBSE) [18] is another neural dual-encoder framework, also trained with parallel data. Unlike in LASER and m-USE, where the encoders are trained from scratch on parallel data, LaBSE training starts from a pretrained mBERT instance (i.e., a 12-layer Transformer network pretrained on the concatenated corpora of 100+ languages). In addition to the multi-task training objective of m-USE, LaBSE additionally uses standard self-supervised objectives used in pretraining of mBERT and XLM: masked and translation language modelling (MLM and TLM, see Sect. 2). For further model details, we refer the reader to the original work.

DISTIL [39] is a teacher-student framework for injecting the knowledge obtained through specialization for semantic similarity from a specialized monolingual transformer (e.g., BERT) into a non-specialized multilingual transformer (e.g., mBERT). It first specializes for semantic similarity a monolingual (English) teacher encoder M using the available semantic sentence-matching datasets for supervision. In the second, *knowledge distillation* step a pretrained multilingual student encoder $\widehat{M}$ is trained to mimic the output of the teacher model. For a given batch of sentence-translation pairs $\mathcal{B} = \{(s_j, t_j)\}$, the teacher-student distillation training minimizes the following loss:

$$\mathcal{J}(\mathcal{B}) = \frac{1}{|\mathcal{B}|}\sum_{j\in\mathcal{B}}\left[\left(M(s_j) - \widehat{M}(s_j)\right)^2 + \left(M(s_j) - \widehat{M}(t_j)\right)^2\right].$$

The teacher model M is Sentence-BERT [38], BERT specialized for embedding sentence meaning on semantic text similarity [7] and natural language inference [48] datasets. The teacher network only encodes English sentences s_i. The student model $\widehat{M}$ is then trained to produce for both s_j and t_j the same representation that M produces for s_j. We benchmark different DISTIL models in our CLIR experiments, with the student $\widehat{M}$ initialized with different multilingual transformers.

4 Experimental Setup

Evaluation Data. We follow the experimental setup of Litschko et al. [27], and compare the models from Sect. 3 on language pairs comprising five languages: English (EN), German (DE), Italian (IT), Finnish (FI) and Russian (RU). For document-level retrieval we run experiments for the following nine language pairs: EN-{FI, DE, IT, RU}, DE-{FI, IT, RU}, FI-{IT, RU}. We use the 2003 portion of the CLEF benchmark [4],[4] with 60 queries per language pair. The document collection sizes are 17K (RU), 55K (FI), 158K (IT), and 295K (DE). For sentence-level retrieval, also following [27], for each language pair we sample from Europarl [24] 1K source language sentences as queries and 100K target language sentences as the "document collection".[5]

Baseline Models. In order to establish whether multilingual encoders outperform CLWEs in a fair comparison, we compare their performance against the strongest CLWE-based CLIR model from the recent comparative study [27], dubbed Proc-B. Proc-B induces a bilingual CLWE space from pretrained monolingual FASTTEXT embeddings[6] using the linear projection computed as the solution of the Procrustes problem given the dictionary of word-translation pairs. Compared to simple Procrustes mapping, Proc-B iteratively (1) augments the word translation dictionary by finding mutual nearest neighbours and (2) induces a new projection matrix using the augmented dictionary. The final bilingual CLWE space is then plugged into the CLIR model from Sect. 3.1.

Our document-level retrieval SEMB models do not get to see the whole document but only the first 128 word-piece tokens. For a more direct comparison, we therefore additionally evaluate the Proc-B baseline (Proc-B$_{LEN}$) which is exposed to exactly the same amount of document text as the multilingual XLM encoder (i.e., the leading document text corresponding to first 128 word-piece tokens) Finally, we compare CLIR models based on multilingual Transformers

[4] http://catalog.elra.info/en-us/repository/browse/ELRA-E0008/.

[5] Russian is not included in Europarl and we therefore exclude it from sentence-level experiments. Further, since some multilingual encoders have not seen Finnish data in pretraining, we additionally report the results over a subset of language pairs that do not involve Finnish.

[6] https://fasttext.cc/docs/en/pretrained-vectors.html.

to a baseline relying on machine translation baseline (MT-IR). In MT-IR, 1) we translate the query to the document language using Google Translate and then 2) perform monolingual retrieval using a standard Query Likelihood Model [35] with Dirichlet smoothing [55].

Model Details. For all multilingual encoders we experiment with different input sequence lengths: 64, 128, 256 subword tokens. For AOC we collect (at most) $\tau = 60$ contexts for each vocabulary term: for a term not present at all in Wikipedia, we fall back to the ISO embedding of that term. We also investigate the impact of τ in Sect. 5.3. For purely self-supervised models (SEMB, ISO, AOC) we independently evaluate representations from different Transformer layers (cf. Sect. 5.3). For comparability, for ISO and AOC – methods that effectively induce static word embeddings using multilingual contextual encoders – we opt for exactly the same term vocabularies used by the Proc-B baseline, namely the top 100K most frequent terms from respective monolingual fastText vocabularies. We additionally experiment with three different instances of the DISTIL model: (i) $\text{DISTIL}_{\text{XLM-R}}$ initializes the student model with the pretrained XLM-R transformer [11]; $\text{DISTIL}_{\text{USE}}$ instantiates the student as the pretrained m-USE instance [51]; whereas $\text{DISTIL}_{\text{DistilmBERT}}$ distils the knowledge from the Sentence-BERT teacher into a multilingual version of DistilBERT [42], a 6-layer transformer pre-distilled from mBERT.[7] For SEMB models we scale embeddings of special tokens (sequence start and end tokens, e.g., `[CLS]` and `[SEP]` for mBERT) with the mean IDF value of input terms.

5 Results and Discussion

5.1 Document-Level Cross-Lingual Retrieval

We show the performance (MAP) of multilingual encoders on document-level CLIR tasks in Table 1. The first main finding is that none of the self-supervised models (mBERT and XLM in ISO, AOC, and SEMB variants) outperforms the CLWE baseline Proc-B. However, the full Proc-B baseline has, unlike mBERT and XLM variants, been exposed to the full content of the documents. A fairer comparison, against Proc-B_{LEN}, which has also been exposed only to the first 128 tokens, reveals that SEMB and AOC variants come reasonably close, albeit still do not outperform Proc-B_{LEN}. This suggests that the document-level retrieval could benefit from encoders able to encode longer portions of text, e.g., [3,54]. For document-level CLIR, however, these models would first have to be ported to multilingual setups. Scaling embeddings by their *idf* (Proc-B) effectively filters out high-frequent terms such as stopwords. We therefore experiment with explicit a priori stopword filtering in $\text{DISTIL}_{\text{DistilmBERT}}$, dubbed $\text{DISTIL}_{\text{FILTER}}$. Results show that performance deteriorates which indicates that stopwords provide important contextualization information. While SEMB and AOC variants exhibit similar performance, ISO variants perform much worse.

[7] Working with mBERT directly instead of its distilled version led to similar scores, while increasing running times.

Table 1. Document-level CLIR results (Mean Average Precision, MAP). **Bold**: best model for each language-pair. *: difference in performance w.r.t. Proc-B significant at $p = 0.05$, computed via paired two-tailed t-test with Bonferroni correction.

	EN-FI	EN-IT	EN-RU	EN-DE	DE-FI	DE-IT	DE-RU	FI-IT	FI-RU	AVG	w/o FI
Baselines											
MT-IR	.276	**.428**	.383	**.263**	**.332**	**.431**	.238	**.406**	.261	**.335**	**.349**
Proc-B	.258	.265	.166	.288	.294	.230	.155	.151	.136	.216	.227
Proc-B$_{\text{LEN}}$	.165	.232	.176	.194	.207	.186	.192	.126	.154	.181	.196
Models based on multilingual Transformers											
SEMB$_{\text{XLM}}$	.199*	.187*	.183	.126*	.156*	.166*	.228	.186*	.139	.174	.178
SEMB$_{\text{mBERT}}$	.145*	.146*	.167	.107*	.151*	.116*	.149*	.117	.128*	.136	.137
AOC$_{\text{XLM}}$	.168	.261	.208	.206*	.183	.190	.162	.123	.099	.178	.206
AOC$_{\text{mBERT}}$	.172*	.209*	.167	.193*	.131*	.143*	.143	.104	.132	.155	.171
ISO$_{\text{XLM}}$	.058*	.159*	.050*	.096*	.026*	.077*	.035*	.050*	.055*	.067	.083
ISO$_{\text{mBERT}}$	.075*	.209	.096*	.157*	.061*	.107*	.025*	.051*	.014*	.088	.119
Similarity-specialized sentence encoders (with parallel data supervision)											
DISTIL$_{\text{FILTER}}$	.291	.261	.278	.255	.272	.217	.237	.221	.270	.256	.250
DISTIL$_{\text{XLM-R}}$	.216	.190*	.179	.114*	.237	.181	.173	.166	.138	.177	.167
DISTIL$_{\text{USE}}$	.141*	.346*	.182	.258	.139*	.324*	.179	.104	.111	.198	.258
DISTIL$_{\text{DistilmBERT}}$	**.294**	.290*	**.313**	.247*	.300	.267*	**.284**	.221*	**.302***	.280	.280
LaBSE	.180*	.175*	.128	.059*	.178*	.160*	.113*	.126	.149	.141	.127
LASER	.142	.134*	.076	.046*	.163*	.140*	.065*	.144	.107	.113	.094
m-USE	.109*	.328*	.214	.230*	.107*	.294*	.204	.073	.090	.183	.254

The direct comparison between ISO and AOC demonstrates the importance of contextual information and seemingly limited usability of multilingual encoders as word encoders, if no context is available.

Similarity-specialized multilingual encoders, which rely on pretraining with parallel data, yield mixed results. Three models, DISTIL$_{\text{DistilmBERT}}$, DISTIL$_{\text{USE}}$ and m-USE, generally outperform the Proc-B baseline[8] LASER is the only encoder trained on parallel data that does not beat the Proc-B baseline. We believe this is because (a) LASER's recurrent encoder provides text embeddings of lower quality than Transformer-based encoders of m-USE and DISTIL variants and (b) it has not been subdued to any self-supervised pretraining like DISTIL models. Even the best-performing CLIR model based on a multilingual encoder (DISTIL$_{\text{DistilmBERT}}$) overall falls behind the MT-based baseline (MT-IR). However, the performance of MT-IR crucially depends on the quality of MT for the concrete language pair: for language pairs with weaker MT (e.g., FI-RU, EN-FI, FI-RU, DE-RU), DISTIL$_{\text{DistilmBERT}}$ can substantially outperform MT-IR (e.g., 9 MAP points for FI-RU and DE-RU); the gap in favor of MT-IR is, as expected, largest for most similar language pairs, for which also the most reliable MT systems exist (EN-IT, EN-DE). In other words, the feasibility and robustness of a strong MT-IR CLIR model seems to diminish with more distant

[8] As expected, m-USE and DISTIL$_{\text{USE}}$ perform poorly on language pairs involving Finnish, as they have not been trained on any Finnish data.

Table 2. Sentence-level CLIR results (MAP). **Bold**: best model for each language-pair. *: difference in performance with respect to Proc-B, significant at $p = 0.05$, computed via paired two-tailed t-test with Bonferroni correction.

	EN-FI	EN-IT	EN-DE	DE-FI	DE-IT	FI-IT	AVG	w/o FI
Baselines								
MT-IR	.659	.803	.725	.541	.694	.698	.687	.740
Proc-B	.143	.523	.415	.162	.342	.137	.287	.427
Models based on multilingual Transformers								
SEMB_{XLM}	.309*	.677*	.465	.391*	.495*	.346*	.447	.545
$\text{SEMB}_{\text{mBERT}}$	.199*	.570	.355	.231*	.481*	.353*	.365	.469
AOC_{XLM}	.099	.527	.274*	.102*	.282	.070*	.226	.361
$\text{AOC}_{\text{mBERT}}$	.095*	.433*	.274*	.088*	.230*	.059*	.197	.312
ISO_{XLM}	.016*	.178*	.053*	.006*	.017*	.002*	.045	.082
$\text{ISO}_{\text{mBERT}}$	.010*	.141*	.087*	.005*	.017*	.000*	.043	.082
Similarity-specialized sentence encoders (with parallel data supervision)								
$\text{DISTIL}_{\text{XLM-R}}$	.935*	.944*	.943*	.911*	.919*	.914*	.928	.935
$\text{DISTIL}_{\text{USE}}$	.084*	.960*	.952*	.137	.920*	.072*	.521	.944
$\text{DISTIL}_{\text{DistilmBERT}}$	.847*	.901*	.901*	.811*	.842*	.793*	.849	.882
LaBSE	.971*	.972*	.964*	.948*	.954*	.951*	.960	.963
LASER	**.974***	**.976***	**.969***	**.967***	**.965***	**.961***	**.969**	**.970**
m-USE	.079*	.951*	.929*	.086*	.886*	.039*	.495	.922

language pairs and lower-resource language pairs. We plan to investigate this conjecture in more detail in future work.

The variation in results with similarity-specialized sentence encoders indicates that: (a) despite their seemingly similar high-level architectures typically based on dual-encoder networks [8], it is important to carefully choose a sentence encoder in document-level retrieval, and (b) there is an inherent mismatch between the granularity of information encoded by the current state-of-the-art text representation models and the document-level CLIR task.

5.2 Sentence-Level Cross-Lingual Retrieval

We show the sentence-level CLIR performance in Table 2. Unlike in the document-level CLIR task, self-supervised SEMB variants here manage to outperform Proc-B. The better relative SEMB performance than in document-level retrieval is somewhat expected: sentences are much shorter than documents (i.e., typically shorter than the maximal sequence length of 128 word pieces). All purely self-supervised mBERT and XLM variants, however, perform worse than the translation-based baseline.

Multilingual encoders specialized with parallel data excel in sentence-level CLIR, all of them substantially outperforming the competitive MT-IR baseline.

This however, does not come as much of a surprise, since these models (a) have been trained using parallel data, and (b) have been optimized exactly on the sentence similarity task. In other words, in the context of the cross-lingual sentence-level task, these models are effectively supervised models. The effect of supervision is most strongly pronounced for LASER, which was, by being also trained on parallel data from Europarl, effectively subdued to in-domain training. We note that at the same time LASER was the weakest model from this group on average in the document-level CLIR task.

5.3 Further Analysis

We further investigate three aspects that may impact CLIR performance of multilingual encoders: (1) layer(s) from which we take vector representations, (2) number of contexts used in AOC variants, and (3) sequence length in document-level CLIR.

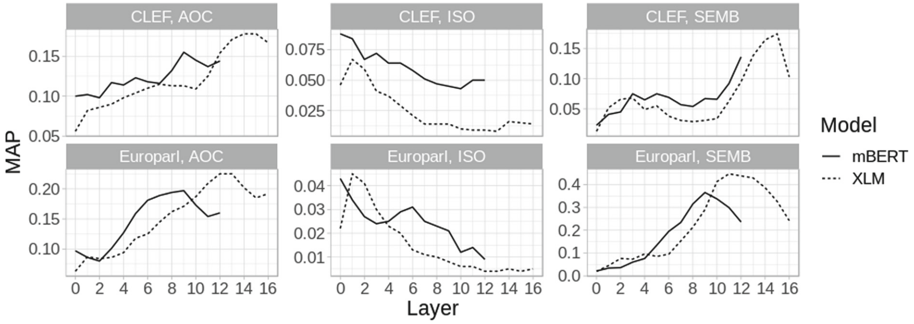


Fig. 1. CLIR performance of mBERT and XLM as a function of the Transformer layer from which we obtain the representations. Results (averaged over all language pairs) shown for all three encoding strategies (SEMB, AOC, ISO).

Layer Selection. All multilingual encoders have multiple layers and one may select (sub)word representations for CLIR at the output of any of them. Figure 1 shows the impact of taking subword representations after each layer for self-supervised mBERT and XLM variants. We find that the optimal layer differs across the encoding strategies (AOC, ISO, and SEMB) and tasks (document-level vs. sentence-level CLIR). ISO, where we feed the terms into encoders without any context, seems to do best if we take the representations from lowest layers. This makes intuitive sense, as the parameters of higher Transformer layers encode compositional rather than lexical semantics [17,40]. For AOC and SEMB, where both models obtain representations by contextualizing (sub)words in a sentence, we get the best performance for higher layers – the optimal layers for document-level retrieval (L9/L12 for mBERT, and L15 for XLM) seem to be higher than for sentence-level retrieval (L9 for mBERT and L12/L11 for XLM).

Number of Contexts in AOC. We construct AOC term embeddings by averaging contextualized representations of the same term obtained from different Wikipedia contexts. This raises an obvious question of a sufficient number of contexts needed for a reliable (static) term embedding. Figure 2 shows the AOC results depending on the number of contexts used to induce the term vectors (cf. τ in Sect. 3). The AOC performance seems to plateau rather early – at around 30 and 40 contexts for mBERT and XLM, respectively. Encoding more than 60 contexts (as we do in our main experiments) would therefore bring only negligible performance gains.

Input Sequence Length. Multilingual encoders have a limited input length and they, unlike CLIR models operating on static embeddings (Proc-B, as well as our AOC and ISO variants), effectively truncate long documents. In our main experiments we truncated the documents to first 128 word pieces. Now we quantify (Table 3) if and to which extent this has a detrimental effect on document-level CLIR performance. Somewhat counterintuitively, encoding a longer chunk of documents (256 word pieces) yields a minor performance deterioration (compared to the length of 128) for *all* multilingual encoders. We suspect that this is a combination of two effects: (1) it is more difficult to semantically accurately encode a longer portion of text, leading to semantically less precise embeddings of 256-token sequences; and (2) for documents in which the query-relevant content is not within the first 128 tokens, that content might often also appear beyond the first 256 tokens, rendering the increase in input length inconsequential to the recognition of such documents as relevant.

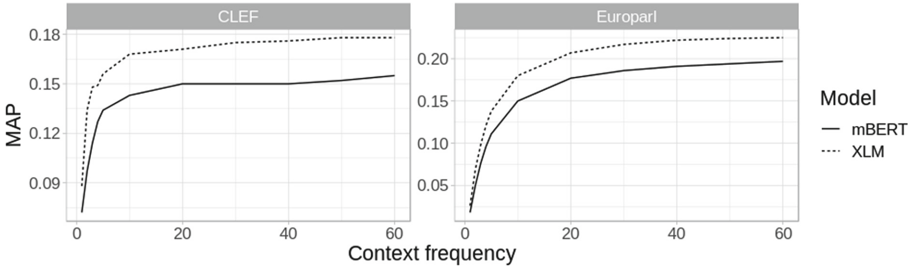


Fig. 2. CLIR performance of AOC variants (mBERT and XLM) w.r.t. the number of contexts used to obtain the term embeddings.

Table 3. Document CLIR results w.r.t. the input text length. Scores averaged over all language pairs not involving Finnish.

Length	SEMB$_{\text{mBERT}}$	SEMB$_{\text{XLM}}$	DIST$_{\text{use}}$	DIST$_{\text{XLM-R}}$	DIST$_{\text{DmBERT}}$	mUSE	LaBSE	LASER
64	.104	.128	.235	.167	.237	.254	.127	.089
128	.137	.178	.258	.162	.280	.247	.125	.068
256	.117	.158	.230	.146	.250	.197	.096	.027

6 Conclusion

Pretrained multilingual encoders have been shown to be widely useful in natural language understanding (NLU) tasks, when fine-tuned in supervised settings on some task-specific data; their utility as general-purpose text encoders in unsupervised settings, such as the ad-hoc cross-lingual IR, has been less investigated. In this work, we systematically validated the suitability of a wide spectrum of cutting-edge multilingual encoders for document- and sentence-level CLIR across several language pairs. Our study invluded self-supervised multilingual encoders, mBERT and XLM, as well as the those that have been specialized for semantic text matching on semantic similarity datasets and parallel data. Opposing the findings from supervised NLU, we demonstrated that self-supervised multilingual encoders (mBERT and XLM), without exposure to task supervision, typically fail to outperform CLIR models based on cross-lingual word embeddings (CLWEs). Semantically-specialized multilingual sentence encoders, on the other hand, do outperform CLWEs, but the gains are pronounced only in the sentence retrieval task. While state-of-the-art multilingual text encoders excel in so many seemingly more complex language understanding tasks, our work renders ad-hoc CLIR in general and document-level CLIR in particular a serious challenge for these models. We make our code and resources available at https://github.com/rlitschk/EncoderCLIR.

Acknowledgments. The work of Ivan Vulić is supported by the ERC Consolidator Grant LEXICAL: Lexical Acquisition Across Languages (no 648909). Robert Litschko and Goran Glavaš are supported by the Baden Württemberg Stiftung (Eliteprogramm, AGREE grant).

References

1. Artetxe, M., Labaka, G., Agirre, E.: A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In: Proceedings of ACL, pp. 789–798 (2018)
2. Artetxe, M., Schwenk, H.: Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. Trans. Assoc. Comput. Linguist. **7**, 597–610 (2019)
3. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: the long-document transformer. arXiv preprint arXiv:2004.05150 (2020)
4. Braschler, M.: CLEF 2003 – Overview of Results. In: Peters, C., Gonzalo, J., Braschler, M., Kluck, M. (eds.) CLEF 2003. LNCS, vol. 3237, pp. 44–63. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30222-3_5
5. Brown, T.B., et al.: Language models are few-shot learners. In: Proceedings of NeurIPS (2020)
6. Cao, S., Kitaev, N., Klein, D.: Multilingual alignment of contextual word representations. In: Proceedings of ICLR (2020)
7. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: SemEval-2017 task 1: semantic textual similarity multilingual and crosslingual focused evaluation. In: Proceedings of SemEval, pp. 1–14 (2017)

8. Cer, D., et al.: Universal sentence encoder for English. In: Proceedings of EMNLP, pp. 169–174 (2018)
9. Chidambaram, M., et al.: Learning cross-lingual sentence representations via a multi-task dual-encoder model. In: Proceedings of the ACL Workshop on Representation Learning for NLP, pp. 250–259 (2019)
10. Clark, K., Luong, M., Le, Q.V., Manning, C.D.: ELECTRA: pre-training text encoders as discriminators rather than generators. In: Proceedings of ICLR (2020)
11. Conneau, A., et al.: Unsupervised cross-lingual representation learning at scale. In: Proceedings of ACL, pp. 8440–8451 (2020)
12. Conneau, A., Kiela, D.: SentEval: an evaluation toolkit for universal sentence representations. In: Proceedings of LREC, pp. 1699–1704 (2018)
13. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of EMNLP, pp. 670–680 (2017)
14. Conneau, A., Lample, G.: Cross-lingual language model pretraining. In: Proceedings of NeurIPS, pp. 7059–7069 (2019)
15. Conneau, A., et al.: XNLI: evaluating cross-lingual sentence representations. In: Proceedings of EMNLP, pp. 2475–2485 (2018)
16. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL, pp. 4171–4186 (2019)
17. Ethayarajh, K.: How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In: Proceedings of EMNLP-IJCNLP, pp. 55–65 (2019)
18. Feng, F., Yang, Y., Cer, D., Arivazhagan, N., Wang, W.: Language-agnostic BERT sentence embedding. arXiv preprint arXiv:2007.01852 (2020)
19. Glavaš, G., Litschko, R., Ruder, S., Vulić, I.: How to (properly) evaluate cross-lingual word embeddings: on strong baselines, comparative analyses, and some misconceptions. In: Proceedings of ACL, pp. 710–721 (2019)
20. Guo, M., et al.: Effective parallel corpus mining using bilingual sentence embeddings. In: Proceedings of WMT, pp. 165–176 (2018)
21. Hoogeveen, D., Verspoor, K.M., Baldwin, T.: CQADupStack: a benchmark data set for community question-answering research. In: Proceedings of ADCS, pp. 3:1–3:8 (2015)
22. Jiang, Z., El-Jaroudi, A., Hartmann, W., Karakos, D., Zhao, L.: Cross-lingual information retrieval with BERT. In: Proceedings of LREC, p. 26 (2020)
23. Karthikeyan, K., Wang, Z., Mayhew, S., Roth, D.: Cross-lingual ability of multilingual BERT: an empirical study. In: Proceedings of ICLR (2020)
24. Koehn, P.: Europarl: a parallel corpus for statistical machine translation. In: Proceedings of the 10th Machine Translation Summit (MT SUMMIT), pp. 79–86 (2005)
25. Lei, T., et al.: Semi-supervised question retrieval with gated convolutions. In: Proceedings of NAACL, pp. 1279–1289 (2016)
26. Liang, Y., et al.: XGLUE: a new benchmark dataset for cross-lingual pre-training, understanding and generation. In: Proceedings of EMNLP (2020)
27. Litschko, R., Glavaš, G., Vulić, I., Dietz, L.: Evaluating resource-lean cross-lingual embedding models in unsupervised retrieval. In: Proceedings of SIGIR, pp. 1109–1112 (2019)
28. Liu, Q., Kusner, M.J., Blunsom, P.: A survey on contextual embeddings. arXiv preprint arXiv:2003.07278 (2020)

29. Liu, Q., McCarthy, D., Vulić, I., Korhonen, A.: Investigating cross-lingual alignment methods for contextualized embeddings with token-level evaluation. In: Proceedings of CoNLL, pp. 33–43 (2019)
30. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
31. MacAvaney, S., Soldaini, L., Goharian, N.: Teaching a new dog old tricks: resurrecting multilingual retrieval using zero-shot learning. In: Proceedings of ECIR, pp. 246–254 (2020)
32. MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: Cedr: contextualized embeddings for document ranking. In: Proceedings of SIGIR, pp. 1101–1104 (2019)
33. Nogueira, R., Yang, W., Cho, K., Lin, J.: Multi-stage document ranking with BERT. arXiv preprint arXiv:1910.14424 (2019)
34. Pires, T., Schlinger, E., Garrette, D.: How multilingual is multilingual BERT? In: Proceedings of ACL, pp. 4996–5001 (2019)
35. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: Proceedings of SIGIR, pp. 275–281 (1998)
36. Ponti, E.M., Glavaš, G., Majewska, O., Liu, Q., Vulić, I., Korhonen, A.: XCOPA: a multilingual dataset for causal commonsense reasoning. In: Proceedings of EMNLP (2020)
37. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
38. Reimers, N., Gurevych, I.: Sentence-BERT: sentence embeddings using siamese BERT-networks. In: Proceedings of EMNLP, pp. 3973–3983 (2019)
39. Reimers, N., Gurevych, I.: Making monolingual sentence embeddings multilingual using knowledge distillation. In: Proceedings of EMNLP (2020)
40. Rogers, A., Kovaleva, O., Rumshisky, A.: A primer in BERTology: what we know about how BERT works. Trans. Assoc. Comput. Linguist. **8**, 842–866 (2020)
41. Ruder, S., Vulić, I., Søgaard, A.: A survey of cross-lingual word embedding models. J. Artif. Intell. Res. **65**, 569–631 (2019)
42. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
43. Smith, S.L., Turban, D.H., Hamblin, S., Hammerla, N.Y.: Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In: Proceedings of ICLR (2017)
44. Vaswani, A., et al.: Attention is all you need. In: Proceedings of NeurIPS, pp. 5998–6008 (2017)
45. Vulić, I., Glavas, G., Reichart, R., Korhonen, A.: Do we really need fully unsupervised cross-lingual embeddings? In: Proceedings of EMNLP, pp. 4406–4417 (2019)
46. Vulić, I., Moens, M.F.: Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In: Proceedings of SIGIR, pp. 363–372 (2015)
47. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: a multi-task benchmark and analysis platform for natural language understanding. In: Proceedings of ICLR (2019)
48. Williams, A., Nangia, N., Bowman, S.: A broad-coverage challenge corpus for sentence understanding through inference. In: Proceedings of NAACL, pp. 1112–1122 (2018)
49. Wu, S., Dredze, M.: Beto, bentz, becas: the surprising cross-lingual effectiveness of BERT. In: Proceedings of EMNLP, pp. 833–844 (2019)

50. Yang, Y., et al.: Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. In: Proceedings of AAAI, pp. 5370–5378 (2019)
51. Yang, Y., et al.: Multilingual universal sentence encoder for semantic retrieval. In: Proceedings of ACL: System Demonstrations, pp. 87–94 (2020)
52. Yang, Y., et al.: Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. In: Proceedings of IJCAI, pp. 5370–5378 (2019)
53. Yu, P., Allan, J.: A study of neural matching models for cross-lingual IR. In: Proceedings of SIGIR, pp. 1637–1640 (2020)
54. Zaheer, M., et al.: Big Bird: transformers for longer sequences. arXiv preprint arXiv:2007.14062 (2020)
55. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst. (TOIS) **22**(2), 179–214 (2004)
56. Zhao, W., Eger, S., Bjerva, J., Augenstein, I.: Inducing language-agnostic multilingual representations. arXiv preprint arXiv:2008.09112 (2020)
57. Zhao, W., Glavaš, G., Peyrard, M., Gao, Y., West, R., Eger, S.: On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In: Proceedings of ACL, pp. 1656–1671 (2020)
58. Ziemski, M., Junczys-Dowmunt, M., Pouliquen, B.: The United Nations parallel corpus v1.0. In: Proceedings of LREC, pp. 3530–3534 (2016)
59. Zweigenbaum, P., Sharoff, S., Rapp, R.: Overview of the third BUCC shared task: spotting parallel sentences in comparable corpora. In: Proceedings of LREC (2018)

Diagnosis Ranking with Knowledge Graph Convolutional Networks

Bing Liu(✉), Guido Zuccon, Wen Hua, and Weitong Chen

The University of Queensland, St. Lucia, Brisbane, Australia
{bing.liu,g.zuccon,w.hua,w.chen9}@uq.edu.au

Abstract. The automatic diagnosis of a medical condition provided the symptoms exhibited by a patient is at the basis of systems for clinical decision support, as well as for applications such as symptom checkers. Existing methods have not fully exploited medical knowledge: this likely hinders their effectiveness. In this work, we propose a knowledge-aware diagnosis ranking framework based on medical knowledge graph (KG) and graph convolutional neural network (GCN). The medical KG is used to model hierarchy and causality relationships between diseases and symptoms. We have evaluated our proposed method using realistic patient cases. The empirical results show that our knowledge-aware diagnosis ranking framework can improve the effectiveness of medical diagnosis.

Keywords: Knowledge graph · Graph Convolutional Networks · Diagnosis ranking

1 Introduction

A common task in medical practice is to identify a diagnosis for a patient presenting with one or more symptoms. To do so, clinicians rely on their extensive medical knowledge about the relationships between symptoms and the possible diagnoses, and weight up symptoms (and laboratory findings) to determine the most likely diagnosis, often through a process called differential diagnosis [25]. Computer assisted or automated methods for medical diagnosis have emerged where computer algorithms are used to mine a large amount of medical data (from medical literature or electronic health records) to provide clinicians with recommendations regarding a patient case [15]. Current methods are limited in that they do not sufficiently exploit medical knowledge [5,6]. In addition, most methods formulate the problem as a classification task and assume diagnosis classes are independent: this is a problem as medical conditions are instead related (e.g., hierarchy of conditions, causality between conditions – see Sect. 2 for details).

We posit that the exploitation of medical knowledge, in particular as encoded in medical KGs, within an end-to-end deep learning architecture for diagnosis identification may improve the effectiveness of current automated medical

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 359–374, 2021.
https://doi.org/10.1007/978-3-030-72113-8_24

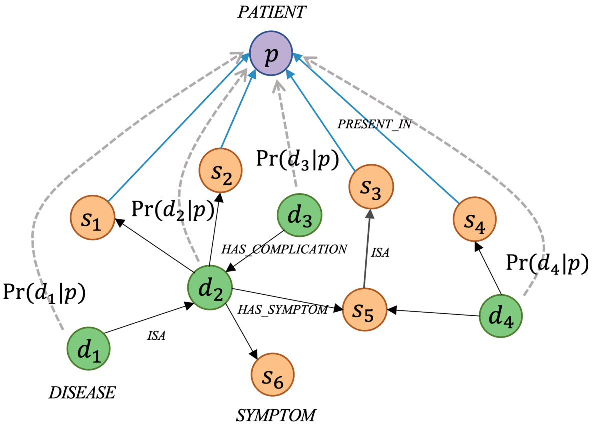


Fig. 1. Exemplified medical KG. Concepts (nodes) belong to different types (e.g., symptoms (orange), diseases (green)) and are linked by various relationships, e.g., ISA, $HAS\_SYMPTOM$. (Color figure online)

diagnosis systems. To this end, we propose a Knowledge Graph Convolutional Network (KGCN) method for ranking diagnosis (Sect. 3), that exploits medical KGs to enable capturing insightful diagnosis patterns. In our method, a patient's symptoms are identified within the KG and used to derive likely diagnoses (diseases) for the patient based on the representations of medical concepts and their relationships encoded in the KGs. We use the concept of message diffusion in Graph Convolutional Networks (GCN) [9,17] to model the relationships between symptoms and diseases encoded in the KG. Specifically, we inject a special node - patient node - to the medical KG and connect its symptoms to it (see Fig. 1). We refer to the formed graph as *diagnosis graph* and each node in this graph has an initial representation. We then employ stacked GCN layers to the diagnosis graph to learn, for each node, a comprehensive representation. Through the message-passing mechanism of GCN, nodes share their information with their neighbours and meanwhile aggregate the received information from their neighbours. By stacking l GCN layers, the nodes can receive messages from their l-hop neighbours. This allows to use different types of relations and multi-hop contexts of nodes. We experiment with different fusion functions to study the most effective way of aggregating context information within a node. After obtaining comprehensive representations of disease concepts and patients, we predict the likelihood of a disease node to be connected to the patient node (link prediction) with a match model. Finally, we use this inferred probability to rank diagnoses for a given patient case.

We have evaluated the proposed method on a dataset of realistic patient vignettes redacted by medical experts (Sects. 4 and 5). Results show that our KGCN provides better diagnosis predictions than existing methods. We further tease out the impact of data sparsity, different medical relations, fusion functions, number of GCN layers, on the effectiveness of KGCN.

2 Related Work

Automatic medical diagnosis aims to assist clinicians with diagnosing patients by using computer algorithms to identify the most probable diagnoses for a patient, given their case description (disease history, symptoms, signs) [15].

Many Machine Learning algorithms have been explored to learn diagnosis patterns automatically from existing medical records to support this task [1,16,24,27], but often the learned models achieved limited effectiveness. This has been because of insufficient data being available and the fact that relationships between medical concepts not being modelled and exploited by these methods.

To improve effectiveness, recent methods have attempted to learn distributed representations of medical concepts, e.g., from ontologies or electronic health records [5,6], and use them to enhance predictive models. Other work has introduced prior medical knowledge in the form of knowledge graph [28] or rules [18] into models to improve the effectiveness of disease prediction. Though promising, also this line of work has limitations.

A first limitation is that existing work formulates medical diagnosis as a (multi-class) classification problem. The underlying assumption in doing so is that the classes (diseases) are assumed independent: this assumption is not true as often diseases are related e.g., due to presenting the same symptoms, being a more specific instance of a general condition, or being common co-morbidities. Adequately modelling this relatedness, instead, may likely allow for better discrimination among diagnoses and thus better diagnosis effectiveness. In this work we take a different stand by formulating medical diagnosis as a matching problem, where patient's descriptions (symptoms) and diagnoses are represented within a knowledge graph using rich features and are matched to produce a ranking of possible diagnoses, starting from the most likely.

Another limitation of previous work is that medical knowledge has often not been fully exploited. Medical knowledge has been extensively modelled by manually curated domain-specific resources such as medical ontologies and terminology, e.g., SNOMED CT [23], MedRA[1], UMLS [3], and automatically mined medical Knowledge Graphs (KGs), e.g., KnowLife [7], Rotmensch et al.'s [19], HighLife [8], etc. In Fig. 1 we provide a schematic example of a Knowledge Graph in this context. While previous work has used such medical knowledge for diagnosis identification, this came with limitations. Some works [5,6] mainly focused on hierarchy information (i.e., *ISA*) and ignored other important relationships, such as *HAS_SYMPTOM* between disease and symptom, *HAS_COMPLICATION* between diseases, etc. Some other works only considered to add direct contexts in KGs to the model but neglected multi-hop contexts. However, multi-hop contexts are common in medicine, often being used for modelling properties such as the transitivity of hierarchy or chains of relationships for causality. Fully relying on the extensive medical knowledge captured in these domain-specific resources, instead, may likely lead to better diagnosis effectiveness.

[1] https://www.meddra.org/.

As mentioned above, our solution relies on a medical KG to estimate the match between a set of symptoms and the likely diagnosis. Three main avenues have been explored in the literature when relying on KGs for matching:

1. use knowledge graph embedding (KGE) algorithms to learn the vector representations of entities and relationships in a KG, and then use these within the downstream applications related to matching [10]. In this way, KGs are used independently of the end-task and thus their use is rather flexible. However, there is a mismatch between the goal of the KGE construction process, which is to encode the semantic relatedness among entities in the KG, and the end-task goal for which the learned embeddings are used, i.e. matching
2. identify various connection patterns among entities in a KG to exploit as additional matching signals. This provides intuitive methods that heavily rely on manually designed meta-graphs: these however are often hard to tune in practice.
3. integrate matching models and KGs in a hybrid graph and inject the structure information of KGs into the matching problem to form an end-to-end task. This solution can avoid the shortcomings of the first two alternatives described above.

The proposed KGCN follows the third solution, integrating the matching model and the KG in a hybrid graph to be used within an end-to-end pipeline.

Our proposed method relies on Graph Convolution Networks (GCN) [4,14,20], which generalized convolutional neural networks to non-Euclidean spaces such as a graph. The key idea of GCNs is to generate node embeddings through message passing or information diffusion processes executed on the graph [9].

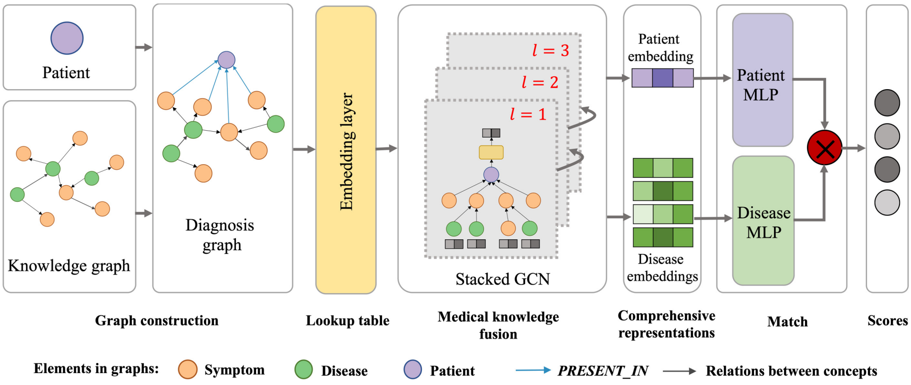


Fig. 2. Overview of our method. The framework consists of several stages: 1) construct the diagnosis graph by linking the patient to the medical KG, 2) fuse medical knowledge using stacked GCN layers to obtain a comprehensive representation of each node, 3) transform the new representations of patient and disease nodes into the same latent space using MLP layers and obtain similarity scores using the inner product.

3 Knowledge Graph Convolution Networks for Diagnosis Ranking

Figure 2 provides an overview of our model. In particular, we add a special node to an existing medical KG to form a diagnosis graph, in which the patient node is linked to nodes representing the symptoms exhibited by the patient (as described in the patient case vignette). GCN is then adopted to learn comprehensive representations of the patient and medical concepts. Finally, we predict the likelihood that a disease node may be linked to the patient node and rank diagnoses based on the probability distribution with respect to the patient case. We elaborate on each component of the proposed model in the following.

3.1 Problem Formulation

Medical diagnosis is the process that attempts to determine the disease $d \in \mathbb{D}$ ($\mathbb{D}$ being the set of possible diseases) affecting a patient p who exhibits a set of symptoms $p = \{s_1, s_2, ..., s_n\}$, $s_i \in \mathbb{S}$ ($\mathbb{S}$ being the set of possible symptoms). We refer to the pair (p, d) as a *case*. To assist the diagnosis process, we exploit a medical KG $\mathscr{K} = \{(h, r, t)|h, t \in \mathbb{D} \cup \mathbb{S}, r \in \mathbb{R}\}$, where $\mathbb{R}$ is the set of relations between medical concepts. The KG is essentially a directed heterogeneous graph.

In the learning process, some cases $\mathrm{Y} = \{(p_i, d_i)\}, 0 \leq i \leq |Y|$ are provided for training the model, with the goal to derive a prediction function $y_{p,d} = \mathscr{F}(p, d \mid \Theta, \mathscr{K}, \mathrm{Y})$. Here, $y_{p,d}$ represents the probability that the disorder d is the true diagnosis for patient p, and Θ denotes the parameters of the prediction function $\mathscr{F}$. In the diagnosis process, given a patient with symptoms, the model uses $\mathscr{F}$ to obtain his matching score with each disease $d \in \mathbb{D}$ and outputs a ranked disease list.

3.2 Construction of the Diagnosis Graph

We construct the diagnosis graph $\mathscr{G}$ by injecting the patient node p to an existing medical KG $\mathscr{K}$. In this paper, we use a subset of SemmedDB [13] as the KG. SemmedDB contains a large amount of predications extracted from biomedical texts (scientific articles); our subset only contains the triples whose head and tail entities are symptom or disease concepts and the relation is of type *isa* or *causes*[2]. To construct the diagnosis graph, we create a special patient node, identify the symptoms of the patient in the KG, and link these symptom nodes to the patient node with edges of type *present_in* [3]. The obtained diagnosis graph is denoted as $\mathscr{G} = \{(u, e_{uv}, v)|u, v \in \mathbb{D} \cup \mathbb{S} \cup \{p\}, e_{uv} \in \{\textit{present\_in}, \textit{causes}, \textit{isa}\}\}$.

[2] Note that the relation *causes* in SemmedDB is rather coarse and encompasses relations that would normally be treated as separate in other medical KGs, including relations such as *has_complication*, *has_symptom*.

[3] We link a patient with the KG through the symptoms' Concept Unique Identifiers (CUIs). Medical concept recognition tools like QuickUMLS [22] and MetaMap [2] can recognize and map terms in patients' records to CUIs; each entity in the medical KG is represented by a CUI.

3.3 Embedding Layer

The embedding layer is used to assign an initial vector representation to each node in the diagnosis graph with a look-up table operation. Every concept node $c \in \mathbb{D} \cup \mathbb{S}$ is assigned a corresponding embedding $\boldsymbol{h}_c \in R^{N^0}$ while different patients share a single initial representation $\boldsymbol{h}_p \in R^{N^0}$. The embedding matrix is:

$$\boldsymbol{E}^{(1+|\mathbb{D}|+|\mathbb{S}|)\times N_0} = [\underbrace{\boldsymbol{h}_p}_{\text{patient}}, \underbrace{\boldsymbol{h}_{d_1}, ..., \boldsymbol{h}_{d_{|\mathbb{D}|}}}_{\text{disease}}, \underbrace{\boldsymbol{h}_{s_1}, ..., \boldsymbol{h}_{s_{|\mathbb{S}|}}}_{\text{symptom}}]. \tag{1}$$

These embeddings are initialized randomly and optimized in an end-to-end fashion.

3.4 Medical Knowledge Fusion Layer

The medical knowledge fusion layer is designed based on GCN, which employs message-passing architecture to capture the relatedness between medical concept nodes. In this process, the patient node also obtains its representation by fusing the symptoms and the potential causes of those symptoms. In the following, we first illustrate the first-order knowledge fusion and then generalize to high-order knowledge fusion.

First-Order Medical Knowledge Fusion. Within a single GCN layer, the message-passing process has two stages: (1) each node constructs messages and sends them to its neighbours through the outbound edges. The content of each message depends on the information contained in the source node, the type of edge, the information contained in the destination node. (2) each node aggregates the received messages from all inbound edges and fuses them with the information it contains.

Message Construction. The message sent from node u to v is represented by $\boldsymbol{m}_{u\to v} = f^{conc}(\boldsymbol{h}_u, r_{uv}, \alpha_{uv})$, where r_{uv} is the type of edge e_{uv}, α_{uv} is the decay factor of passing a message on edge e_{uv}, and $f^{conc}(\cdot)$ is the message construction function which takes the representation of node u, the edge type r_{uv} and the decay factor α_{uv} as input. In this work, we implement $f^{conc}(\cdot)$ as:

$$\boldsymbol{m}_{u\to v} = \alpha_{uv}(\boldsymbol{W}_{r_{uv}}\boldsymbol{h}_u + \boldsymbol{b}_{r_{uv}}), \tag{2}$$

where $\boldsymbol{W}_{r_{uv}} \in \boldsymbol{R}^{N^0\times N^1}$ and $\boldsymbol{b}_{r_{uv}} \in \boldsymbol{R}^{N^1}$ are trainable parameters to distill useful information for propagation.

Message Aggregation. We aggregate the received messages at node v by summing them as $\boldsymbol{a}_v = \sum_{u'\in\mathcal{N}_v} \boldsymbol{m}_{u'\to v}$, where $\mathcal{N}_v$ is the set of neighbours. Then, we fuse the aggregated context $\boldsymbol{a}_v$ with the node $\boldsymbol{h}_v$ itself as $\boldsymbol{h}_v^{(1)} = f^{fuse}(\boldsymbol{h}_v, \boldsymbol{a}_v)$, where $f^{fuse}(\cdot)$ is the fusion function. In this work, we exploit *GRU* as the fusion function as done by Li et al. [17]:

$$\boldsymbol{h}_v^{(1)} = GRU(\boldsymbol{h}_v, \boldsymbol{a}_v). \tag{3}$$

Comparison of Context Fusion Methods. The fusion function is a key component of our method since it determines if the context information can be effectively introduced. Intuitively, a node eagerly seeks to incorporate context when its representation is not informative enough, and its context can provide beneficial information. The way in which the context is to be fused with the node should depend on the representation of the node itself, the messages received from the context, and their interaction. In our method, we use GRU as the fusion function because its model structure can support this intuition. As comparison methods, we also implemented two alternative fusion functions, which are comparatively simple even though widely used in other tasks – these are described next.

SumFus takes the summation of two context vectors, followed by a non-linear transformation: $\boldsymbol{h}_v^{(1)} = \sigma(\boldsymbol{W}^{sg}(\boldsymbol{a}_v + \boldsymbol{h}_v) + \boldsymbol{b}^{sg})$, where $\boldsymbol{W}^{sg}$ and $\boldsymbol{b}^{sg}$ are the parameters, σ is the activation function.

ConcatFus concatenates two context vectors first before non-linear activation $\boldsymbol{h}_v^{(1)} = \sigma(\boldsymbol{W}^{cg}(\boldsymbol{a}_v \oplus \boldsymbol{h}_v) + \boldsymbol{b}^{cg})$, where $\oplus$ is the concatenation operation, $\boldsymbol{W}^{cg}$ and $\boldsymbol{b}^{cg}$ are the parameters, σ is the activation function.

High-Order Medical Knowledge Fusion. First-order context aggregation is primary for our medical diagnosis model since only symptom concepts are connected to the patients. To make the patient aware of the potential causes of the symptoms he shows, we need to do high-order context aggregation. By stacking l context aggregation layers, one node in the graph can receive messages propagated from l-hop neighbours. Formally, we repeat the context aggregation process by applying graph convolution operation on the graph and use the context vectors obtained from $(l-1)$-th GCN layer as the node representations, as in equation

$$\boldsymbol{m}_{u \rightarrow v}^{(l)} = \alpha_{uv}(\boldsymbol{W}_{r_{uv}}^{(l)} \boldsymbol{h}_u^{(l-1)} + \boldsymbol{b}_{r_{uv}}^{(l)}) \ . \tag{4}$$

Then, the new context representation of node v is obtained by aggregating the received messages from its neighbours $u' \in \mathcal{N}_v$ and fusing it with $\boldsymbol{h}_v^{(l-1)}$:

$$\boldsymbol{a}_v^{(l)} = \sum_{u' \in \mathcal{N}_v} \boldsymbol{m}_{u' \rightarrow v}^{(l)} \ , \quad \boldsymbol{h}_v^{(l)} = GRU(\boldsymbol{h}_v^{l-1}, \boldsymbol{a}_v^{(l)}) \ . \tag{5}$$

Here, $\boldsymbol{W}^l \in \boldsymbol{R}^{N^{l-1} \times N^l}$, $\boldsymbol{b}^l \in \boldsymbol{R}^{N^l}$ are trainable parameters in the l-th GCN layer.

3.5 Feature Transformation and Matching

After aggregating the medical knowledge with L GCN layers, each node obtained a comprehensive representation, which entails its original representation as well as the aggregated context information at each GCN layer. At the matching stage, we transform the patient node and disease nodes using MLP layers separately to get their final representation in the same latent space as $\boldsymbol{h}_p^o = MLP^p(\boldsymbol{h}_p^{(L)})$, $\boldsymbol{h}_d^o = MLP^d(\boldsymbol{h}_d^{(L)})$. Both of the MLPs have hyper-parameters: the number of hidden layers and the unit number of each hidden layer. After getting the final representations of the patient and each disease concept, we conduct inner product to calculate their similarity score as $y_{d_i,p} = \boldsymbol{h}_p^{o\top}\boldsymbol{h}_d^o$. We can further apply the softmax to these similarity scores to get the probability $\Pr(d_i|p)$ that a certain disease d_i is the true diagnosis of the patient p.

3.6 Ranking Diagnosis

We can rank the diseases $d \in \mathbb{D}$ according to their matching scores with a certain patient and then return a ranked list of diseases. It should be noticed that the patient nodes only have inbound edges and thus have no effect on the contextual representations of medical concepts. Therefore, the contextual representations $\boldsymbol{h}_c^{(l)}, c \in \mathbb{D} \cup \mathbb{S}, 0 \leq l \leq L$ of medical concepts only have to be calculated once and then put in cache for subsequent usage.

3.7 Training Model

To learn the model parameters, we choose Ranking Cross-Entropy, which has been widely used in matching models, as the loss function. Specifically, for a given patient $p_i = \{s_j\}$ and his ground truth diagnosis d_i^T, we sample N diseases $\{d_{i,k}^F\}_{1 \leq k \leq N}$ randomly from the disease set $\mathbb{D} \setminus \{d_i\}$ as negative diagnoses. Then, we calculate their matching scores $y_{p_i d_i^T}$ and $\{y_{p_i d_{i,k}^F}\}_{0 \leq k \leq N}$. Afterwards, we apply softmax function on those scores and get their normalized probabilities

$$\begin{aligned}&[\Pr(d_i^T|p_i), \Pr(d_{i,1}^F|p_i), ..., \Pr(d_{i,N}^F|p_i)]\\&= \mathrm{softmax}(y_{p_i,d_i^T}, y_{p_i,d_{i,1}^F}, ..., y_{p_i,d_{i,N}^F}).\end{aligned} \tag{6}$$

The cross entropy loss of training instance (p_i, d_i^T) is formulated as $loss_{p_i} = -\log \Pr(d_i^T|p_i)$. For a batch of training instances $\{(p_i, d_i^T)\}$, the batch loss is

$$Loss = -\sum_i \log \Pr(d_i^T|p_i) + \lambda \parallel \Theta \parallel^2 , \tag{7}$$

where the L2 norm of parameters are added with factor λ. Besides, we adopt min-batch Adam to optimize the model and update the parameters.

4 Experimental Setup

4.1 Dataset and Evaluation Measures

Training Data. Although ML is now widely used to assist with numerous medical tasks, publicly available datasets are limited. To train the proposed method we require datasets containing patient cases, consisting of reports of symptoms and associated diagnoses. The MIMIC-III [12] and the TREC Medical Records [26] datasets both contain patient records and associated diagnoses. However, MIMIC III data contains little information about symptoms, and the diagnosis codes (in ICD) do change over time during the patient encounter (no discharge diagnosis is recorded). MIMIC III also presents a strong bias in that the records relate to intensive care unit hospitalizations only. The TREC Medical Records dataset contains descriptions of complaints and symptoms for each patient encounter along with diagnoses (also at discharge); however it is not any more publicly available.

Previous work by Xia et al. [27] has shown that the abstracts from biomedical literature articles contain descriptions of diseases and associated key symptoms can be used for disease diagnosis. Motivated by this observation, we then constructed training instances from medical literature abstracts, following a similar procedure to that used by Xia et al. [27]. Specifically, we acquired biomedical abstracts annotated with UMLS concepts, made available from Medline 2019[4]. Then, we only selected articles associated with diseases and symptoms. Finally, we generated several cases from each abstract using the occurring symptoms as the description of patients and each occurring diseases as the possible diagnoses.

Test Collection. To test the effectiveness of automated diagnosis methods, we constructed a test collection using the free-text vignettes from a previous work that evaluated the correctness of symptom checkers [21]. These vignettes were sourced from clinical notes and text-book cases; each vignette contains a brief free-text description of the patient, a diagnosis made by a clinician, and a triage urgency (three levels: *emergent care is required*, *non-emergent care is reasonable*, and *self care is sufficient*).

In our collection, a test instance was constructed using a vignette by extracting symptom concepts from the patient's free-text description and mapping the free-text of the correct diagnosis provided for the patient case to a disease concept, using QuickUMLS [22], a tool that performs unsupervised biomedical concept extraction from free-text. When assembling our collection, we had to exclude two of the vignettes from the original dataset by Semigran et al. [21] as the free-text associated with the correct diagnosis could not be mapped to any disease concept by QuickUMLS. In total, 43 test instances were obtained for evaluation.

Limitation of Experiments. Our experimental findings are limited by the following factors: 1) the used test collection is small – this aspect makes it less likely experiments will detect statistical significant differences between methods 2) clinical notes are not available as training data and thus there may be a mismatch between training and test data, 3) the public medical KG we are using is noisy.

Evaluation Metrics. For each vignette, the ground truth contains only one correct diagnosis. In addition, when considering the medical diagnosis task, it is likely that end-users may be wanted only to consider a handful of diagnoses: the cognitive load of considering a large array of diagnoses would render a clinical decision support application for diagnosis recommendation not worth it. These characteristics are akin to the problem of known-item retrieval, with a strong preference on early rank retrieval, if not even a dismissal of results above a

[4] https://mbr.nlm.nih.gov/Download/MetaMapped_Medline/2019/MMO/.

certain rank cut-off. With this in mind, we select *hit*@*k* (with $k = 1, \ldots, 5$) as evaluation metrics for our experiments – *hit*@*k* $= 1$ if the correct diagnosis is ranked among the top k results, 0 otherwise. We also include *nDCG*@*k* in our evaluation. While we do not have graded relevance in our task at the moment, this may be introduced in the future if approximate matching of ground truth diagnosis was added. For example, a diagnosis may be considered as *partially* correct if it is a specification or generalisation of the ground truth diagnosis (e.g., tension headache vs. headache). Nevertheless, *nDCG*@*k*, unlike *hit*@*k*, does assign a discount to the rank position at which the correct diagnosis is retrieved, and thus it rewords methods that retrieve the correct diagnosis early in the ranking.

4.2 Baselines

To contextualise the effectiveness of the proposed method, we implemented a number of baseline systems for the disease diagnosis task. Naïve Bayes Classifier (NB) [27] and Multiple Layer Perceptron (MLP) [24] are two simple baselines commonly used for the disease prediction task. NB assumes all medical concepts are independent of each other, while MLP, as a multi-class classification model, assumes the disease concepts are independent. Deep Structured Semantic Models (DSSM) [11] is a representative neural matching model, which represents medical concepts as vectors, and then, similar to our method, matches a group of symptoms (associated to a patient) with disease concepts to obtain an overall similarity score, which is then used to rank diagnoses. ContextCare treats diagnosis ranking as a link prediction problem, similarly to what we do, but models the diagnosis pattern with an energy function, a popular method for link prediction task. The Graph-based Attention Model (GRAM) [5] and LSTM-KGAtt [28] address the task of risk prediction, e.g., mortality risk prediction, using time series data regarding the progression of the patient picture. We adapt these methods to the diagnosis prediction (ranking) task considered in this paper. GRAM obtains representations of medical concepts by combining their hierarchy information (ancestors) within their representations. LSTM-KGAtt incorporates the direct context of medical concepts in KG into the diagnosis process using the attention mechanism.

4.3 Parameter Settings

The GCN was implemented using Python 3.7, PyTorch 1.3.1 and DGL 0.4.3 (https://docs.dgl.ai/). The hyper-parameters were selected using the following strategies. The dimension of concept embeddings and node features in the graph share a single value. The number of hidden layers and the unit numbers of hidden layers in the two MLP modules are set to the same value. The hyper-parameters were optimised using grid-search and 5-fold cross-validation. The number of GCN layers was chosen from $\{1, 2, 3, 4, 5\}$, the dimension of features was selected from $\{100, 200, 400\}$, the number of MLP hidden layers was tuned in $\{0, 1, 2\}$, the unit number of MLP hidden layers was tuned amongst $\{100, 200, 400\}$, the dropout rate was chosen from $\{0.0, 0.1, 0.3, 0.5\}$. The learning rate was set as $1e^{-3}$ and reduces when the validation loss stops decreasing. The number of negative samples for matching was set to 1000 and the kaiming initializer was used to initialize the model parameters.

5 Results and Analysis

With our empirical experiments, we aimed to answer the following research questions related to the proposed KGCN method:

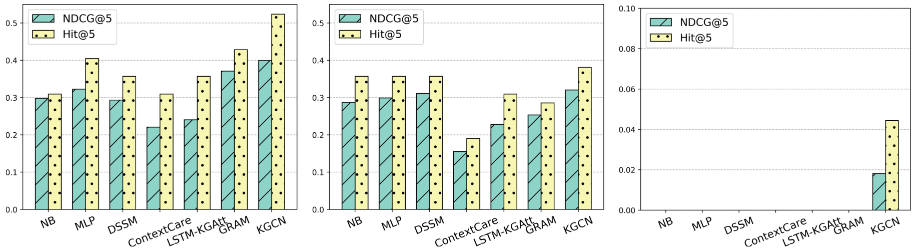
RQ1: Does our KGCN method outperform the baselines?
RQ2: How does our KGCN method perform with respect to the level of urgency of the patient case (triaging)?
RQ3: How does relationship type affect the effectiveness of our KGCN method?
RQ4: How does the fusion function affect the effectiveness of our KGCN method?
RQ5: How does the number of GCN layers affect the effectiveness of our KGCN method?

5.1 RQ1: Overall Effectiveness

Table 1 reports the overall effectiveness of each method. Note that none of the differences are statistically significant (paired t-test, $alpha = 0.05$); this is likely

Table 1. Overall effectiveness of methods for diagnosis ranking. The proposed KGCN achieved the best effectiveness across all metrics.

	Hit@1	Hit@2	Hit@3	Hit@4	Hit@5	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5
NB	0.1473	0.2093	0.2171	0.2171	0.2171	0.1473	0.1864	0.1903	0.1903	0.1903
MLP	0.1550	0.1860	0.2171	0.2248	0.2481	0.1550	0.1746	0.1901	0.1934	0.2024
DSSM	0.1550	0.1860	0.2171	0.2326	0.2326	0.1550	0.1746	0.1901	0.1968	0.1968
CtxCare	0.0775	0.1163	0.1318	0.1473	0.1628	0.0775	0.1020	0.1097	0.1164	0.1224
LSTM-KGAtt	0.0775	0.1473	0.1705	0.2093	0.2326	0.0775	0.1215	0.1332	0.1499	0.1589
GRAM	0.1550	0.2016	0.2326	0.2481	**0.2636**	0.1550	0.1844	0.1999	0.2066	0.2126
KGCN	**0.1783**	**0.2248**	**0.2403**	**0.2558**	**0.2636**	**0.1783**	**0.2076**	**0.2154**	**0.2221**	**0.2251**



(a) *Emergent care is required.* (b) *Non-emergent care is reasonable.* (c) *Self care is sufficient.*

Fig. 3. Effectiveness with respect to level of urgency. Note that all methods cannot find a correct diagnosis among the top 5 ranks for any of the self-care scenarios, apart from our KGCN, which does retrieve the correct diagnosis for a handful of self-care vignettes.

due to the limited number of vignettes and to all methods not identifying a correct diagnosis for a subset of cases (self-care vignettes, see Sect. 5.2) and thus obtaining the same evaluation scores in these cases.

NB and MLP, which are representative traditional methods for disease diagnosis, provided quite good effectiveness, especially when compared with more complex methods.

DSSM obtained similar performance to MLP, suggesting that formulating disease diagnosis as a matching problem does not effect effectiveness, while though offering greater flexibility in the way external knowledge can be incorporated.

ContextCare obtained the worst result: this highlights the limitation of the energy function in the diagnosis ranking task.

LSTM-KGAtt also performed poorly, although this method relied on the medical KG and thus exploits medical knowledge. This may be because the underlying LSTM architecture is not suitable for this task, even though it is widely adopted for tasks such as disease progression task.

GRAM provided improvements over NB, MLP and DSSM. This is done by exploiting the hierarchy information associated with medical concepts; a characteristic that simpler deep learning methods like MLP and DSSM do not model.

Finally, our model achieved the highest effectiveness across all metrics. Compared with MLP, our method is more flexible in that it exploits relationships between medical concepts. When compared with DSSM, we observe that our model does make effective use of the KG. Unlike GRAM, which only models hierarchy relationships, our method can model different types of knowledge in the medical KG: the empirical comparison with GRAM shows this is an important factor.

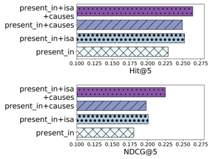


Fig. 4. Effect of medical knowledge.

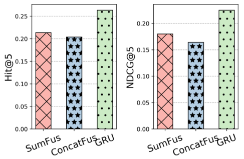


Fig. 5. Effect of fusion functions.

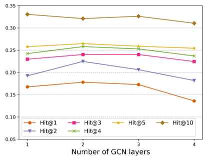


Fig. 6. Effect of the number of GCN layers.

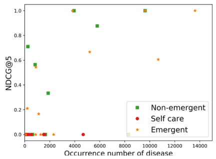


Fig. 7. Correlation between effectiveness and training data size.

5.2 RQ2: Effectiveness with Respect to the Level of Urgency (Triaging)

We further analyse the empirical results by considering the level of urgency (triaging) of each patient case. The results of our analysis are shown in Fig. 3 and suggest that KGCN outperforms other methods across all urgency levels. It also highlights how the effectiveness of the diagnosis ranking methods largely varies across the different levels of urgency, regardless of the actual method used. In particular, we find that all methods performed poorly for patient cases that required self-care, while they did perform well for the emergent and non-emergent care cases (vignettes).

We further analysed the results to understand why this may have been the case. In particular, we considered the number of occurrences of the target disease concepts used by the ground truth diagnoses in the vignettes. Specifically, we studied whether the effectiveness of KGCN was correlated with the number of such disease concepts in the data used for training (the analysis provided similar results for the other methods). Results are reported in Fig. 7 and suggest that the more a target disease concepts occurred in the training data, the better the KGCN performed on the associated patient case (vignette). We further analysed these results with respect to the level of urgency associated with each vignette. Diseases that require self-care were typically rare in the training data and indeed KGCN performed poorly on this type of patient cases. Conversely, diseases that require emergent and non-emergent care occurred more frequently in the training data, and our KGCN obtained higher effectiveness on these types of cases.

5.3 RQ3: Effect of Relationship Type

To explore the effect of the type of relationships (edges) present in a medical KG, we execute the proposed KGCN method on medical KGs populated with different combinations of relationship types. Our experiments considered three relationship types: *isa*, *present_in* and *causes*. The results of this comparison are reported in Fig. 4. When only *present_in* was used, our method performed worst. When adding to this relationships either *isa* or *causes*, effectiveness increased. This suggests that both hierarchy information and causality are helpful relationships for medical diagnosis. The best effectiveness is however achieved when all relationships are considered (*present_in+isa+causes*): this is likely because hierarchy and causality provide complementary information.

5.4 RQ4: Effect of Fusion Function

A key component of the proposed KGCN is the fusion of knowledge of different orders. To do so, our method relies on *GRU* as the fusion function, although we have indicated how other two widely used fusion functions, *SumFus* and *ConcatFus*, can also be used. In the next set of experiments, we compared the effectiveness of GRU compared to the two alternatives.

Empirical results related to this comparison are shown in Fig. 5. According to the results, the *GRU* substantially outperformed *SunFus* and *ConcatFus*, with the latter being the worst-performing fusion function amongst the three considered. performs the worst.

These results may be due to the fact that the architecture design of the *GRU* allows the parameters in low layers to be optimized better than when using the two alternative fusion functions. This caters to the fact that, for medical diagnosis, low-order information is more preferable than high-order knowledge. For example, if a clinician could have diagnosed a case simply by the symptoms, without considering the relationships between symptoms and conditions, they would not require the complex reasoning that underpins medical diagnosis. Another explanation for these results may be that the *GRU* fuses the representation of the input node and the aggregated context using their content interaction, while *SunFus* and *ConcatFus* can only combine them linearly. This advantage renders the model able to fuse these variables according to their contents. For instance, if the medical concepts do not have good representations, more medical knowledge would be needed.

5.5 RQ5: Effect of Number of GCN Layers

Finally, we analyzed the effect of the number of GCN layers in KGCN, while keeping the other hyper-parameters fixed. Overall, the KGCN method performs best when using two GCN layers, as shown in Fig. 6, while more GCN layers led to a decrease in diagnosis effectiveness. These results can be explained by that it is beneficial to aggregate more broad context to the representations of medical concepts and the patient in the disease diagnosis process. When the number of GCN layers is 3 or more, however, more noise is introduced; in addition, a model with more layers makes optimization more challenging.

6 Conclusions

In this paper we proposed a Knowledge Graph Convolutional Networks model, named KGCN, for ranking diagnosis. This method exploits medical KGs, which contain rich relations between medical concepts, in a more effective and general way compared with existing approaches. We formulated the disease diagnosis as a matching problem instead of a classification problem (as done in most of the previous work). To aggregate the medical knowledge for each concept in the KG and surface it with respect to the patient case at hand (patient node in the diagnosis graph), we exploited the message-passing mechanism of GCN to learn comprehensive concept representations. By stacking GCN layers, our model can propagate multi-hop contexts to each node.

Experiments were executed to assess the effectiveness of KGCN and tease out the aspects that influence its effectiveness. Our method outperformed existing approaches and we showed that both hierarchy and causality relationships provide complementary, valuable information for the diagnosis ranking task. We also compared different fusion functions in the context of KGCN, showing that the *GRU* fusion function outperformed the alternatives, and investigated the effect of the number of GCN layers and the availability of training data regarding the target ground-truth diagnosis had on effectiveness.

Our future work will consider two directions: (1) acquire more patient vignettes for evaluation, also including partial matches between diagnoses; (2) design special message-passing mechanisms within the GCN architecture for disease diagnosis. For example, we will explore a message-passing model with multiple channels to maintain the transitivity of hierarchy and causality relationships. Along this line, we will also consider exploiting a wider array of relationships.

Acknowledgements. This research is supported by the Shenyang Science and Technology Plan Fund (No. 20-201-4-10), the Member Program of Neusoft Research of Intelligent Healthcare Technology, Co. Ltd. (No. NRMP001901)). A/Prof Guido Zuccon is the recipient of an Australian Research Council DECRA Research Fellowship (DE180101579) and a Google Faculty Award.

References

1. Amato, F., López, A., Peña-Méndez, E.M., Vaňhara, P., Hampl, A., Havel, J.: Artificial neural networks in medical diagnosis (2013)
2. Aronson, A.R., Lang, F.: An overview of metamap: historical perspective and recent advances. J. Am. Med. Inform. Assoc. **17**(3), 229–236 (2010)
3. Bodenreider, O.: The unified medical language system (UMLS): integrating biomedical terminology. Nucleic Acid Res. **32**(suppl–1), D267–D270 (2004)
4. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. In: Bengio, Y., LeCun, Y. (eds.) 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14–16, 2014, Conference Track Proceedings (2014)
5. Choi, E., Bahadori, M.T., Song, L., Stewart, W.F., Sun, J.: GRAM: graph-based attention model for healthcare representation learning. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13–17, 2017, pp. 787–795. ACM (2017)
6. Choi, E., Xiao, C., Stewart, W.F., Sun, J.: Mime: multilevel medical embedding of electronic health records for predictive healthcare. In: Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3–8 December 2018, Canada, Montréal, pp. 4552–4562 (2018)
7. Ernst, P., Siu, A., Weikum, G.: Knowlife: a versatile approach for constructing a large knowledge graph for biomedical sciences. BMC Bioinform. **16**, 157:1–157:13 (2015)
8. Ernst, P., Siu, A., Weikum, G.: Highlife: higher-arity fact harvesting. In: Champin, P., Gandon, F.L., Lalmas, M., Ipeirotis, P.G. (eds.) Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23–27, 2018, pp. 1013–1022. ACM (2018)
9. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017. Proceedings of Machine Learning Research, PMLR, vol. 70, pp. 1263–1272 (2017)
10. Huang, J., Zhao, W.X., Dou, H., Wen, J., Chang, E.Y.: Improving sequential recommendation with knowledge-enhanced memory networks. In: Collins-Thompson, K., Mei, Q., Davison, B.D., Liu, Y., Yilmaz, E. (eds.) The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08–12, 2018, pp. 505–514. ACM (2018)
11. Huang, P., He, X., Gao, J., Deng, L., Acero, A., Heck, L.P.: Learning deep structured semantic models for web search using clickthrough data. In: He, Q., Iyengar, A., Nejdl, W., Pei, J., Rastogi, R. (eds.) 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, San Francisco, CA, USA, 27 October–1 November 2013, pp. 2333–2338. ACM (2013)

12. Johnson, A.E., et al.: MIMIC-III, a freely accessible critical care database. Sci. Data **3**(1), 1–9 (2016)
13. Kilicoglu, H., Shin, D., Fiszman, M., Rosemblat, G., Rindflesch, T.C.: SemMedDB: a PubMed-scale repository of biomedical semantic predications. Bioinform. **28**(23), 3158–3160 (2012)
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings. OpenReview.net (2017)
15. Kononenko, I.: Machine learning for medical diagnosis: history, state of the art and perspective. Artif. Intell. Med. **23**(1), 89–109 (2001). https://doi.org/10.1016/S0933-3657(01)00077-X
16. Kononenko, I.: Machine learning for medical diagnosis: history, state of the art and perspective. Artif. Intell. Med. **23**(1), 89–109 (2001)
17. Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.S.: Gated graph sequence neural networks. In: Bengio, Y., LeCun, Y. (eds.) 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings (2016)
18. Ma, F., Gao, J., Suo, Q., You, Q., Zhou, J., Zhang, A.: Risk prediction on electronic health records with prior medical knowledge. In: Guo, Y., Farooq, F. (eds.) Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19–23, 2018, pp. 1910–1919. ACM (2018)
19. Rotmensch, M., Halpern, Y., Tlimat, A., Horng, S., Sontag, D.: Learning a health knowledge graph from electronic medical records. Sci. Reports **7**(1), 1–11 (2017)
20. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Netw. **20**(1), 61–80 (2009)
21. Semigran, H.L., Linder, J.A., Gidengil, C., Mehrotra, A.: Evaluation of symptom checkers for self diagnosis and triage: audit study. bmj **351**, h3480 (2015)
22. Soldaini, L., Goharian, N.: Quickumls: a fast, unsupervised approach for medical concept extraction. In: MedIR workshop, sigir, pp. 1–4 (2016)
23. Stearns, M.Q., Price, C., Spackman, K.A., Wang, A.Y.: SNOMED clinical terms: overview of the development process and project status. In: AMIA 2001, American Medical Informatics Association Annual Symposium, Washington, DC, USA, November 3–7, 2001. AMIA (2001)
24. Šter, B., Dobnikar, A.: Neural networks in medical diagnosis: comparison with other methods. In: International Conference on Engineering Applications of Neural Networks, pp. 427–430 (1996)
25. Stern, S.D.: Symptom To Diagnosis An Evidence-Based Guide. Second Edition, New York (NY): McGraw-Hill Education/Medical (2010)
26. Voorhees, E.M., Hersh, W.R.: Overview of the TREC 2012 medical records track. In: TREC (2012)
27. Xia, E., Sun, W., Mei, J., Xu, E., Wang, K., Qin, Y.: Mining disease-symptom relation from massive biomedical literature and its application in severe disease diagnosis. In: AMIA Annual Symposium Proceedings. American Medical Informatics Association, vol. 2018, p. 1118 (2018)
28. Yin, C., Zhao, R., Qian, B., Lv, X., Zhang, P.: Domain knowledge guided deep learning with electronic health records. In: Wang, J., Shim, K., Wu, X. (eds.) 2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8–11, 2019, pp. 738–747. IEEE (2019)

Studying Catastrophic Forgetting in Neural Ranking Models

Jesús Lovón-Melgarejo[1](✉), Laure Soulier[2], Karen Pinel-Sauvagnat[1], and Lynda Tamine[1]

[1] Université Paul Sabatier, IRIT, Toulouse, France
{jesus.lovon,sauvagnat,tamine}@irit.fr
[2] Sorbonne Université, CNRS, LIP6, 75005 Paris, France
laure.soulier@lip6.fr

Abstract. Several deep neural ranking models have been proposed in the recent IR literature. While their transferability to one target domain held by a dataset has been widely addressed using traditional domain adaptation strategies, the question of their cross-domain transferability is still under-studied. We study here in what extent neural ranking models catastrophically forget old knowledge acquired from previously observed domains after acquiring new knowledge, leading to performance decrease on those domains. Our experiments show that the effectiveness of neural IR ranking models is achieved at the cost of catastrophic forgetting and that a lifelong learning strategy using a cross-domain regularizer successfully mitigates the problem. Using an explanatory approach built on a regression model, we also show the effect of domain characteristics on the rise of catastrophic forgetting. We believe that the obtained results can be useful for both theoretical and practical future work in neural IR.

Keywords: Neural ranking · Catastrophic forgetting · Lifelong learning

1 Introduction

Neural ranking models have been increasingly adopted in the information retrieval (IR) and natural language processing (NLP) communities for a wide range of data and tasks [35,39]. One common underlying issue is that they learn relationships that may hold only in the domain from which the training data is sampled, and generalize poorly in unobserved domains[1] [6,39]. To enhance the transferability of neural ranking models from a source domain to a target domain, transfer learning strategies such as fine-tuning [52], multi-tasking [29], domain adaptation [40], and more recently adversarial learning [7], have

[1] According to Jialin and Qiang [40], a domain consists of at most two components: a feature space over a dataset and a marginal probability distribution within a task.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 375–390, 2021.
https://doi.org/10.1007/978-3-030-72113-8_25

been widely used[2]. However, these strategies have by essence two critical limitations reported in the machine learning literature [6,22]. The first one, as can be acknowledged in the NLP and IR communities [7,29], is that they require all the domains to be available simultaneously at the learning stage (except the fine-tuning). The second limitation, under-studied in both communities, is that the model leans to *catastrophically forget* existing knowledge (source domain) when the learning is transferred to new knowledge (target domain) leading to a significant drop of performance on the source domain. These limitations are particularly thorny when considering open-domain IR tasks including, but not limited to, conversational search. In the underlying settings (e.g., QA systems and chatbots [15,25,33,42]), neural ranking models are expected to continually learn features from online information streams, sampled from either observed or unobserved domains, and to scale across different domains but without forgetting previously learned knowledge.

Catastrophic forgetting is a long-standing problem addressed in machine learning using *lifelong learning* approaches [6,41]. It has been particularly studied in neural-network based classification tasks in computer vision [22,26] and more recently in NLP [32,37,45,48]. However, while previous work showed that the level of catastrophic forgetting is significantly impacted by dataset features and network architectures, we are not aware of any existing research in IR providing clear lessons about the transferability of neural ranking models across domains, nor basically showing if state-of-the-art neural ranking models are actually faced with the catastrophic forgetting problem and how to overcome it if any. Understanding the conditions under which these models forget accumulated knowledge and whether a lifelong learning strategy is a feasible way for improving their effectiveness, would bring important lessons for both practical and theoretical work in IR. This work contributes to fill this gap identified in the neural IR literature, by studying the transferability of ranking models. We put the focus on catastrophic forgetting which is the bottleneck of lifelong learning.

The main contributions of this paper are as follows. 1) We show the occurrence of catastrophic forgetting in neural ranking models. We investigate the transfer learning of five representative neural ranking models (DRMM [14], PACRR [17], KNRM [49], V-BERT [31] and CEDR [31]) over streams of datasets from different domains[3] (MS MARCO [3], TREC Microblog [44] and TREC COVID19 [46]); 2) We identify domain characteristics such as relevance density as signals of catastrophic forgetting; 3) We show the effectiveness of constraining the objective function of the neural IR models with a forget cost term, to mitigate the catastrophic forgetting.

[2] We consider the definition of transfer learning in [40]. Please note that several other definitions exist [13].

[3] In our work, different domains refer to different datasets characterized by different data distributions w.r.t. to their source and content as defined in [40].

2 Background and Related Work

From Domain Adaptation to Lifelong Learning of Neural Networks. Neural networks are learning systems that must commonly, on the one hand, exhibit the ability to acquire new knowledge and, on the other hand, exhibit robustness by refining knowledge while maintaining stable performance on existing knowledge. While the acquisition of new knowledge gives rise to the well-known *domain shift* problem [18], maintaining model performance on existing knowledge is faced with the *catastrophic forgetting* problem. Those problems have been respectively tackled using *domain adaptation* [40] and *lifelong learning* strategies [6,41]. Domain adaptation, commonly known as a specific setting of *transfer learning* [40], includes machine learning methods (e.g., fine-tuning [48] and multi-tasking [29]) that assume that the source and the target domains from which are sampled respectively the training and testing data might have different distributions. By applying a transfer learning method, a neural model should acquire new specialized knowledge from the target domain leading to optimal performance on it.

One of the main issues behind common transfer learning approaches is catastrophic forgetting [11,12]: the newly acquired knowledge interfers with, at the worst case, overwrites, the existing knowledge leading to performance decrease on information sampled from the latter. Lifelong learning [6,41] tackles this issue by enhancing the models with the ability to continuously learn over time and accumulate knowledge from streams of information sampled across domains, either previously observed or not. The three common lifelong learning approaches are [41]: 1) regularization that constrains the objective function with a forget cost term [22,26,48]; 2) network expansion that adapts the network architecture to new tasks by adding neurons and layers [5,43]; and 3) memory models that retrain the network using instances selected from a memory drawn from different data distributions [2,32].

On the Transferability of Neural Networks in NLP and IR. Transferability of neural networks has been particularly studied in classification tasks, first in computer vision [4,53] and then only recently in NLP [19,37,38]. For instance, Mou et al. [38] investigated the transferability of neural networks in sentence classification and sentence-pair classification tasks. One of their main findings is that transferability across domains depends on the level of similarity between the considered tasks. Unlikely, previous work in IR which mainly involves ranking tasks, have only casually applied transfer learning methods (e.g., fine-tuning [52], multi-tasking [29] and adversarial learning [7]) without bringing generalizable lessons about the transferability of neural ranking models. One consensual result reported across previous research in the area, is that traditional retrieval models (e.g., learning-to-rank models [28]) that make fewer distributional assumptions, exhibit more robust cross-domain performances [7,39]. Besides, it has been shown that the ability of neural ranking models to learn new features may be achieved at the cost of poor performances on domains not observed during training [35]. Another consensual result is that although

embeddings are trained using large scale corpora, they are generally sub-optimal for domain-specific ranking tasks [39].
Beyond domain adaptation, there is a recent research trend in NLP toward lifelong learning of neural networks, particularly in machine translation [45], and language understanding tasks [37,48,50]. For instance, Xu et al. [50] recently revisited the domain transferability of traditional word embeddings [34] and proposed *lifelong domain embeddings* using a meta-learning approach. The proposed meta-learner is fine-tuned to identify similar contexts of the same word in both past domains and the new observed domain. Thus, its inference model is able to compute the similarity scores on pairs of feature vectors representing the same word across domains. These embeddings have been successfully applied to a topic-classification task. Unlikely, catastrophic forgetting and lifelong learning are still under-studied in IR. We believe that a thorough analysis of the transferability of neural ranking models from a lifelong learning perspective would be desirable for a wide range of emerging open-domain IR applications including but not limited to conversational search [15,25,33,42].

3 Study Design

Our study mainly addresses the following research questions:

RQ1: Does catastrophic forgetting occur in neural ranking models?
RQ2: What are the dataset characteristics that predict catastrophic forgetting?
RQ3: Is a regularization-based lifelong learning method effective to mitigate catastrophic forgetting in neural ranking models?

3.1 Experimental Set up

Given a neural model M designed for an ad-hoc ranking task, the primary objectives of our experiments are twofold: O1) measuring the catastrophic forgetting of model M while applying a domain adaptation method $\mathcal{D}$, in line of RQ1 and RQ2; and O2) evaluating the effect of a lifelong learning method $\mathcal{L}$ to mitigate catastrophic forgetting in model M, in line of RQ3. We assume that model M learns a ranking task across a stream of n domain datasets $\{D_1, \ldots, D_n\}$ coming in a sequential manner one by one. At a high level, our experimental set up is:

1. Set up an ordered dataset stream **setting** $D_1 \rightarrow \ldots D_{n-1} \rightarrow D_n$
2. Learn oracle models $M_i^*, i = 1 \ldots n$, with parameters $\hat{\theta}^{i*}$ by training **the neural ranking model** M on training instances of **dataset** $D_i, i = 1 \ldots n$.
3. Measure the retrieval performance $R_{i,i}^*$ of each oracle model M_i^* on testing instances of the same dataset D_i.
4. Launch a **domain adaptation method** $\mathcal{D}$ w.r.t. to objective O1 (resp. a **lifelong learning method** $\mathcal{L}$ w.r.t. to objective O2) along the considered setting as follows:
   – Initialize ($k = 1$) model M_k, with $\hat{\theta}^{1*}$, parameters of model M_1^* (trained on the dataset base D_1).
   – Repeat
      - Apply to model M_k a method $\mathcal{D}$ w.r.t to objective O1 (resp. method $\mathcal{L}$ w.r.t. to objective O2) to transfer knowledge to the right dataset D_{k+1} (forward transfer). The resulting model is noted M_{k+1} with parameters $\hat{\theta}^{k+1}$. Its performance on dataset D_{k+1} is noted $R_{k+1,k+1}$.
      - Measure the retrieval performance $R_{k+1,k}$ of model M_{k+1} obtained on the testing instances of left dataset D_k (backward transfer)
      - Move to the next right dataset : $k = k + 1$
   – Until the end of the dataset stream setting ($k = n$).
5. **Measure catastrophic forgetting** in model M.

This experimental pipeline, illustrated in Fig. 1, follows general guidelines adopted in previous work [2,20,26]. We detail below the main underlying components highlighted in bold.

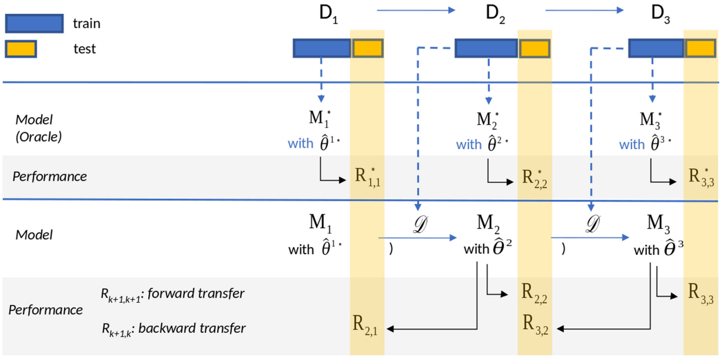


Fig. 1. Experimental pipeline using a 3-dataset stream setting for a given model M

Neural Ranking Models. We evaluate catastrophic forgetting in five (5) state-of-the-art models selected from a list of models critically evaluated in Yang et al. [51]: 1) interaction-based models: DRMM [14] and PACRR [17] and KNRM [49]; 2) BERT-based models: Vanilla BERT [31] and CEDR-KNRM [31]. We use the OpenNIR framework [30] that provides a complete neural ad-hoc document ranking pipeline. Note that in this framework, the neural models are trained by linearly combining their own score (S_{NN}) with a BM25 score (S_{BM25}).

Datasets and Settings. We use the three following datasets: 1) MS MARCO (***ms***) [3] a passage ranking dataset which includes more than 864 K question-alike queries sampled from the Bing search log and a large-scale web document set including 8841823 documents; 2) TREC Microblog (***mb***) [27], a real-time ad-hoc search dataset from TREC Microblog 2013 and 2014, which contains a public Twitter sample stream between February 1 and March 31, 2013 including 124969835 tweets and 115 queries submitted at a specific point in time; 3) TREC CORD19 (***c19***) [46] an ad-hoc document search dataset which contains 50 question-alike queries and a corpora of 191175 published research articles dealing with SARS-CoV-2 or COVID-19 topics. It is worth mentioning that these datasets fit with the requirement of cross-domain adaptation [40] since they have significant differences in both their content and sources. Besides, we consider four settings (See Table 1, column **"Setting"**) among which three 2-dataset ($n = 2$) and one 3-dataset ($n = 3$) settings. As done in previous work [2,26], these settings follow the patterns ($D_1 \rightarrow D_2$) or ($D_1 \rightarrow D_2 \rightarrow D_3$) where dataset orders are based on the decreasing sizes of the training sets assuming that larger datasets allow starting with well-trained networks.

Domain Adaptation and Lifelong Learning Methods. We adopt fine-tuning (training on one domain and fine-tuning on the other) as the representative domain adaptation task $\mathcal{D}$ since it suffers from the catastrophic forgetting problem [2,22]. Additionally, we adopt the Elastic Weight Consolidation (EWC) [22] as the lifelong learning method $\mathcal{L}$. The EWC constrains the loss function with an additional forget cost term that we add to the objective function of each of the five neural models studied in this work. Basically speaking, EWC constrains the neural network-based model to remember knowledge acquired on left datasets by reducing the overwriting of its most important parameters as:

$$\mathcal{L}(\hat{\theta}^k) = \mathcal{L}(\hat{\theta}^k) + \varSigma_{1 \leq i < k} \frac{\lambda}{2} \mathcal{F}_i (\hat{\theta}^k - \hat{\theta}^i)^2 \tag{1}$$

where $\mathcal{L}(\hat{\theta}^k)$ is the loss of the neural ranking model with parameters θ^k obtained right after learning on (D_k), λ is the importance weight of the models parameters trained on left datasets ($D_i, i < k$) with the current one (D_k), $\mathcal{F}$ is the Fisher information matrix.

Measures. Given the setting ($D_1 \rightarrow \cdots \rightarrow D_n$), we use the *remembering* measure (REM) derived from the *backward transfer measure* (BWT) proposed by Rodriguez et al. [10] as follows:

- **BWT:** measures the intrinsic effect (either positive or negative) that learning a model M on a new dataset (right in the setting) has on the model performance obtained on an old dataset (left in the setting), referred as *backward transfer*. Practically, in line with a lifelong learning perspective, this measure averages along the setting the differences between the performances of the model obtained right after learning on the left dataset and the performances of the oracle model trained and tested on the same left dataset. Thus, while positive values handle positive backward transfer, negative values handle catastrophic forgetting.

Formally, the BWT measure is computed as:

$$BWT = \frac{\sum_{i=2}^{n}\sum_{j=1}^{i-1}(R_{i,j} - R^*_{j,j})}{\frac{n(n-1)}{2}} \tag{2}$$

$R_{i,j}$ is the performance measure of model M_i obtained right after learning on dataset D_j. $R^*_{j,j}$ is the performance of the oracle model M^*_j trained on dataset D_j and tested on the same dataset. To make fair comparisons between the different studied neural models, we normalize the differences in performance $(R_{i,j} - R^*_{j,j})$ on model agnostic performances obtained using $BM25$ model on each left dataset D_j. In our work, we use the standard IR performance measures MAP, NDCG@20 and P@20 to measure $R_{i,j}$ but we only report the REM values computed using the MAP measure, as they all follow the same general trends.

• **REM**: because the BWT measure assumes either positive values for positive backward transfer and negative values for catastrophic forgetting, it allows to map with a positive remembering value in the range $[0, 1]$ as follows:

$$REM = 1 - |min(BWT, 0)| \tag{3}$$

A REM value equals to 1 means that the model does not catastrophically forget.

To better measure the intrinsic ability of the neural ranking models to remember previously acquired knowledge, we deploy in the OpenNIR framework two runs for each neural model based on the score combination ($score_G = \alpha \times S_{NN} + (1 - \alpha) \times S_{BM25}$). The first one by considering the neural model after a re-ranking setup ($0 < \alpha < 1$) leading to compute an overall REM measure on the ranking model. The second one by only considering the neural ranking based on the S_{NN} score by totally disregarding the BM25 scores ($\alpha = 1$). $REMN$ denotes the remembering measure computed in this second run.

3.2 Implementation Details

We use the OpenNIR framework with default parameters and the pairwise hinge loss function [8]. To feed the neural ranking models, we use the GloVe pre-trained embeddings (42b tokens and 300d vectors). The datasets are split into training and testing instance sets. For MS MARCO, we use the default splits provided in the dataset. For TREC CORD19 and TREC Microblog, where no training instances are provided, we adopt the splits by proportions leading to 27/18 and 92/23 training/testing queries respectively. In practice, we pre-rank documents using the BM25 model. For each relevant document-query pair (positive pair), we randomly sample a document for the same query with a lower relevance score to build the negative pair. We re-rank the top-100 BM25 results and use P@20 to select the best-performing model. For each dataset, we use the optimal BM25 hyperparameters selected using grid-search. In the training phase, we consider a maximum of 100 epochs or early-stopping if no further improvement is found. Each epoch consists of 32 batches of 16 training pairs. All the models are optimized using Adam [21] with a learning rate of 0.001. BERT layers are trained at a rate of 2e−5 following previous work [31]. For the EWC, we fixed $\lambda = 0.5$. The code is available at https://github.com/jeslev/OpenNIR-Lifelong.

4 Results

4.1 Empirical Analysis of Catastrophic Forgetting in Neural Ranking Models

Within- and Across-Model Analysis (RQ1). Our objective here is to investigate whether each of the studied neural models suffer from catastrophic forgetting while it is fine-tuned over a setting ($D_1 \rightarrow D_2$ or $D_1 \rightarrow D_2 \rightarrow D_3$). To carry out a thorough analysis of each model-setting pair, we compute the following measures in addition to the REM/REMN measures: 1) the MAP@100 performance ratio ($PR = \frac{1}{(n-1)} \sum_{i=2}^{n} \frac{R_{i,i}}{R^*_{i,i}}$) of the model learned on the right dataset and normalized on the oracle model performance; 2) the relative improvement in MAP@100 Δ_{MAP} (resp. Δ_{MAPN}) achieved with the ranking based on the global relevance score $Score_G$ (resp. $Score_{NN}$) trained and tested on the left dataset over the performance of the BM25 ranking obtained on the same testing dataset. Table 1 reports all the metric values for each model/setting pairwise. In line with this experiment's objective, we focus on the **"Fine-tuning"** columns.

Looking first at the PR measure reported in Table 1, we notice that it is greater than 0.96 in 100% of the settings, showing that the fine-tuned models are successful on the right dataset, and thus allow a reliable investigation of catastrophic forgetting as outlined in previous work [37]. It is worth recalling that the general evaluation framework is based on a pre-ranking (using the BM25 model) which is expected to provide positive training instances from the left dataset to the neural ranking model being fine-tuned on the right dataset.

Table 1. Per model-setting results in our fine-tuning and EWC-based lifelong learning experiments. All the measures are based on the MAP@100 metric. The improvements $\Delta_{MAP(MAPN)}$ and $\Delta_{REM(REMN)}$ are reported in percent (%).

Model	Setting	Fine-tuning				EWC-based lifelong learning		
		REM(REMN)	$\Delta_{MAP(MAPN)}$	$\Delta_{MAP(MAPN)}$	*PR*	*REM(REMN)*	$\Delta_{REM(REMN)}$	*PR*
DRMM	$ms \rightarrow c19$	1.000(1.000)	0.023(−0.715)	+2.2(−73.6)	1.008	1.000(1.000)	0(0)	1.005
	$ms \rightarrow mb$	0.962(0.943)	−0.017(−0.793)	+2.2(−73.6)	1.021	0.971(0.974)	+0.9(**+3.3**)	1.011
	$mb \rightarrow c19$	1.000(0.965)	−0.017(−0.112)	−1.7(−7.7)	0.993	1.000(0.662)	0(−31.4)	0.995
	$ms \rightarrow mb \rightarrow c19$	0.976(0.938)	−0.008(−0.726)	+2(−73.6)	1.011	0.979(1.000)	**+0.3**(**+6.6**)	1.004
PACRR	$ms \rightarrow c19$	1.000(0.760)	0.026(-0.54)	+2.5(−30.1)	1.000	1.000(0.756)	0(−0.5)	1.000
	$ms \rightarrow mb$	1.000(1.000)	0.026(−0.243)	+2.5(−30.1)	0.999	1.000(1.000)	0(0)	1.014
	$mb \rightarrow c19$	1.000(0.523)	−0(−37.6)	0(+10)	1.000	1.000(0.940)	0(**+79.7**)	1.002
	$ms \rightarrow mb \rightarrow c19$	1.000(0.759)	0.026(-0.636)	+2.5(−30)	1.000	1.000(0.874)	0(**+15.2**)	1.015
KNRM	$ms \rightarrow c19$	1.000(1.000)	−0.032(−0.862)	−12.1(−89)	1.069	1.000(1.000)	0(0)	1.058
	$ms \rightarrow mb$	1.000(1.000)	−0.088(−0.784)	−12.1(−89)	0.991	1.000(1.000)	0(0)	0.991
	$mb \rightarrow c19$	1.000(0.810)	0.011 (−0.328)	−2(−13.8)	1.135	1.000(0.902)	0(**+11.4**)	1.141
	$ms \rightarrow mb \rightarrow c19$	1.000(1.000)	−0.045(−0.802)	−12.1(−89)	1.086	1.000(0.963)	0(−3.7)	1.087
VBERT	$ms \rightarrow c19$	0.930(1.000)	−0.175(0.006)	−10.6(0)	1.028	1.000(1.000)	**+7.5**(0)	0.990
	$ms \rightarrow mb$	1.000(0.883)	−0.003(−0.111)	−10.6(0)	1.030	1.000(1.000)	0(**+13.3**)	0.992
	$mb \rightarrow c19$	0.913(1.000)	0.086(0.258)	+17.4(+25.8)	0.963	1.000(1.000)	**+9.5**(0)	1.010
	$ms \rightarrow mb \rightarrow c19$	0.989(0.922)	−0.145(−0.111)	−10.6(0)	1.011	1.000(1.000)	**+1.1**(**+8.5**)	0.987
CEDR	$ms \rightarrow c19$	0.826(1.000)	−0.148(0.142)	+2.6(+14.2)	1.013	1.000(1.000)	**+21.1**(0)	1.008
	$ms \rightarrow mb$	0.510(0.920)	−0.463(0.062)	+2.6(+14.2)	1.003	1.000(1.000)	**+96.1**(**+8.7**)	0.976
	$mb \rightarrow c19$	0.940(1.000)	0.136(0.292)	+19.6(+29.2)	1.011	1.000(1.000)	**+6.4**(0)	0.984
	$ms \rightarrow mb \rightarrow c19$	0.771(0.946)	−0.194(0.062)	+2.6(+14.2)	0.996	0.891(1.000)	**+15.6**(**+5.7**)	0.961

The joint comparison of the REM (resp. $REMN$) and $\varDelta_{MAP}$ (resp.$\varDelta_{MAPN}$) measures lead us to highlight the following statements:

• We observe that only CEDR and VBERT models achieve positive improvements w.r.t to both the global ranking ($\varDelta_{MAP}$: +19.6%, +17.4% resp.) and the neural ranking ($\varDelta_{MAP}$: +29.2%, +25.8% resp.), particularly under the setting where mb is the left dataset ($mb \rightarrow c19$). Both models are able to bring effectiveness gains additively to those brought by the exact-based matching signals in BM25. These effectiveness gains can be viewed as new knowledge in the form of semantic matching signals which are successfully transferred to the left dataset ($c19$) while maintaining stable performances on the left dataset (mb) (REMN=0.940 and 0.913 for resp. CEDR and VBERT). This result is consistent with previous work suggesting that the regularization used in transformer-based models has an effect of alleviating catastrophic forgetting [23].

• We notice that the CEDR model achieves positive improvements w.r.t to the neural ranking score ($\varDelta_{MAPN}$: +14.2%) in all the settings (3/4) where ms is the left dataset while very low improvements are achieved w.r.t. to the global score ($\varDelta_{MAP}$: +2.6%). We make the same observation for the PACRR model but only for 1/4 model-setting pair ($\varDelta_{MAPN}$: +10% vs. $\varDelta_{MAPN}$: 0%) with mb as the left dataset. Under these settings, we can see that even the exact-matching signals brought by the BM25 model are very moderate (leading to a very few positive training instances), the CEDR and, to a lower extent, the PACRR models, are able to inherently bring significant new knowledge in terms of semantic matching signals at however the cost of significant forget on the global ranking for CEDR (REM is the range [0.510; 0.826]) and on the neural ranking for PACRR (REM=0.523).

• All the models (DRMM, PACRR, KNRM and VBERT (for 3/4 settings) that do not significantly beat the BM25 baseline either by using the global score ($\varDelta_{MAP}$ in the range [−12.1%; +2.2%]) nor by using the neural score ($\varDelta_{MAPN}$ in the range [−89%; +0%]), achieve near upper bound of remembering (both REM and REMN are in the range [0.94; 1]). Paradoxically, this result does not allow us to argue about the ability of these models to retain old knowledge. Indeed, the lack or even the low improvements over both the exact matching (using the BM25 model) and the semantic-matching (using the neural model) indicate that a moderate amount of new knowledge or even no knowledge about effective relevance ranking has been acquired from the left dataset. Thus, the ranking performance of the fine-tuned model on the left dataset only depends on the level of mismatch between the data available in the right dataset for training and the test data in the left dataset. We can interestingly see that upper bound remembering performance ($REM = 1$) is particularly achieved when ms is the left dataset (settings $ms \rightarrow c19$, $ms \rightarrow mb$, $ms \rightarrow mb \rightarrow c19$). This could be explained by the fact that the relevance matching signals learned by the neural model in in-domain knowledge do not degrade its performances on general-domain knowledge.

Assuming a well-established practice in neural IR which consists in linearly interpolating the neural scores with the exact-based matching scores (e.g., BM25 scores), these observations give rise to three main findings: 1) the more a neural ranking model is inherently effective in learning additional semantic matching signals, the more likely it catastrophically forgets. In other terms, intrinsic effectiveness of neural ranking models comes at the cost of forget; 2) transformer-based language models such as CEDR and VBERT exhibit a good balance between effectiveness and forget as reported in previous work in NLP [37]; 3) given the variation observed in REM and REMN, there is no clear trend about which ranking (BM25-based ranking vs. neural ranking) impacts more importantly the level of overall catastrophic forgetting of the neural models

Across Dataset Analysis (RQ2). Our objective here is to identify catastrophic forgetting signals from the perspective of the left dataset. As argued in [1], understanding the relationships between data characteristics and catastrophic forgetting allows to anticipating the choice of datasets in lifelong learning settings regardless of the neural ranking models. We perform a regression model to explain the REM metric (dependent variable) using nine datasets characteristics (independent variables). The latter are presented in Table 2 and include dataset-based measures inspired from [1,9] and effectiveness-based measures using the BM25 model.

Table 2. Linear regression explaining catastrophic forgetting (REM metric) at the left dataset level. Significance: $* * * : p \leq 0.001; ** : 0.001 < p \leq 0.01; * : 0.01 < p \leq 0.5$

		Characteristic	Description	Coeff
		R^2		0.544
Independent variables	Dataset	Constant		0.7014$* * *$
		RS	Retrieval space size: $log_{10}(D \times Q)$	-0.1883
		RD	Relevance density: $log_{10} \frac{Qrels}{D \times Q}$	$-0.3997*$
		SD	Score relevance divergence: $KL(RSV_{D+}, RSV_{D-})$	0.0009
		Vocab	Size of the vocabulary	-0.0932 $*$
		DL	Average length of documents	-0.0349
		QL	Average length of queries	0.1803$*$
		QD	Average query difficulty: $avg_q(\frac{1}{\lvert q\rvert} \sum_{w \in q} idf_w)$	0.0044
	Eff.	MAP	Effectiveness of the BM25: MAP	-0.0220$*$
		std-AP	Variation of BM25 effectiveness (AP metric): $\sigma_q(AP_q)$	0.0543$*$
Residual Variables		$Dataset_i$	MSmarco	0.18038
			Microblog	0.5211$**$
		M_j	DRMM	0.1798$* * *$
			PACRR	0.1965$* * *$
			KNRM	0.1924$* * *$
			VBERT	0.1313$* * *$
			CEDR	0.0014

To artificially-generate datasets with varying data characteristics, we follow the procedure detailed in [1]: we sample queries within each left dataset in the settings presented in Table 1 (15 for *mb* and 50 for *ms*) to create sub-datasets composed of those selected queries and the 100 top corresponding documents retrieved by the BM25 model.

Then, we replace in each setting the left dataset by the corresponding sub-dataset. We estimate for each model-setting pair the REM value as well as the characteristic values of the left sub-dataset. We repeat this procedure 300 times to obtain 300 new settings per model, based on the 300 sampled sub-datasets. This leads to 300 (sub-setting-model) pairs with a variation for both the dependent and the independent variables. Finally, we build the following explanatory regression model, referring to the "across dataset analysis" in [1]:

$$REM_{ij} = \sum_k C_k f_{ik} + Dataset_i + M_j + \epsilon_i \tag{4}$$

where i denotes the i^{th} sub-setting and j refers to the neural ranking model M_j. C_k and f_{ik} denote respectively the weight and the value of the k^{th} characteristic of the left dataset in the i^{th} sub-setting. Please note, that dataset feature values are independent of the model M_j. $Dataset_i$ and M_j are the residual variables of resp. the left dataset and the model. The characteristic values f_{ik} are centered before the regression as suggested in Adomavicius and Zhang [1].

Table 2 presents the result of the regression model. From R^2 and $Constant$, we can see that our regression model can explain 54.4% of the variation of the REM metric, highlighting an overall good performance in explaining the remembering metric with a good level of prediction (0.7014). From the independent variables, we can infer that the difficulty of the dataset positively impacts the remembering (namely, decreasing the catastrophic forgetting). More precisely, lower the relevance density (RD), the BM25 effectiveness (MAP) and higher the variation in terms of BM25 performances over queries (std-AP) are, the higher the REM metric is. This suggests that relevance-matching difficulty provides positive feedback signals to the neural model to face diverse learning instances, and therefore to better generalize over different application domains. This is however true to the constraint that the vocabulary of the dataset ($Vocab$) is not too large so as to boost neural ranking performance as outlined in [16,36]. Looking at the residual variables ($Dataset_j$ and M_j), we can corroborate the observations made at a first glance in RQ1 regarding the model families clearly opposing (DRMM-PACRR-KNRM-VBERT) and CEDR since the former statistically exhibit higher REM metrics values than CEDR.

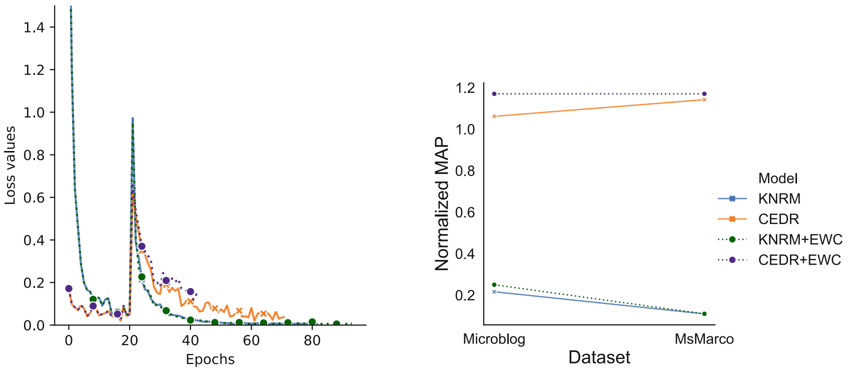
4.2 Mitigating Catastrophic Forgetting (RQ3)

From RQ1, we observed that some models are more prone to the catastrophic forgetting problem than others. Our objective here is to examine whether an EWC-based lifelong strategy can mitigate the problem. It is worth mentioning that this objective has been targeted in previous research in computer vision but without establishing a consensus [24,45,47]. While some studies reveal that EWC outperforms domain adaptation strategies in their settings [24,45], others found that it is less effective [47]. To achieve the experiment's objective, we particularly report the following measures in addition to the $REM/REMN$ measures: 1) $\Delta_{REM(REMN)}$ that reveals the improvement (positive or negative) of the $REM/REMN$ measures achieved using an EWC-based lifelong learning strategy over the $REM/REMN$ measure achieved using a fine-tuning strategy;

2) the PR measure introduced in Sect. 4.1. Unlikely, our aim through this measure here, is to highlight the performance stability of the learned model on the right dataset while avoiding catastrophic forgetting on the left dataset.

We turn now our attention to the **"EWC-based lifelong learning"** columns in Table 1. Our experiment results show that among the 9 (resp. 11) settings that exhibit catastrophic forgetting in the combined model (resp. neural model), EWC strategy allows to improve 9/9 i.e., 100% (resp. 9/11 i.e., 88%) of them in the range $[+0.3\%, +96.1\%]$ (resp.$[+3.3\%, +79.7\%]$). Interestingly, this improvement in performance on the left dataset does not come at the cost of a significant decrease in performance on the right dataset since 100% of the models achieve a PR ratio greater than 0.96. Given, in the one hand, the high variability of the settings derived from the samples, and in the other hand, the very low number of settings (10% i.e., 2/20) where a performance decrease is observed in the left dataset, we could argue that the EWC-based lifelong learning is not inherently impacted by dataset order leading to a general effectiveness trend over the models. We emphasize this general trend by particularly looking at the CEDR model which we recall (See Sect. 4.1, RQ1), clearly exhibits the catastrophic forgetting problem. As can be seen from Table 1, model performances on the left datasets are significantly improved ($+6.4\% \leq \Delta_{REM} \leq +96.1\%$; $0\% \leq \Delta_{REMN} \leq +8.7\%$) while keeping model performances on the right dataset stable ($0.961 \leq PR \leq 1.008$). This property is referred to as the stability-plasticity dilemma [41].

To get a better overview of the effect of the EWC strategy, we compare in Fig. 2 the behavior of the CEDR and KNRM models which exhibit respectively low level ($REM = 0.510$) and high level of remembering ($REM = 1$) particularly in the setting $ms \rightarrow mb$. The loss curves in Fig. 2(a) highlight a peak after the 20^{th} epoch for both CEDR and KNRM. This peak denotes the beginning of the fine-tuning on the mb dataset. After this peak, we can observe that the curve representing the EWC-based CEDR loss (in purple) is slightly above the CEDR



(a) Loss function **(b)** Performance on the mb dataset

Fig. 2. Impact of the EWC strategy on loss and performance for the $ms \rightarrow mb$ setting.

loss (in orange), while both curves for the KNRM model (green and blue resp. for with and without EWC) are overlayed. Combined with the statements outlined in RQ1 concerning the ability of the CEDR model to accumulate knowledge, this suggests that EWC is able to discriminate models prone to catastrophic forgetting and, when necessary, to relax the constraint of good ranking prediction on the dataset used for the fine-tuning to avoid over-fitting. This small degradation of knowledge acquisition during the fine-tuning on the *ms* dataset is carried out at the benefit of the previous knowledge retention to maintain retrieval performance on the *mb* dataset (Fig. 2(b)). Thus, we can infer that the EWC strategy applied on neural ranking models plays fully its role to mitigate the trade-off between stability and plasticity.

5 Conclusion

We investigated the problem of catastrophic forgetting in neural-network based ranking models. We carried out experiments using 5 SOTA models and 3 datasets showing that neural ranking effectiveness comes at the cost of forget and that transformer-based models allow a good balance between effectiveness and remembering. We also show that the EWC-based strategy mitigates the catastrophic forgetting problem while ensuring a good trade-off between transferability and plasticity. Besides, datasets providing weak and varying relevance signals are likely to be remembered. While previous work in the IR community mainly criticized neural models regarding effectiveness [35,39,51], we provide complementary insights on the relationship between effectiveness and transferability in a lifelong learning setting that involves cross-domain adaptation. We believe that our study, even under limited setups, provides fair and generalizable results that could serve future research and system-design in neural IR.

Acknowledgement. We would like to thank projects ANR COST (ANR-18-CE23-0016) and ANR JCJC SESAMS (ANR-18-CE23-0001) for supporting this work.

References

1. Adomavicius, G., Zhang, J.: Impact of data characteristics on recommender systems performance. ACM Trans. Manage. Inf. Syst. **3**(1), 1–17 (2012)
2. Asghar, N., Mou, L., Selby, K.A., Pantasdo, K.D., Poupart, P., Jiang, X.: Progressive memory banks for incremental domain adaptation. arXiv preprint arXiv:1811.00239 (2020)
3. Bajaj, P., et al.: Ms marco: a human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268 (2016)
4. Bengio, Y.: Deep learning of representations for unsupervised and transfer learning. In: UTLW2011, pp. 17–37 (2011)
5. Cai, H., Chen, H., Zhang, C., Song, Y., Zhao, X., Yin, D.: Adaptive parameterization for neural dialogue generation. In: EMNLP-IJCNLP, pp. 1793–1802 (2019)
6. Chen, Z., Liu, B.: Lifelong machine learning. Synth. Lect. Artif. Intell. Mach. Learn. **12**(3), 1–207 (2018)

7. Cohen, D., Mitra, B., Hofmann, K., Croft, W.B.: Cross domain regularization for neural ranking models using adversarial learning. In: ACM SIGIR, pp. 1025–1028 (2018)
8. Dehghani, M., Zamani, H., Severyn, A., Kamps, J., Croft, W.B.: Neural ranking models with weak supervision. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 65–74 (2017)
9. Deldjoo, Y., Di Noia, T., Di Sciascio, E., Merra, F.A.: How dataset characteristics affect the robustness of collaborative recommendation models. In: ACM SIGIR, pp. 951–960 (2020)
10. Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., Maltoni, D.: Don't forget, there is more than forgetting: new metrics for Continual Learning. arXiv preprint arXiv:1810.13166 (2018)
11. French, R.M.: Catastrophic forgetting in connectionist networks. Trends in Cogn. Sci. **3**(4), 128–135 (1999)
12. Goodfellow, I.J., Mirza, M., Da, X., Courville, A.C., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint arXiv:1312.6211 (2014)
13. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=lQdXeXDoWtI
14. Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep relevance matching model for ad-hoc retrieval. In: CIKM 2016, pp. 55–64. Association for Computing Machinery (2016)
15. Hancock, B., Bordes, A., Mazare, P.E., Weston, J.: Learning from dialogue after deployment: feed yourself, chatbot! In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3667–3684 (2019)
16. Hofstätter, S., Rekabsaz, N., Eickhoff, C., Hanbury, A.: On the effect of low-frequency terms on neural-ir models. In: SIGIR, pp. 1137–1140 (2019)
17. Hui, K., Yates, A., Berberich, K., de Melo, G.: PACRR: a position-aware neural IR model for relevance matching. arXiv preprint arXiv:1704.03940 (2017)
18. Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D.: Dataset shift in machine learning. The MIT Press (2009)
19. Jha, R., Lovering, C., Pavlick, E.: When does data augmentation help generalization in NLP? arXiv preprint arXiv:2004.15012 (2020)
20. Kemker, R., McClure, M., Abitino, A., Hayes, T.L., Kanan, C.: Measuring catastrophic forgetting in neural networks. In: AAAI-18, pp. 3390–3398 (2018)
21. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
22. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. Proc. Natl. Acad. Sci. **114**(13), 3521–3526 (2016)
23. Lee, C., Cho, K., Kang, W.: Mixout: effective regularization to finetune large-scale pretrained language models. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020. OpenReview.net (2020). https://openreview.net/forum?id=HkgaETNtDB
24. Lee, S.W., Kim, J.H., Jun, J., Ha, J.W., Zhang, B.T.: Overcoming catastrophic forgetting by incremental moment matching. In: NIPS2017, Curran Associates Inc., Red Hook, NY, USA, pp. 4655–4665 (2017)
25. Li, J., Miller, A.H., Chopra, S., Ranzato, M., Weston, J.: Learning through dialogue interactions by asking questions. In: ICLR 2017 (2017)
26. Li, Z., Hoiem, D.: Learning without forgetting. IEEE Trans. Pattern Anal. Mach. Intell. **12**, 2935–2947 (2018)

27. Lin, J., Efron, M.: Overview of the trec-2013 microblog track. In: Text REtrieval Conference (TREC), Gaithersburg, Maryland, USA (2013)
28. Liu, T.Y.: Learning to rank for information retrieval. Found. Trends Inf. Retr. **3**(3), 225–331 (2009)
29. Liu, X., Gao, J., He, X., Deng, L., Duh, K., Wang, Y.: Representation learning using multi-task deep neural networks for semantic classification and information retrieval. NAACL HLT **2015**, 912–921 (2015)
30. MacAvaney, S.: OpenNIR: a complete neural ad-hoc ranking pipeline. In: WSDM 2020 (2020)
31. MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: CEDR: contextualized embeddings for document ranking. In: ACM SIGIR, pp. 1101–1104 (2019)
32. d'Autume, C.D.M., Ruder, S., Kong, L., Yogatama, D.: Episodic memory in lifelong language learning. arXiv preprint arXiv:1906.01076 (2019)
33. Mazumder, S., Ma, N., Liu, B.: Towards a continuous knowledge learning engine for chatbots. arXiv preprint arXiv:1802.06024 (2018)
34. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR 2013 (2013)
35. Mitra, B., Craswell, N.: An introduction to neural information retrieval. Found. Trend Inf. Retrieval **13**(1), 1–126 (2018)
36. Mitra, B., Craswell, N.: An updated duet model for passage re-ranking. arXiv preprint arXiv:1903.07666 (2019)
37. Mosbach, M., Andriushchenko, M., Klakow, D.: On the stability of fine-tuning bert: misconceptions, explanations, and strong baselines. arXiv preprint arXiv:2006.04884 (2020)
38. Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., Jin, Z.: How transferable are neural networks in NLP applications? In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 479–489 (2016)
39. Onal, K.D., et al.: Neural information retrieval: at the end of the early years. Inf. Retrieval J. **21**, 111–182 (2017)
40. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. on Knowl. and Data Eng. 22(10), 1345–1359 (Oct 2010)
41. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: a review. Neural Netw. **113**, 54–71 (2019)
42. Roller, S., Boureau, Y.L., Weston, J., Bordes, A., Dinan, E., Fan, A., Gunning, D., Ju, D., Li, M., Poff, S., et al.: Open-domain conversational agents: Current progress, open problems, and future directions. arXiv preprint arXiv:2006.12442 (2020)
43. Rusu, A.A., et al.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)
44. Soboroff, I., Ounis, I., Macdonald, C., Lin, J.J.: Overview of the TREC-2012 microblog track. In: Proceedings of The Twenty-First Text REtrieval Conference, TREC 2012. NIST Special Publication, vol. 500 p. 298 (2012)
45. Thompson, B., Gwinnup, J., Khayrallah, H., Duh, K., Koehn, P.: Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In: NAACL, pp. 2062–2068 (2019)
46. Wang, L.L., et al.: Cord-19: the covid-19 open research dataset. ArXiv (2020)
47. Wen, S., Itti, L.: Overcoming catastrophic forgetting problem by weight consolidation and long-term memory. arXiv preprint arXiv:1805.07441 (2018)
48. Wiese, G., Weissenborn, D., Neves, M.: Neural domain adaptation for biomedical question answering. CoNLL **2017**, 281–289 (2017)

49. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling. In: ACM SIGIR, pp. 55–64 (2017)
50. Xu, H., Liu, B., Shu, L., Yu, P.S.: Lifelong domain word embedding via meta-learning. In: IJCAI-18, pp. 4510–4516 (2018)
51. Yang, W., Lu, K., Yang, P., Lin, J.: Critically examining the "neural hype" weak baselines and the additivity of effectiveness gains from neural ranking models. In: ACM SIGIR, pp. 1129–1132 (2019)
52. Yang, W., Xie, Y., Tan, L., Xiong, K., Li, M., Lin, J.: Data augmentation for BERT fine-tuning in open-domain question answering. arXiv preprint arXiv:1904.06652 (2019)
53. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: NIPS2014, pp. 3320–3328 (2014)

Extracting Search Tasks from Query Logs Using a Recurrent Deep Clustering Architecture

Luis Lugo(✉), Jose G. Moreno, and Gilles Hubert

IRIT UMR 5505 CNRS, U. de Toulouse, Toulouse, France
{luis.lugo,jose.moreno,gilles.hubert}@irit.fr

Abstract. Users fulfill their information needs by expressing them using search queries and running the queries in available search engines. The mining of query logs from search engines enables the automatic extraction of search tasks by clustering related queries into groups representing search tasks. The extraction of search tasks is crucial for multiple user supporting applications like query recommendation, query term prediction, and results ranking depending on search tasks. Most existing search task extraction methods use graph-based or nonparametric models, which grow as the query log size increases. Deep clustering methods offer a parametric alternative, but most deep clustering architectures fail to exploit recurrent neural networks for learning text data representations. We propose a recurrent deep clustering model for extracting search tasks from query logs. The proposed architecture leverages self-training and dual recurrent encoders for learning suitable latent representations of user queries, outperforming previous deep clustering methods. It is also a parametric approach that offers the possibility of having a fixed-sized architecture for analyzing increasingly large search query logs.

Keywords: Search task extraction · Deep clustering · Recurrent neural networks

1 Introduction

Users carry out their search tasks running groups of related queries in available search engines, fulfilling a wide range of information needs and desires [15]. Query logs record the queries that users submit to search engines. Therefore, proper extraction of search tasks from query logs helps to support users while they fulfill their information needs, facilitating multiple goals like query term prediction, query recommendation, advertisement, results ranking depending on the search task, query-task mapping, and prediction of user satisfaction based on search tasks [9,23,30,32].

Along with the search queries that users submit, query logs also contain timestamps and other user-related information. Initially, query logs were segmented using the time between query timestamps to establish a session boundary and delimiting search tasks. If the time between a pair of subsequent queries

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 391–404, 2021.
https://doi.org/10.1007/978-3-030-72113-8_26

was above a certain threshold in minutes, a boundary was established, signaling the end of a search task [18]. However, according to multiple analyses of search query logs, users tend to interleave search tasks in a single time session. Also, some tasks are performed during multiple time sessions [18,23,30]. Hence, clustering models have been utilized to extract search tasks by grouping semantically related queries.

Recent models for search task extraction rely on graph-based methods or nonparametric approaches [9,18,19,22,23,30], which grow as search query logs increase in size, making them more computationally expensive as the number of queries increases. By contrast, deep clustering methods [1,24] offer a parametric alternative to learn latent representations of query log entries and simultaneously cluster them into interrelated groups of search queries.

Most existing deep clustering approaches do not exploit the modeling power of recurrent neural networks (RNNs), which are widely used for natural language processing (NLP) and sequential data processing [12,17,25]. Therefore, we propose a recurrent deep clustering (RDC) model to extract search tasks from query logs. RDC leverages the modeling power of recurrent neural networks in a dual encoder configuration, along with self-training, to learn a suitable latent space of user queries and simultaneously cluster them in groups of search tasks.

2 Related Work

The need for large query log datasets that are cleaned and labeled by humans represents a challenge for supervised task extraction models [9,25,33]. As unsupervised learning approaches do not rely on labels [27], they could represent a better alternative for search task extraction. Clustering is an unsupervised learning approach that groups related items using abstract similarities or learns new categories by analogy to existing ones [25]. Clustering methods are essential in multiple data-driven applications. They are primarily based on partitioning, density, and hierarchies [1,24].

Several clustering methods based on graphs were proposed to extract search tasks from query logs, including QC-WCC and a faster variation named QC-HTC [18]. QC-WCC is a query clustering method based on weighted connected components. It relies on the construction of a graph where nodes correspond to queries, and weights in the edges depend on similarities between the queries. The similarities come from two features: a content-based feature using both Jaccard indexes on tri-grams and Levenshtein distances, and a semantic-based feature that exploits Wikipedia to infer query semantics. The graph is pruned by removing weak edges; then, the query clusters are obtained from the remaining subgraphs. QC-HTC is a variation of QC-WCC based on head-tail components, a faster clustering method because it avoids computing the full similarity graph.

QRY-VEC [30] is another graph-based method utilized for search task extraction from query logs. However, instead of relying on lexical similarities and documents retrieved from the Wikipedia collection [18], it uses a tempo lexical word vector representation and documents retrieved from the ClueWeb12B collection

[4,5]. MGBC [19] improves foregoing search task extraction methods by combining graph-based clustering with a latent multilingual space for query representation, using the angular distance to group related queries. Another nonparametric approach represents queries as a linear combination of vectors for its terms [22]. Weights in the linear combinations represent the maximum likelihood for each term according to the query task relationships. Based on that representation, the hierarchies of tasks are extracted from the query logs. The Chinese Restaurant Process provides the algorithm to compute task relatedness from the hierarchies. An improved variant of this method [23] relies on Bayesian Rose Trees (BRTs) [3] to model query logs as a hierarchical structure of search tasks. This improved variant uses a Bayesian nonparametric approach to compute the model that best represents the search task hierarchies in search logs.

Nonetheless, graph-based and nonparametric models grow as the size of datasets increases [27], becoming more computationally expensive. For example, the number of leaves in BRTs [23] is directly related to the number of queries in the search query log; for graph-based methods, every entry in the search query log ends up being a node in the underlying graph. Likewise, the representation of the data is crucial for the subsequent results of clustering methods. High dimensional data tends to affect clustering methods because distances in high dimensional spaces are less effective. Dimensionality reduction methods have been widely used, including linear methods, non-linear methods, and spectral methods. Nevertheless, the latent representation obtained from dimensionality reduction can affect clustering performance; thus, deep neural networks are a viable alternative to compute latent representations [1,24,27] for input data, without performing dimensionality reduction as a preprocessing step.

Deep neural networks can be used to simultaneously learn latent representations and cluster data, using a method commonly known as deep clustering [1,24]. Also, in contrast with graph-based methods and nonparametric approaches, deep clustering models do not grow with the size of the search query log [27]. Deep clustering appeared initially in acoustic separation and then spread to other areas of research [24]. Before deep clustering appeared, research focused on data representation and clustering methods independently. However, learning latent representations is at the heart of deep clustering.

Models in deep clustering rely on several neural network architectures, including autoencoders, variational autoencoders (VAEs), feedforward neural networks, convolutional neural networks, deep belief networks, and generative adversarial networks (GANs) [1,24]. All architectural variations are trained to learn cluster-friendly representations, combining representation learning and clustering. Deep neural networks are trained to minimize the clustering loss, optimizing the network weights for improving the predicted labels of input samples. Both GAN and VAE based architectures are generative. They do not only learn to cluster inputs; they are also able to generate samples from the clustering categories [1,24].

Existing deep clustering models fail to exploit RNNs for learning latent representations of text data samples. Text data naturally fits the sequential modeling power of recurrent neural networks [12]. Because of this, RNNs have been widely used for processing text data in NLP, generating state-of-the-art results in multiple applications [12,20,25,40]. Our proposed architecture differs from prior deep clustering methods [1,24] by using RNNs in a dual encoder configuration [38] to simultaneously learn latent representations of user queries and cluster them in groups of search tasks. Also, in contrast with other approaches [9,18,19,23,30], it provides a parametric model, which preserves its size despite the query log length.

3 Search Task Extraction

The proposed RDC model has an RNN encoder as the central component of its architecture. The architecture uses a dual encoder setup [38], a widely used configuration in representation learning, neural machine translation, and other NLP applications [7,38,39]. The recurrent encoder comprises a bidirectional recurrent layer, an attention mechanism, and a projection head [7,20]. Input queries comprise a list of word embeddings $q_i = [w_1, w_2, w_3, ...]$. To form the input query's latent representation, we concatenate the output of the attention mechanism and the hidden state of the bidirectional recurrent layer, passing the concatenated tensor through a projection head (Fig. 1). Regarding recurrent unit types for the encoder, we consider both Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) [8].

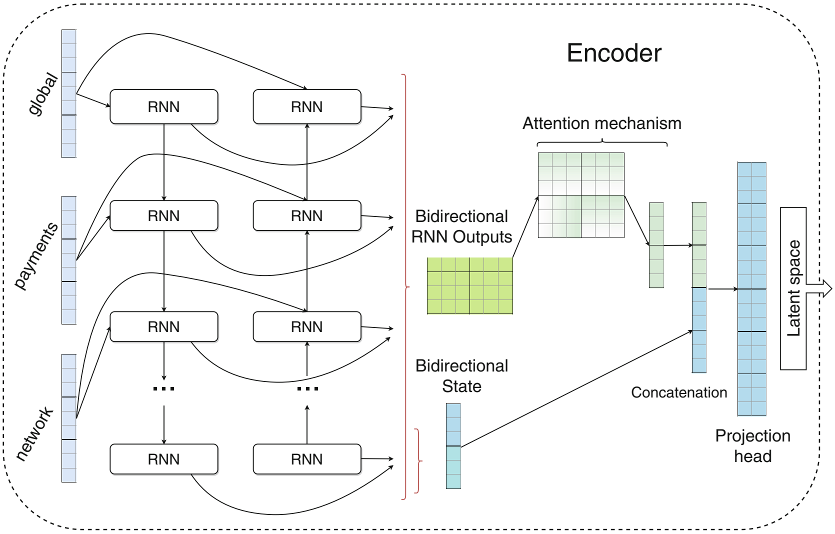


Fig. 1. Recurrent neural network encoder for learning latent representations of queries. The encoder comprises a bidirectional recurrent layer, an attention mechanism, and a projection head.

Two phases optimize RDC in tandem: a pretraining phase and a clustering phase [13,24,36]. Each phase has its loss; thus, we adapt the architecture of the model depending on the loss that we are optimizing (Fig. 2).

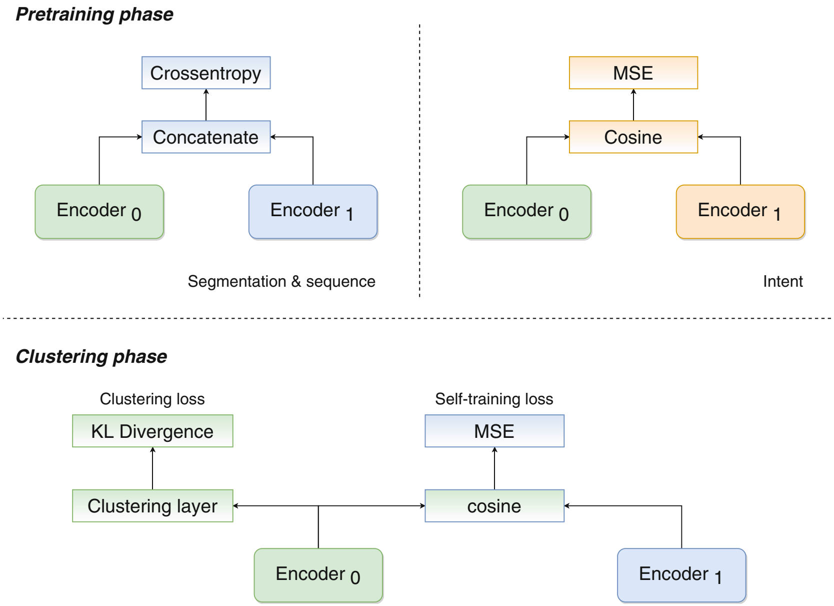


Fig. 2. Pretraining and clustering phases for search task extraction using a dual encoder configuration.

3.1 Pretraining Phase

Deep clustering methods tend to pretrain neural network layers before the clustering phase, which allows the initialization of the latent representation for input samples [1,24]. During the pretraining phase, we optimize the encoder with a supervised objective. We use the dual encoder configuration [38] to pretrain the recurrent encoders according to the following objectives:

- *Segmentation.* In this supervised pretraining objective, the recurrent encoders are trained using the search task segmentation approach [20]. This pretraining objective determines if two adjacent queries in a chronologically ordered query log are part of the same search task or not. The expected output of this objective is binary, and we use cross-entropy to compute the pretraining loss $\mathcal{L}_P$.
- *Sequence.* The sequence pretraining objective determines if a pair of queries appear adjacent in a chronologically ordered query log or not. Similar to the segmentation objective, the expected output of this objective is also binary, and we use cross-entropy to calculate the pretraining loss $\mathcal{L}_P$.

- *Intent.* For the intent pretraining objective, queries representing the user's intent for the same search task are close in the latent representation space [40]. Therefore, we compute the cosine proximity between encoder outputs and use the Mean Squared Error (MSE) [31] between predicted cosine proximity and expected cosine proximity to calculate the pretraining loss $\mathcal{L}_P$. The expected cosine proximity is set to one for pairs of queries pertaining to the same search task, zero otherwise.

3.2 Clustering Phase

Once the foregoing objectives have been used to pretrain the recurrent encoders, we discard the cross-entropy layer used during segmentation and sequence pretraining. The objective loss for the clustering phase $\mathcal{L}_O$ comprises two losses: the clustering loss $\mathcal{L}_C$ and the self-training loss $\mathcal{L}_S$ [13,21].

For the clustering loss, following previous work [1,24,36], we connect the pretrained encoder output to a clustering layer, where the Student's t-distribution provides a mean to compute the soft assignments for the queries. The soft assignment should match an auxiliary target distribution by using the Kullback–Leibler (KL) divergence to compute the clustering loss. Formally, given a query q_i and initial cluster centroids μ_j, the clustering loss $\mathcal{L}_C$ is calculated as follows [13,36]:

$$z_i = encoder_0(q_i) \tag{1}$$

$$s_{ij} = \frac{(1 + \|z_i - \mu_j\|^2)^{-1}}{\sum\limits_{j'}(1 + \|z_i - \mu'_j\|^2)^{-1}} \tag{2}$$

$$f_j = \sum_i s_{ij} \tag{3}$$

$$p_{ij} = \frac{s_{ij}^2/f_j}{\sum\limits_{j'} s_{ij'}^2/f_{j'}} \tag{4}$$

$$\mathcal{L}_C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{s_{ij}} \tag{5}$$

K-means generates the initial cluster centroids μ_j from the pretrained encoder representation. The Student's t-distribution in Eq. 2 has one degree of freedom. Also, the auxiliary target distribution p_{ij} emphasizes high confidence data point assignments, strengthens predictions, and normalizes the loss contribution from each cluster by using the soft cluster frequencies f_j [36].

For the self-training loss, we use the dual encoder configuration along with back translation [10], a self-training technique for unsupervised data augmentation [37] that preserves the semantics of the query encodings. Adding noise to query encodings can be ineffective for creating augmented samples because the resulting samples hardly match variations from real case scenarios. Hence, to create realistic augmented samples, strong data augmentation methods focus on creating modifications that march real variations. For instance, in computer vision, it is common to use cropping, rotation, or scaling to create augmented samples for images [7,41]. As queries are short texts, we use back translation to create realistic augmented samples for entries in the search query log. Back translation creates paraphrases for a search query while preserving the semantics of the original query [10]. Formally, given a query q_i, with augmented sampled b_i, and because the semantics from back translation remains the same, target cosine proximity $t_i = 1.0$, the self-training loss $\mathcal{L}_S$ is calculated as follows [13,24,37]:

$$u_i = encoder_0(q_i) \tag{6}$$

$$v_i = encoder_1(b_i) \tag{7}$$

$$p_i = \frac{u_i v_i}{|u_i|\,|v_i|} \tag{8}$$

$$\mathcal{L}_S = \sum_i (p_i - t_i)^2 \tag{9}$$

During the clustering phase, both the cluster centroids and the dual encoder weights are updated by optimizing the objective loss $\mathcal{L}_O$ [1,13,24]:

$$\mathcal{L}_O = \mathcal{L}_S + \gamma \mathcal{L}_C \tag{10}$$

where γ is a constant. The range of γ is $0.0 < \gamma < 1.0$ to help the model preserve the semantic space of the query encodings with the optimization of $\mathcal{L}_S$, while simultaneously optimizing $\mathcal{L}_C$ to improve the clustering performance.

4 Experimental Setup

Metrics to evaluate the performance of the models include the unsupervised accuracy (ACC), the Normalized Mutual Information (NMI), and the Adjusted Rand Index (ARI) [31]. The Student's paired t-test provides the test for statistical significance. To evaluate the effectiveness of the proposed approach, we compare RDC performance with the following methods:

- Deep Embedded Clustering (DEC) combines feature extraction with autoencoders and clustering. It learns clustering centers by first pretraining the autoencoder on the input dataset to learn a latent representation. Then, DEC discards the decoder part of the autoencoder and uses the encoder to calculate input representations. DEC uses K-means to initialize cluster centroids, then, it minimizes the clustering loss by minimizing the KL divergence (Eq. 5) [6,36].
- Improved Deep Embedded Clustering (IDEC) extends DEC by including the decoder part of the autoencoder during the clustering. Doing so aims to preserve the original structure of the input data in the latent representation space. To include the encoder, IDEC uses a loss to simultaneously optimize the clustering on the encoder output and the representational accuracy of the decoder output [13].

- Point Symmetry-based Deep Clustering (SymDEC) replaces the Euclidean distance that DEC uses for computing the clustering loss with the point symmetry-based distance, improving the results when clustering datasets with symmetrical input samples [26].
- Deep adaptive clustering (DAC) joins feature extraction and clustering into a single neural method. To extract features, DAC relies on a deep convolutional neural network and adds a constraint, so that resulting labels converge to a one-hot encoding. The constraint assumes that a pair of input samples either pertain to the same cluster or pertain to a different cluster. Input sample similarities are unknown beforehand; thus, an adaptive approach inspired by curriculum learning [2] is proposed. First, only pairs of input samples with similarities above or under a threshold are considered. With those pairs of images, the weights of the convolutional network are updated using backpropagation. As the training advances and the model improves, more pairs meet the threshold criteria. When the model converges, all pairs of input samples are part of the loss computation, and the loss stabilizes. Once the loss stabilizes, it selects the label with the highest value inside the one-hot vector to determine the cluster of the input sample [6].
- Chimera network [21,34] uses stacked layers of bidirectional LSTMs for audio separation models. This stacked recurrent model can handle problems like speaker-independent multi-speaker speech separation and music source separation. The Chimera architecture comprises four bidirectional layers, a dense layer to compute the vectors in the latent space, and two heads for multi-task learning: one head for unsupervised source separation and the other head for supervised time-frequency mask inference. We replace the multi-task learning heads with the clustering layer in Sect. 3.2 to adapt the architecture for search task extraction from query logs.

For reference, we also include results for k-means [31], Density-based spatial clustering of applications with noise (DBSCAN) [11], and Hierarchical Agglomerative Clustering (HAC) [28,35]. Scikit-learn[1] with default parameters provides the implementation for k-means, DBSCAN, and HAC. We also use publicly available implementations for DEC[2], IDEC[2], SymDEC[2], DAC[3], and Chimera[4] with the best performing hyperparameters reported for each method.

Two datasets are considered for evaluating RDC performance: the Cross-Session Task Extraction (CSTE) dataset [30], with 1424 user queries, and the Query-Task-Mapping based on TREC (QTMT) dataset, with 7771 user queries [32]. For pretraining, two datasets are considered as well: Sequence and Segmentation pretraining objectives use the Webis Search Mission Corpus 2012 (WSMC12) dataset [14], which has 8840 entries with 2881 search task labels. The WSMC12 dataset has timestamps so that we can guarantee a chronologically ordered query log. The Intent pretraining objective uses the

[1] https://scikit-learn.org/.

[2] https://github.com/XifengGuo/DEC-DA.

[3] https://github.com/HongtaoYang/DAC-tensorflow.

[4] https://github.com/leichtrhino/ChimeraNet.

Query-Task-Mapping based on WikiHow (QTMWH) dataset [32], which has 119292 queries with labels for 7202 search tasks.

The GloVe publicly available pre-trained word vectors[5] provide the representation for the search queries [29]. We use the same query representation for all the methods under testing. To train the RDC model, we use the Adam optimizer [16]. The learning rate is set to 10^{-4}, batch size to 256, and dropout to 0.3. The bidirectional layer contains 32 recurrent units, and the projection head has two feedforward layers, one with 512 units and the other with 256 units. Using the Google Cloud Translation API[6], we perform the back translation augmentation for the self-training loss (Eq. 9). Back translation is realized offline for practical purposes, using English (en) - French (fr) [37] to create the augmented samples.

5 Results and Discussion

Results for RDC with several pretraining configurations appear in Table 1. RDC outperforms all the other deep clustering methods used for comparison, for both the CSTE and the QTMT datasets. RDC also outperforms reference methods like k-means and DBSCAN. When comparing clustering performance against HAC, we find that RDC outperforms HAC when extracting short-lived search tasks, while in long search tasks, it improves over HAC in two out of three metrics ($p \leq 0.05$). The CSTE dataset has mostly short-lived search tasks because the average number of user queries per task is 3.2, while the QTMT dataset has an average of 28.2 user queries per task, reflecting behaviors like exploration, specification, or paraphrasing that users undertake in long search tasks [40]. These results are essential because short-lived search tasks, including fact-finding, browsing, or transactions, can account for up to 85% of all the entries in a search query log [15].

When comparing RDC with autoencoder-based models, such as DEC, IDEC, and SymDEC, the results are higher in all the metrics used for assessing clustering performance; we observe the same behavior when considering DAC, which uses convolutional neural networks. This outperformance reflects the advantage of using the modeling power of recurrent neural networks for learning representations of search queries. Chimera, a stacked recurrent architecture, also outperforms deep clustering models based on autoencoders and convolutional neural networks. However, RDC has a better clustering performance than Chimera in the three metrics used for comparison. Similarly, RDC has a more straightforward configuration than Chimera because RDC only uses two bidirectional recurrent layers for the dual encoder setup, while Chimera uses a stack of four bidirectional recurrent layers.

Self-training with back translation for queries renders pretraining effects negligible. Indeed, back translation using English (en) - French (fr) is a strong data augmentation technique. It augments data samples while preserving the semantics of the original queries. For instance, “effects of tide on columbia river”gets

[5] http://nlp.stanford.edu/data/glove.42B.300d.zip.
[6] https://cloud.google.com/translate.

Table 1. Clustering performance for CSTE and QTMT datasets, including RDC and other methods used for comparison. Differences in RDC results against all baseline methods have $p \leq 0.05$ for the Student's t-test.

Dataset	Method	Pretraining	ACC	NMI	ARI
CSTE	k-means	None	0.395	0.670	0.231
	DBSCAN	None	0.199	0.343	0.027
	HAC	None	0.407	0.719	0.310
	DEC	Autoencoder	0.362	0.684	0.345
	IDEC	Autoencoder	0.347	0.681	0.348
	SymDEC	Autoencoder	0.337	0.652	0.325
	DAC	None	0.318	0.644	0.344
	Chimera	None	0.387	0.707	0.339
	RDC	Sequence	**0.420**	**0.735**	**0.355**
	RDC	Segmentation	0.408	0.730	0.354
	RDC	Intent	0.331	0.641	0.334
	RDC	None	0.415	0.734	0.355
QTMT	k-means	None	0.219	0.535	0.050
	DBSCAN	None	0.026	0.105	0.001
	HAC	None	0.276	**0.613**	0.086
	DEC	Autoencoder	0.097	0.419	0.019
	IDEC	Autoencoder	0.097	0.418	0.018
	SymDEC	Autoencoder	0.104	0.396	0.022
	DAC	None	0.095	0.368	0.025
	Chimera	None	0.214	0.523	0.061
	RDC	Sequence	**0.285**	0.594	0.094
	RDC	Segmentation	0.246	0.566	0.080
	RDC	Intent	0.187	0.508	0.055
	RDC	None	0.284	0.590	**0.095**

translated to “effects de la marée sur la rivière Columbia”, and then back translated to “tidal effects on the columbia river”; “farm houses for rent in broom county”gets translated to “Maisons de ferme à louer dans le comté de broome”, and then back translated to “farms for lease in broom county”. Sometimes back translation corrects spelling, for instance “the cost of haveing a horse in new york”gets translated to “le coût d’avoir un cheval à new york”, and then back translated to “the cost of having a horse in new york”, but in general, the semantics remain the same, so the query encoding space is preserved during the clustering phase by minimizing the self-training loss.

Consequently, although the best pretraining scheme for the RDC models is the Sequence objective, surpassing the results of both Segmentation and Intent objectives, it represents no change when compared against RDC with no pretraining. For the CSTE dataset, accuracy is only 0.5% higher, and NMI is only 0.1% higher ($p = 0.8$); ARI has no change at all. We observe a similar behavior with the QTMT dataset. In some cases, pretraining can even end up hurting performance, as we can see with the Intent pretraining objective. Preceding results are in agreement with previous work about the effect of pretraining neural architectures [41], where self-training with strong data augmentation diminishes the effect of pretraining, making it negligible. Therefore, it is possible to discard neural network pretraining, an essential result because pretraining needs labeled datasets, which can be challenging to create [33], while self-training with back translation is unsupervised.

Regarding recurrent units, the decision of which to choose depends on the task and the dataset [8]; therefore, we analyze the RDC model with both GRU and LSTM cells (Table 2). Replacing the LSTM cells with GRUs generates a slight decrease in model performance for CSTE and QTMT datasets. The biggest difference happens with the QTMT dataset, using intent pretraining, where changing GRUs to LSTMs makes accuracy decrease 2.7%, NMI 2.5%, and ARI 1.1% (p $\leq$ 0.05). These changes imply that less computationally expensive GRUs are a better choice for the RDC architecture than LSTMs, although changes observed for the metrics are low, especially with the sequence or no pretraining configurations, which are the best performing setups for the RDC model.

Table 2. Comparison between LSTM and GRU cells in RDC. Results include all the pretraining alternatives for the CSTE and QTMT datasets. Differences between recurrent cell results have $p \leq 0.05$ for the Student's t-test.

Dataset	Cell	Pretraining	ACC	NMI	ARI
CSTE	LSTM	Sequence	0.410	0.730	0.354
		Segmentation	0.399	0.718	0.352
		Intent	0.320	0.632	0.334
		None	0.409	0.729	0.354
	GRU	Sequence	0.420	0.735	0.355
		Segmentation	0.408	0.730	0.354
		Intent	0.331	0.641	0.334
		None	0.415	0.734	0.355
QTMT	LSTM	Sequence	0.278	0.592	0.096
		Segmentation	0.217	0.544	0.070
		Intent	0.160	0.483	0.044
		None	0.268	0.586	0.092
	GRU	Sequence	0.285	0.594	0.094
		Segmentation	0.246	0.566	0.080
		Intent	0.187	0.508	0.055
		None	0.284	0.590	0.095

6 Conclusion

This paper presented RDC, a recurrent deep clustering method for extracting search tasks from query logs. The proposed method leverages self-training and dual recurrent encoders to find latent representations for user queries, clustering them into search task groups. Experimental results show the proposed clustering method outperforms prior deep embedding clustering architectures in all the metrics used for testing. Also, RDC offers a parametric architecture for search task extraction, which preserves its size despite changes in the query log size. This size preservation represents an advantage compared to nonparametric methods and graph-based models that grow with the query log size, making them more computationally expensive as the number of queries in the search log grows. In future work, we will compare the RDC model to generative architectures for deep clustering, including models based on GANs and VAEs. We also plan to replace query representations based on GloVe with transformer-based query encodings.

Acknowledgements. This work was supported by the Agence National de la Recherche (ANR), through project CoST, code ANR-18-CE23-0016.

References

1. Aljalbout, E., Golkov, V., Siddiqui, Y., Strobel, M., Cremers, D.: Clustering with deep learning: taxonomy and new methods. arXiv preprint arXiv:1801.07648 (2018)
2. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 41–48 (2009)
3. Blundell, C., Teh, Y.W., Heller, K.A.: Bayesian rose trees. In: Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence, UAI 2010, pp. 65–72. AUAI Press, Arlington, Virginia, United States (2010)
4. Callan, J.: The Lemur project and its ClueWeb12B dataset. In: Invited talk at the SIGIR 2012 Workshop on Open-Source Information Retrieval (2012)
5. Carterette, B., Clough, P., Hall, M., Kanoulas, E., Sanderson, M.: Evaluating retrieval over sessions: the trec session track 2011–2014. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 685–688. ACM (2016)
6. Chang, J., Wang, L., Meng, G., Xiang, S., Pan, C.: Deep adaptive image clustering. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 5880–5888. IEEE (2017)
7. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. arXiv preprint arXiv:2002.05709 (2020)
8. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
9. Du, C., Shu, P., Li, Y.: CA-LSTM: search task identification with context attention based LSTM. In: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1101–1104. ACM (2018)
10. Edunov, S., Ott, M., Auli, M., Grangier, D.: Understanding back-translation at scale. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 489–500 (2018)

11. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: The 2nd International Conference on Knowledge Discovery and Data Mining vol. 96, pp. 226–231 (1996)
12. Graves, A.: Supervised Sequence Labelling with Recurrent Neural Networks. Springer, Heidelberg (2012)
13. Guo, X., Gao, L., Liu, X., Yin, J.: Improved deep embedded clustering with local structure preservation. In: International Joint Conference on Artificial Intelligence, pp. 1753–1759 (2017)
14. Hagen, M., Gomoll, J., Beyer, A., Stein, B.: From search session detection to search mission detection. In: Proceedings of the 10th Conference on Open Research Areas in Information Retrieval, pp. 85–92 (2013)
15. Hearst, M.: Search User Interfaces. Cambridge University Press, Cambridge, CB2 8BS, UK (2009)
16. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
17. Lipton, Z.C., Berkowitz, J., Elkan, C.: A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019 (2015)
18. Lucchese, C., Orlando, S., Perego, R., Silvestri, F., Tolomei, G.: Identifying task-based sessions in search engine query logs. In: Proceedings of the 4th ACM International Conference on Web Search and Data mining, pp. 277–286. ACM (2011)
19. Lugo, L., Moreno, J.G., Hubert, G.: A multilingual approach for unsupervised search task identification. In: The 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2041–2044. ACM (2020)
20. Lugo, L., Moreno, J.G., Hubert, G.: Segmenting search query logs by learning to detect search task boundaries. In: The 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2037–2040. ACM (2020)
21. Luo, Y., Chen, Z., Hershey, J.R., Le Roux, J., Mesgarani, N.: Deep clustering and conventional networks for music separation: stronger together. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 61–65. IEEE (2017)
22. Mehrotra, R., Bhattacharya, P., Yilmaz, E.: Deconstructing complex search tasks: a Bayesian nonparametric approach for extracting sub-tasks. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 599–605 (2016)
23. Mehrotra, R., Yilmaz, E.: Extracting hierarchies of search tasks & subtasks via a Bayesian nonparametric approach. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 285–294. ACM (2017)
24. Min, E., Guo, X., Liu, Q., Zhang, G., Cui, J., Long, J.: A survey of clustering with deep learning: From the perspective of network architecture. IEEE Access **6**, 39501–39514 (2018)
25. Mitchell, M.: Artificial Intelligence: A Guide for Thinking Humans. Farrar, Straus and Giroux, New York, NY, US (2019)
26. Moreno, J.G.: Point symmetry-based deep clustering. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 1747–1750. ACM (2018)
27. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. The MIT Press, Cambridge (2012)
28. Murtagh, F., Legendre, P.: Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion? J. Classif. **31**(3), 274–295 (2014)

29. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
30. Sen, P., Ganguly, D., Jones, G.: Tempo-lexical context driven word embedding for cross-session search task extraction. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 283–292 (2018)
31. Tan, P.N., Steinbach, M., Karpatne, A., Kumar, V.: Introduction to Data Mining, 2nd edn. Pearson Education, London (2018)
32. Völske, M., Fatehifar, E., Stein, B., Hagen, M.: Query-task mapping. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 969–972 (2019)
33. Wang, H., Song, Y., Chang, M.W., He, X., White, R.W., Chu, W.: Learning to extract cross-session search tasks. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1353–1364. ACM (2013)
34. Wang, Z.Q., Le Roux, J., Hershey, J.R.: Alternative objective functions for deep clustering. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 686–690. IEEE (2018)
35. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. J. Am. Stat. Assoc. **58**(301), 236–244 (1963)
36. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: International Conference on Machine Learning, pp. 478–487 (2016)
37. Xie, Q., Dai, Z., Hovy, E., Luong, M.T., Le, Q.V.: Unsupervised data augmentation for consistency training. arXiv preprint arXiv:1904.12848 (2019)
38. Yang, Y., et al.: Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 5370–5378. AAAI Press (2019)
39. Yang, Y., et al.: Multilingual universal sentence encoder for semantic retrieval. In: Proceedings of the 58th Annual Meeting of the ACL: System Demonstrations, pp. 87–94. ACL (2020)
40. Zhang, H., et al.: Generic intent representation in web search. In: The 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2019)
41. Zoph, B., et al.: Rethinking pre-training and self-training. arXiv preprint arXiv:2006.06882 (2020)

Modeling User Search Tasks with a Language-Agnostic Unsupervised Approach

Luis Lugo(✉), Jose G. Moreno, and Gilles Hubert

IRIT UMR 5505, CNRS,U. de Toulouse, Toulouse, France
{luis.lugo,jose.moreno,gilles.hubert}@irit.fr

Abstract. Conversational information seeking is a major emerging research area because of the increasing popularity of conversational AI systems users utilize to perform their search tasks. Search systems and multiple other user supporting applications benefit from modeling the search tasks users carry out to satisfy their information needs. Most existing search task modeling methods are monolingual, and few methods leverage user clicks even though clicked URLs are crucial for modeling user intent. We propose a language-agnostic, user intent aware approach to model search tasks from user interactions with search systems. The proposed approach leverages user intent modeling from clicked query-document pairs, latent representations of queries in a language-agnostic space, and graph-based clusteringto model search tasks in an unsupervised approach. Experimental results demonstrate the proposed approach outperforms recent work in search task modeling, supporting user queries in multiple languages. It can also produce search task modeling results in the order of milliseconds, an essential aspect for conversational systems and user support applications requiring realtime results.

Keywords: Conversational search · User intent modeling · Language-agnostic query representation

1 Introduction

Conversational AI systems are becoming increasingly popular because of advances in speech recognition, natural language understanding, text-to-speech synthesis, and the availability of digital personal assistants [15,24,26]. Personal assistants like Amazon Alexa, Apple Siri, Google Assistant, and Microsoft Cortana are now available in smartphones, tablets, desktops, and dedicated smart speakers [15,30]. Consequently, the increasing popularity and availability of conversational systems make conversational information seeking a major emerging area of research [1,30].

In conversational information seeking and other search systems, modeling the search tasks that users perform to satisfy their information needs is a crucial step [18,22]. Search task modeling is a step in the process to make search

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 405–418, 2021.
https://doi.org/10.1007/978-3-030-72113-8_27

systems more coherent, natural, engaging, and conversational [15,21,26]. Multiple user supporting applications benefit from search task modeling, including conversational question suggestion, personalization in e-commerce, product recommendations, query term prediction, query suggestions, query reformulation, and results ranking [13,20,21,23,27]. Even informative conversations with digital assistants can benefit from correctly modeling the search tasks, as the subjective perception of the quality in the conversation is strongly related to the accurate tracking of the topic [26].

Users around the world access search systems in multiple languages, making it essential to process users' requests with language-agnostic models. Also, search systems and user supporting applications require realtime responses when processing user information needs. For instance, multimodal search in conversational systems runs multiple processes in parallel, post-processing their outputs to generate a message answering the user request; hence, modeling can not exceed the timeout periods set on the search system [30]. Similarly, user clicks are strongly related to the user intent [31]. Different queries with similar clicked URLs can pertain to the same information need [20], and analyzing clicked URLs can help disambiguate queries [5].

Our contributions are threefold. First, we propose a language-agnostic search task modeling (LASTM) approach to model search tasks from user interactions with search systems. Second, given the relationship between clicked URLs and user intent, we propose a user intent modeling technique leveraging a large scale query - clicked document collection in the query latent space. Third, to enable the utilization of LASTM in conversational search systems and user supporting applications requiring responses on the fly, we propose a realtime method for mapping incoming queries to the modeled user search tasks directly on the query latent space.

2 Related Work

Mining user interactions with search systems enable modeling the search tasks that users perform to satisfy their information needs [13]. In particular, search query logs can be mined for search task modeling using methods such as heuristics-based models, semi-supervised clustering, Bayesian approaches, and graph-based clustering. A model based on a cascade of heuristics [12] first segments the search query log in logical sessions; then, it performs a post-processing step to detect search tasks based on the queries pertaining to the logical sessions. However, several manually set thresholds in the heuristics make it challenging to adapt heuristics-based models to other datasets without manually adjusting them.

Semi-supervised clustering approaches combine a supervised component and an unsupervised method to model search tasks. Bestlink SVM [28] first trains a support vector machine to detect if a pair of adjacent queries from a user pertains to the same search tasks or not. Then, it clusters the related queries in the search log using the SVM output to establish links between queries.

Bestlink SVM uses a backward context from users' queries to improve the task clustering results. The two most important features from the query representation in Bestlink SVM includes the cosine similarity between query embeddings and the similarity between clicked URLs. Context Attention based LSTM (CA-LSTM) [9] relies on recurrent neural networks instead of SVMs, using both backward and forward queries to provide context while training the neural network to detect if a pair of adjacent queries pertain to the same task or not. CA-LSTM then uses graphs to cluster related queries.

Bayesian approaches include Latent Dirichlet Allocation with Hawkes processes (LDA-Hawkes) [16], Distance Dependent Chinese Restaurant Process (DD-CRP) [19], and Bayesian Rose Trees (BRTs) [20]. LDA-Hawkes combines LDA with Hawkes processes to identify and label search tasks from query logs. LDA performs topic modeling, identifying semantically related queries from different users, while Hawkes processes take into account time lapses between query timestamps in individual query sequences, assigning temporally close queries to the same search task. DD-CRP extracts a single-level hierarchy of tasks from query logs, linking related queries by word embedding distances. DD-CRP assumes a restaurant with an infinite number of tables. Customers enter the restaurant in tandem; they are assigned to a nonempty table based on the number of existing customers in the table, or an empty table depending on a hyper-parameter. Entries in the query log are customers, while search tasks are tables. BRTs extend the single level hierarchy of DD-CRP to multiple levels, modeling search tasks by clustering related nodes in the hierarchical structure.

Graph-based clustering is used in several approaches for search task modeling [17,18,22]. QC-WCC [17] builds a graph where nodes correspond to queries, and edges are weighted according to the similarities between queries. Similarities are based on two features: one content-based from Jaccard similarity on tri-grams, and the other semantic-based exploiting Wikipedia and Wiktionary to infer the semantics. QC-HTC [17] is a computationally simpler algorithm based on QC-WCC, although less accurate. It exploits the sequential nature of queries to decrease the computational complexity of the graph based method. QC-HTC first builds sequences of queries according to the distance between them, creating the first set of clusters, then takes the first and last queries of the sequence to represent a cluster and group it with other clusters depending on query distances. Using only the first and last queries in each cluster avoids the computation of the full similarity graph required for QC-WCC, making QC-HTC less computationally expensive.

QRY-VEC [22] improves over the QC-WCC algorithm using word embedding similarities instead of lexically based similarities. Queries for the same task clusters tend to be semantically similar rather than lexically similar, as queries in the same tasks contain more synonym words than exact words [17,27]. Because of this, instead of relying on lexically based similarities and retrieved documents from the Wikipedia collection, QRY-VEC uses the cosine similarity on tempo-lexical word embeddings and documents retrieved from the ClueWeb12B collection [3]. Multilingual Graph-Based Clustering (MGBC) [18] outperforms previous models by combining a multilingual query encoding with graph-based clustering, supporting queries in several languages through the use of the Multilingual Universal Sentence Encoder (MUSE) [29]

However, most search task modeling methods [9,12,16,17,19,20,22,28] are monolingual. Although MGBC supports several languages through MUSE, it can only process queries in sixteen languages. Additionally, when using ClueWeb12B for calculating query similarities, MGBC can only support user queries in English. By the same token, most search task modeling methods [9,16–19,22] fail to take into account clicked URLs when processing search query logs, even though clicked URLs have a critical correlation to the user intent [31]. Also, conversational information seeking systems and multiple applications supporting users search efforts require results on the fly. Building models from scratch when a user submits a query could create large processing times, forcing search systems to trigger timeout intervals [30]. Similarly, waiting for forward queries to provide context [9] can render models unfeasible in realtime setups. Also, some models requiring user identifiers [9,12,16,20] can not be used in user-independent [5,18,22] modeling scenarios.

3 User Search Task Modeling

LASTM is an unsupervised method that leverages latent representations of queries in a language-agnostic space, user intent modeling from clicked query-document pairs, and graph-based clustering to model user search tasks. It can also produce a realtime mapping of queries to modeled search tasks. In contrast with previous work [9,12,16,17,19,20,22,28], our proposed approach supports multiple languages through a language-agnostic latent space. The proposed approach is also independent of user identifiers, enabling the modeling of search tasks in both user-independent and personalized scenarios. It also differs from some prior methods [9,16–19,22] by leveraging clicked URLs to model user intent [31] in the query latent space.

3.1 Language-Agnostic Query Representation

Users worldwide submit queries in different languages to satisfy their information needs. Language-agnostic BERT Sentence Embedding (LABSE) [10] provides the sentence embeddings to represent user queries in a language-agnostic latent space. Using a 12-layer transformer architecture [7,25] in a dual configuration, LABSE takes the transformer's hidden state for the last token in the sentence to generate the query representation.

The query representation using LABSE has the ability to perform zero-shot cross-lingual transfer, supporting queries in languages that are not part of the training dataset. When performing tests with the TAOEBA dataset [2], LABSE obtains an 83.7% accuracy, while the baseline Language-agnostic Sentence Representations [2] gets 65.5%, even though more than 30 languages in the TAOEBA dataset were not part of the LABSE training data [10].

We use the cosine proximity [10,22] to compute the similarity between query representations in the language-agnostic latent space. Formally, given a pair of queries q_i, q_j with latent representations u_i, u_j, the similarity between query representations S_{lat} is calculated as follows [10,22]:

$$S_{lat}(u_i, u_j) = \frac{u_i u_j}{|u_i|\,|u_j|} \tag{1}$$

3.2 User Intent Modeling

User clicks play a critical role in modeling user intent – the information need the user wants to satisfy by performing the search task [31]. Query term match between queries for the same information need can be very low; even lexically different queries pertaining to the same search task can have similar clicked URLs [20,31]. Also, analysis of clicked URLs can help disambiguate queries, revealing which documents users clicked when performing their search tasks [5].

To model user intent, we use the Open Resource for Click Analysis in Search (ORCAS) [5], a collection containing 18.8 million clicked document - query pairs for 10.4 million unique queries. Clicked documents are represented using the TREC document identifier in the TREC Deep Learning document collection [6]. We encode queries in ORCAS in the language-agnostic latent space [10], creating a user intent database $\mathcal{D}_M$ with clicked document - query pairs. To retrieve the most relevant documents for a given user query in the database, we use Scalable Nearest Neighbor (ScaNN) [11], a state-of-the-art method for large-scale retrieval tasks. ScaNN performs maximum inner product search (MIPS) using an anisotropic vector quantization, which allows a fast rate of document scoring.

Even though ORCAS has queries exclusively in English, doing MIPS directly on the language-agnostic latent space enables user intent modeling in any language LABSE can support. Hence, we can leverage the existing relationship between clicked URLs and user intent [31] by searching the $\mathcal{D}_M$ database.

Formally, given a database $\mathcal{D}_M = \{m_i\}_{i=1,2,...,n}$ formed from a clicked query-document dataset $\mathcal{D}_Q$ with n data points, where each data point $m_i \in \mathbb{R}^p$ is the latent representation of the query $q \in \mathcal{D}_Q$ in the p-dimensional language-agnostic latent space, we want to find the most relevant documents $\{d_j\}_{j=1,2,...,k} \in \mathcal{D}_M$ for the user query $u \in \mathbb{R}^p$. Therefore, we search for the k points with the maximum inner product with the user query u as follows [11]:

$$MIPS(\mathcal{D}_M, u) = \{d_j\}_{j=1,2,...,k} = \arg \max_{m_i \in \mathcal{D}_M} \langle u, m_i \rangle \tag{2}$$

Given a user query pair q_i, q_j with latent representations u_i, u_j, the similarity based on user intent S_{int} is calculated using the Jaccard coefficient for the top thousand relevant documents in the database $\mathcal{D}_M$ [18,22]:

$$D_i = MIPS(\mathcal{D}_M, u_i) \tag{3}$$

$$D_j = MIPS(\mathcal{D}_M, u_j) \tag{4}$$

$$S_{int}(u_i, u_j) = \frac{|D_i \cap D_j|}{|D_i \cup D_j|} \tag{5}$$

3.3 Unsupervised Search Task Modeling

We now integrate user intent modeling and language-agnostic query representations with graph-based clustering [4] to model search tasks (Algorithm 1). First, we encode queries in the latent space (Section 3.1); every query embedding becomes a node in the weighted graph. Then, we compute the similarities between pairs of queries to create the edges of the weighted graph. The similarity between queries S_{qry} is a convex combination of the similarity in the latent space S_{lat} and the similarity based on user intent S_{int}. Given a pair of queries q_i, q_j with latent representations u_i, u_j, query similarity S_{qry} is calculated as follows [18]:

$$S_{qry}(u_i, u_j) = \alpha * S_{lat}(u_i, u_j) + (1 - \alpha) * S_{int}(u_i, u_j) \tag{6}$$

After finishing edge weight calculations, we prune the weighted graph, deleting edges with $S_{qry} < \eta$. The resulting connected components $\mathcal{C}$ in the graph constitute the search tasks, so we assign a unique task label $task_i$ to every connected component. All the queries pertaining to a connected component receive the same task label. A grid search optimizes parameters η and α, using $\eta = k/10, \alpha = k/10, 0 < k \leq 10, k \in \mathbb{N}$ [4,17,18,22].

3.4 Realtime Mapping of New Queries

Most search systems and user supporting applications require results in realtime. Applications like contextual topic modeling in conversational search [15], query suggestion, or query reformulation can not afford to wait for large processing times. It is essential to return an answer in a few milliseconds. Hence, once the user performs a search request, we map the new incoming query to the labels extracted with Algorithm 1 so that we can model the search task in realtime. To do the mapping, we use the same MIPS method with anisotropic vector quantization [11] that we used in Section 3.2.

The search task database maps the latent representation of the queries in the search log $\mathcal{Q}_L$ to the extracted task labels $\mathcal{L}_T$. Formally, given a database $\mathcal{Q}_T = \{m_i\}_{i=1,2,\ldots,n}$ formed from the search query log $\mathcal{Q}_L$ with search task labels $\mathcal{L}_T$ returned from Algorithm 1, where each datapoint $m_i \in \mathbb{R}^p$ is the latent representation of the query $q \in \mathcal{Q}_L$ in the p-dimensional language-agnostic space. For an incoming query q_i, we compute the latent representation u_i; then, we retrieve the search task labels T of the k closest queries in the language-agnostic latent space using MIPS:

$$T = MIPS(\mathcal{Q}_T, u_i) \tag{7}$$

Once we have the search task labels T of the k closest queries, we return the task label with the highest number of occurrences in T.

Algorithm 1. LASTM

```
Inputs: Search query log $\mathcal{Q}_L$, Clicked query-document collection $\mathcal{D}_Q$
Output: Task labels $\mathcal{L}_T$

  // Build database for user intent
  $\mathcal{D}_M \leftarrow \{\}$
  for all $q_i, d_i \in \mathcal{D}_Q$ do
      $x_i \leftarrow language\_agnostic\_space(q_i)$
      $\mathcal{D}_M \leftarrow \mathcal{D}_M \cup \{x_i, d_i\}$
  end for

  // Model search tasks
  $V \leftarrow \{\}, E \leftarrow \{\}, G(V, E) \leftarrow (V, E)$
  for all $q_i \in \mathcal{Q}_L$ do
      $u_i \leftarrow language\_agnostic\_space(q_i)$
      $V \leftarrow V \cup \{u_i\}$
  end for

  for all $v_i, v_j \in V$ do
      $S_{lat}(v_i, v_j) = cos(v_i, v_j)$
      $D_i, D_j \leftarrow$ document IDs for $v_i, v_j$ from $\mathcal{D}_M$
      $S_{int}(v_i, v_j) = Jaccard(D_i, D_j)$
      $\mathbf{e}_k \leftarrow \alpha * S_{lat}(v_i, v_j) + (1 - \alpha) * S_{int}(v_i, v_j)$
      $E \leftarrow E \cup \{\mathbf{e}_k\}$
  end for

  for all $\mathbf{e}_k \in E$ do
      if $\mathbf{e}_k < \eta$ then
          $E \leftarrow E \setminus \{\mathbf{e}_k\}$
      end if
  end for

  for all $\mathcal{C}_i \in G(V, E)$ do
      $task_i \leftarrow i$
      for all $v_k \in \mathcal{C}_i$ do
          $\mathcal{L}_T\,[v_k] \leftarrow task_i$
      end for
  end for

  return $\mathcal{L}_T$
```

4 Results and Discussion

In this section, we analyze LASTM in user independent search task modeling and realtime mapping of incoming queries. Following previous work [9,18], we calculate model performance with the F_β score:

$$F_\beta = \frac{(1+\beta^2)*p*r}{\beta^2*p+r} \tag{8}$$

where p is precision and r is recall. We consider both $\beta = 1.0$ and $\beta = 0.6$ [9], which gives more weight to the precision of the model. The Student's paired t-test provides statistical significance calculations [31].

We use open source implementations for ScaNN[1], NetworkX[2] in graph-based clustering, and the publicly available pretrained model for LABSE.[3]

Table 1. Search task modeling results for the CSTE dataset in all the languages supported by the MGBC method. Differences between MGBC and LASTM results have $p \leq 0.05$ for the Student's t-test.

Language	ISO 639-1	F_1		$F_{0.6}$	
		MGBC	LASTM	MGBC	LASTM
Arabic	ar	0.447	**0.521**	0.395	**0.490**
Chinese PRC	zh	0.480	**0.539**	0.473	**0.513**
Chinese Taiwan	zh-tw	0.482	**0.540**	0.476	**0.515**
Dutch	nl	0.449	**0.534**	0.431	**0.511**
English	en	0.456	**0.538**	0.437	**0.512**
German	de	0.450	**0.533**	0.432	**0.511**
French	fr	0.484	**0.539**	**0.547**	0.512
Italian	it	0.452	**0.540**	0.434	**0.517**
Portuguese	pt	0.458	**0.537**	0.438	**0.514**
Spanish	es	0.450	**0.541**	0.432	**0.516**
Japanese	ja	0.453	**0.522**	0.436	**0.495**
Korean	ko	0.451	**0.523**	0.396	**0.501**
Russian	ru	0.449	**0.533**	0.429	**0.508**
Polish	pl	0.460	**0.536**	**0.524**	0.512
Thai	th	0.444	**0.522**	0.427	**0.489**
Turkish	tr	0.429	**0.538**	0.401	**0.513**

4.1 Search Task Modeling

The Cross-Session Task Extraction (CSTE) dataset [22] and the Complex User Search Task Analysis (CUSTA) dataset [8] are used for experiments. CSTE has 1424 entries with 224 ground truth labels corresponding to cross-session search tasks. CUSTA has 2390 entries with 15 ground truth search task labels. As a

[1] https://github.com/google-research/google-research/tree/master/scann.
[2] https://networkx.github.io/.
[3] https://tfhub.dev/google/LaBSE/1.

baseline, we use MGBC, a state-of-the-art method for search task modeling, calculating metrics for all the languages supported by the baseline. Queries in the CSTE dataset are in English, while queries in the CUSTA dataset are mostly in French, with very few English entries. Hence, we perform machine translation with the Google Cloud Translation API[4] for evaluating LASTM in all the languages supported by MGBC.

Table 2. Search task modeling results for the CUSTA dataset in all the languages supported by the MGBC method. Differences between MGBC and LASTM results have $p \leq 0.05$ for the Student's t-test.

Language	ISO 639-1	F_1		$F_{0.6}$	
		MGBC	LASTM	MGBC	LASTM
Arabic	ar	0.595	**0.608**	0.648	**0.665**
Chinese PRC	zh	0.658	**0.667**	0.667	**0.688**
Chinese Taiwan	zh-tw	0.632	**0.672**	0.604	**0.694**
Dutch	nl	0.594	**0.648**	0.577	**0.761**
English	en	0.597	**0.657**	0.544	**0.705**
German	de	0.550	**0.642**	0.542	**0.715**
French	fr	0.656	**0.732**	0.748	**0.750**
Italian	it	0.559	**0.604**	0.492	**0.602**
Portuguese	pt	0.616	**0.622**	0.610	**0.636**
Spanish	es	0.641	**0.643**	0.593	**0.712**
Japanese	ja	**0.697**	0.619	**0.737**	0.571
Korean	ko	**0.573**	0.563	**0.639**	0.561
Russian	ru	0.633	**0.641**	0.742	**0.754**
Polish	pl	0.541	**0.598**	0.578	**0.605**
Thai	th	0.541	**0.603**	0.533	**0.636**
Turkish	tr	0.618	**0.653**	0.640	**0.711**

The proposed approach improves the search task modeling performance of the baseline method in the two datasets used for testing. Using the CSTE dataset (Table 1), LASTM surpasses MGBC in all the languages supported by the baseline, obtaining up to 10.9% ($p \leq 0.05$) improvement in the F_1 score for the Turkish language; similarly, LASTM obtains better $F_{0.6}$ scores in fourteen out of sixteen languages, getting an improvement of up to 11.2% ($p \leq 0.05$) in the Turkish language. Furthermore, the monolingual QRY-VEC method, which supports queries in English, obtains an F_1 score of 0.538 and an $F_{0.6}$ score of 0.488 [22]. Consequently, there is no loss in modeling performance when comparing LASTM to the QRY-VEC method. For the CUSTA dataset (Table 2),

[4] https://cloud.google.com/translate.

we observe improvements in fourteen out of the sixteen languages supported by MGBC; LASTM generates up to 9.2% ($p \leq 0.05$) improvement in the F_1 score for the German language and up to 18.4% ($p \leq 0.05$) improvement in the $F_{0.6}$ score for the Dutch language.

Both the similarity between query representations S_{lat} and the similarity based on user intent S_{int} contribute to the search task modeling results. In the grid search for the CSTE dataset, α values averaged 0.238 ± 0.099. For the CUSTA dataset, α values in the grid search averaged 0.731 ± 0.157. These α values indicate that the convex combination (Eq. 6) effectively relies on the two similarities to compute the edges for the weighted graph.

From a language coverage perspective, the query representation for LASTM is trained with 109 languages and can perform zero-shot cross-lingual transfer to multiple more languages [10]. In contrast, the baseline only supports sixteen languages, making LASTM coverage at least seven times larger when considering training languages only. The improvements in modeling results and language coverage highlight the importance of considering user intent along with language-agnostic query representation for modeling search tasks.

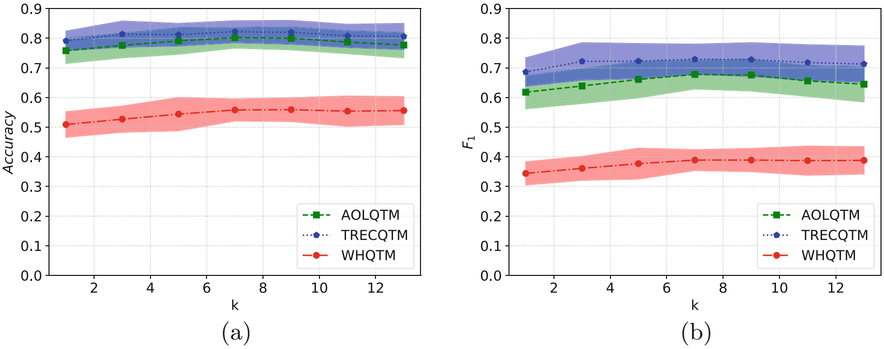


Fig. 1. Search task mapping results in the language-agnostic latent space for AOLQTM, TRECQTM, and WHQTM datasets. Results include several values of top k from the ScaNN index, considering (a) Accuracy (b) F_1.

4.2 Mapping of Incoming Queries

To analyze the performance of LASTM for mapping new incoming queries, we run the mapping method using three benchmark datasets previously proposed for query-task mapping [27]:

- AOL-based Query-Task-Mapping (AOLQTM) dataset, which has 41780 queries and labels for 1423 search tasks.
- TREC-based Query-Task-Mapping (TRECQTM) dataset, which has 47514 queries with labels for 276 search tasks.

- WikiHow-based Query-Task-Mapping (WHQTM) dataset, which has 119292 queries with labels for 7202 search tasks.

We use a leave-one-out evaluation, independently selecting one hundred random queries from the dataset and repeating the evaluation for fifty runs. Experiments run on a virtual machine instance with 8 CPUs of 3 GHz and 60 GB of RAM. Metrics include accuracy, F_1, $F_{0.6}$, and query time. To measure query time, we take the average time for mapping a single query, using 10^4 mappings to compute the average [18,27]. As a baseline, we use the MGBC approach for query task mapping. MGBC combines the Neighborhood Graph and Tree approximate nearest neighbor method [14] with the MUSE latent space for query encoding. For reference, we also include results using the Trie[5] data structure and the BM25[6] retrieval model [18,27,29].

Figure 1 depicts the optimization experiments for the number of top k results from ScaNN to consider. After running tests for $k = [1, 3, 5, 7, 9, 11, 13]$, we found that top $k = 7$ results from ScANN generates the optimal configuration, providing the best results for task mapping while keeping the time per query under a millisecond (Table 3). Low response time is an essential aspect for applications supporting users in realtime setups. Long answer times could affect the interaction of the search system with the users, especially in conversational and multimodal search systems, where a post-processing step is required to generate a response to the user request [15,30]. Similarly, long answer times could trigger internal timeout intervals [30], forcing search systems to ignore search task mapping results while doing internal post-processing.

Table 3. Realtime mapping of queries to search tasks. Differences against baseline MGBC results have $p \leq 0.05$ for the Student's t-test.

Dataset	Method	Accuracy	F_1	$F_{0.6}$	Query time
AOLQTM	Trie	0.693	0.543	0.543	0.029 ms
	BM25	**0.809**	**0.689**	**0.689**	0.947 s
	MGBC	0.751	0.608	0.607	0.308 ms
	LASTM	0.802	0.678	0.677	0.490 ms
TRECQTM	Trie	0.650	0.519	0.518	0.030 ms
	BM25	0.791	0.688	0.688	2.532 s
	MGBC	0.804	0.705	0.704	0.299 ms
	LASTM	**0.822**	**0.729**	**0.728**	0.481 ms
WHQTM	Trie	0.471	0.310	0.311	0.032 ms
	BM25	0.621	0.453	0.454	6.572 m
	MGBC	**0.648**	**0.481**	**0.481**	0.368 ms
	LASTM	0.558	0.389	0.389	0.982 ms

[5] https://github.com/google/pygtrie.

[6] https://github.com/nhirakawa/BM25.

LASTM surpasses the baseline and reference methods in the TREC-based dataset, improving the F_1 score by 2.4% ($p \leq 0.05$), while keeping processing times under a millisecond. For the AOL-based dataset, LASTM surpasses the baseline method, obtaining a 7.0% improvement in the F_1 score ($p \leq 0.05$); likewise, LASTM obtains similar results to BM25, but it is faster when comparing to the BM25 implementation used for experiments. For the WikiHow-based dataset, LASTM underperforms MGBC and BM25 (Table 3). Regarding the number of user queries per task, we find that the TREC-based dataset has an average of 28 user queries per search task, while the WikiHow-based dataset has an average of 2 user queries per task. Hence, the WikiHow-based dataset contains mostly simple tasks, which users can solve with a few queries [13]. Task mapping results suggest that LASTM is better than the baseline and reference methods when mapping search tasks containing multiple queries, while MGBC is better when mapping simple search tasks in realtime.

5 Conclusion

In this paper, we proposed LASTM, an unsupervised method for modeling search tasks from user interactions with search systems. The proposed model outperforms the baseline both in modeling performance as well as the number of languages it can support, highlighting the importance of language-agnostic latent spaces for query representation and the importance of considering clicked URLs to model user intent. Also, it is independent of user identifiers, enabling modeling search tasks in user-independent or personalized applications. The modeling performance of LASTM, its language-agnostic capacity, and its ability to support realtime modeling can benefit search systems and user supporting applications, constituting an essential step in the effort to make search more coherent, conversational, engaging, and natural. For future work, we plan to explore unsupervised alternatives for graph-based clustering to further improve search task modeling.

Acknowledgement. This work was supported by the Agence National de la Recherche (ANR), through project CoST, code ANR-18-CE23-0016.

References

1. Anand, A., Cavedon, L., Joho, H., Sanderson, M., Stein, B.: Conversational search (Dagstuhl seminar 19461). In: Dagstuhl Reports, vol. 9. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)
2. Artetxe, M., Schwenk, H.: Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. Trans. Assoc. Comput. Linguist. **7**, 597–610 (2019)
3. Callan, J.: The Lemur project and its ClueWeb12B dataset. In: Invited talk at the SIGIR 2012 Workshop on Open-Source Information Retrieval (2012)
4. Chen, Z., Ji, H.: Graph-based clustering for computational linguistics: a survey. In: Proceedings of the 2010 workshop on Graph-based Methods for Natural Language Processing, pp. 1–9. Association for Computational Linguistics (2010)

5. Craswell, N., Campos, D., Mitra, B., Yilmaz, E., Billerbeck, B.: ORCAS: 18 million clicked query-document pairs for analyzing search. In: Proceedings of the 29th ACM International Conference on Information and Knowledge Management. ACM (2020)
6. Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Voorhees, E.M.: Overview of the TREC 2019 deep learning track. arXiv preprint arXiv:2003.07820 (2020)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186 (2019)
8. Dosso, C., Chevalier, A., Tamine, L.: How to support search activity of users without prior domain knowledge when they are solving learning tasks? In: Proceedings of the 29th ACM International Conference on Information and Knowledge Management. 1st International Workshop on Investigating Learning During Web Search. ACM (2020)
9. Du, C., Shu, P., Li, Y.: CA-LSTM: search task identification with context attention based LSTM. In: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1101–1104. ACM (2018)
10. Feng, F., Yang, Y., Cer, D., Arivazhagan, N., Wang, W.: Language-agnostic BERT sentence embedding. arXiv preprint arXiv:2007.01852 (2020)
11. Guo, R., Sun, P., Lindgren, E., Geng, Q., Simcha, D., Chern, F., Kumar, S.: Accelerating large-scale inference with anisotropic vector quantization. In: Proceedings of the 37th International Conference on Machine Learning (2020)
12. Hagen, M., Gomoll, J., Beyer, A., Stein, B.: From search session detection to search mission detection. In: Proceedings of the 10th Conference on Open Research Areas in Information Retrieval, pp. 85–92 (2013)
13. Hearst, M.: Search user Interfaces. Cambridge University Press, Cambridge, CB2 8BS, UK (2009)
14. Iwasaki, M., Miyazaki, D.: Optimization of indexing based on k-nearest neighbor graph for proximity search in high-dimensional data. arXiv preprint arXiv:1810.07355 (2018)
15. Khatri, C., Goel, R., Hedayatnia, B., Metanillou, A., Venkatesh, A., Gabriel, R., Mandal, A.: Contextual topic modeling for dialog systems. In: 2018 IEEE Spoken Language Technology Workshop (SLT), pp. 892–899. IEEE (2018)
16. Li, L., Deng, H., Dong, A., Chang, Y., Zha, H.: Identifying and labeling search tasks via query-based Hawkes processes. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 731–740 (2014)
17. Lucchese, C., Orlando, S., Perego, R., Silvestri, F., Tolomei, G.: Identifying task-based sessions in search engine query logs. In: Proceedings of the 4th ACM International Conference on Web Search and Data Mining, pp. 277–286. ACM (2011)
18. Lugo, L., Moreno, J.G., Hubert, G.: A multilingual approach for unsupervised search task identification. In: The 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2041–2044. ACM (2020)

19. Mehrotra, R., Bhattacharya, P., Yilmaz, E.: Deconstructing complex search tasks: a Bayesian nonparametric approach for extracting sub-tasks. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 599–605 (2016)
20. Mehrotra, R., Yilmaz, E.: Extracting hierarchies of search tasks and subtasks via a Bayesian nonparametric approach. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 285–294. ACM (2017)
21. Rosset, C., et al.: Leading conversational search by suggesting useful questions. In: Proceedings of The Web Conference 2020. pp. 1160–1170 (2020)
22. Sen, P., Ganguly, D., Jones, G.: Tempo-lexical context driven word embedding for cross-session search task extraction. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 283–292 (2018)
23. Tamine, L., Melgarejo, J.L., Pinel-Sauvagnat, K.: What can task teach us about query reformulations? In: Jose, J.M., et al. (eds.) ECIR 2020. LNCS, vol. 12035, pp. 636–650. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45439-5_42
24. Thomas, P., McDuff, D., Czerwinski, M., Craswell, N.: Expressions of style in information seeking conversation with an agent. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1171–1180 (2020)
25. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
26. Venkatesh, A., et al.: On evaluating and comparing open domain dialog systems. arXiv preprint arXiv:1801.03625 (2018)
27. Völske, M., Fatehifar, E., Stein, B., Hagen, M.: Query-task mapping. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 969–972 (2019)
28. Wang, H., Song, Y., Chang, M.W., He, X., White, R.W., Chu, W.: Learning to extract cross-session search tasks. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1353–1364. ACM (2013)
29. Yang, Y., et al.: Multilingual universal sentence encoder for semantic retrieval. In: Proceedings of the 58th Annual Meeting of the ACL: System Demonstrations, pp. 87–94. ACL (2020)
30. Zamani, H., Craswell, N.: Macaw: An extensible conversational information seeking platform. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2193–2196 (2020)
31. Zhang, H., et al.: Generic intent representation in web search. In: The 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2019)

DSMER: A Deep Semantic Matching Based Framework for Named Entity Recognition

Yufeng Lyu(✉) and Jiang Zhong

College of Computer Science, Chongqing University, Chongqing Shapingba 400044,
People's Republic of China
{lvyufeng,zhongjiang}@cqu.edu.cn

Abstract. The task of named entitiy recognition(NER) is normally regarded as a sequence labeling problem. However, this kind of NER framework does not utilize any prior knowledge. In this paper, we propose a novel framework called **DSMER**, which stands for **D**eep **S**emantic **M**atching based Framework for Named **E**ntity **R**ecognition. DSMER is a two-phase framework: 1) detect the boundary and extract candidate span, 2) calculate the distance between candidates and entity type. Meanwhile, the representation of each entity type is encoded from its corresponding annotation rules and example set. Since the combination of various textual data, DSMER has the ability to integrate informative prior knowledge. Additionally, we introduce the Word Mover's Distance to measure the similarity between sequences of different lengths. We conduct experiments on CoNLL 2003 and OntoNotes 5.0 dataset. Experimental result shows our approach achieve state of the art performance, and demonstrates the effectiveness of the proposed framework.

Keywords: Named entity recognition · Semantic matching · Entity boundary detection

1 Introduction

Named entity recognition(NER) is a subtask of information extraction, which refers to a task of detecting spans from text and classifying their types. Among mainstream research methods, the NER task is commonly considered as a sequence labeling problem [1,3,6,12,24]: for each token of the input sequence, predict a class label assigned to it. The sequence labeling framework solves NER with an end-to-end way, and has achieved effective results on various datasets.

However, this formalization of NER is quite different from the recognition process of humans. Figure 1 shows human conventions when annotating entity labels. The annotation rules should first be summarized according to human experience and background knowledge. Then the annotator would try to annotate a few examples according to the rules and adjust the rules based on example

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 419–432, 2021.
https://doi.org/10.1007/978-3-030-72113-8_28

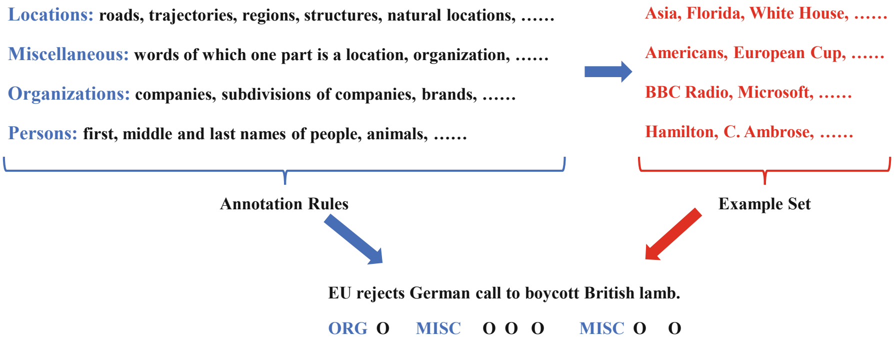

Locations: roads, trajectories, regions, structures, natural locations, ……
Miscellaneous: words of which one part is a location, organization, ……
Organizations: companies, subdivisions of companies, brands, ……
Persons: first, middle and last names of people, animals, ……
Asia, Florida, White House, ……
Americans, European Cup, ……
BBC Radio, Microsoft, ……
Hamilton, C. Ambrose, ……
Annotation Rules
Example Set
EU rejects German call to boycott British lamb.
ORG O MISC O O O MISC O O

Fig. 1. Human annotation process of named entity extraction and recognition. The annotation rules and example set are chosen from CoNLL 2003 dataset.

set. Finally, the annotation rule and the example set are combined together as prior knowledge to carry out the complete data annotation process.

Inspired by human convention, we propose a new framework that is capable of integrating knowledge from annotation rules and example set. Instead of treating NER as a sequence labeling problem, we formulate it as a deep semantic matching task [5,14,22]. Following the principle of two-phase framework [10], we design three sub-modules: 1) Prior Knowledge Encoding: encode the representation of entity types from annotation rules and example set, 2) Boundary Detection: predict the start and end index of candidate entities and extract the representation of them, 3) Semantic Matching: calculate the similarity between candidate span and different types. The input sentence is first sent to the boundary detection module to extract a set of candidates.

At the same time, we combine the annotation rules and example set corresponding to each entity type, and encode them to obtain the representation vector of the entity type. In the second phase, we input the representation vector of each candidate span and entity types into the semantic matching module. The label of candidate span is determined by the similarity of semantic representation between them. In order to measure the similarities between spans and entity types with different lengths, we introduce Word Mover's Distance(WMD) [7], which is a novel distance function based on Earth Mover's Distance(EMD) [20].

We conduct experiments on public NER datasets to show the effectiveness of our approach. Experimental results show that our deep semantic matching based framework outperforms both sequence labeling and machine reading comprehension based frameworks. In addition, we also conducted ablation experiments to verify the influence of different prior knowledge on our method. Our main contributions are summarized as follows:

- We propose a novel deep semantic matching based NER framework which exploits prior knowledge and is closer to human annotation behavior.

- Our boundary detection module overcomes the problem of excessive sample size and imbalance between positive and negative samples in previous entity classification methods.
- We first introduce the Word Mover's Distance into semantic modeling to directly measure the similarity of unequal length sequences.

2 Related Work

Named Entity Recognition(NER). Traditional entity recognition methods treat NER task as a sequence labeling problem and use CRFs as the backbone [8,25]. More recently, neural models was introduced for NER under the sequence labeling framework. Collobert et al. [2] presented a CNN-CRF structure, Huang et al. [6] first applied BiLSTM-CRF model to NER, Lample et al. [9] proposed a BiLSTM-CRF model with character-based word representations, Ma and Hovy [12] and Chiu and Nchols [1] extend the BiLSTM-CRF structure with a character CNN to extract features, Sturbell et al. [24] proposed a iterated dilated convolutions NER model to accelerate the parallel computing on GPU. With the rise of large-scale pre-trained language models [3,16,18,19], sequence labeling style NER models achieved state of the art performance.

In addition to the recognition of flat entities, there are also some studies on nested entities. Previous work was mainly based on the two-phase framework, which first enumerated all possible spans, and then predicted entity type. According to this idea, Sohrab et al. [23] proposed a deep exhaustive model which limited all the regions within a specified maximum length. Zheng et al. [28] leveraged the entity boundaries to improve the performance of identifying entities.

Moreover, Li et al. [11] migrate the NER task to machine reading comprehension framework and make the model compatible with recognizing both flat and nested entities.

Semantic Textual Matching. Huang et al. [5] first proposed the deep structured semantic model(DSSM) in web search area to map a query to its relevant documents at semantic level. The principle is that the query and documents are embedded to semantic vectors, and the distance between them is calculated by cosine distance, and finally the semantic matching model is trained. Aiming at the shortcoming of the bag-of-words model used by DSSM, Shen et al. [22] replaced the DNN with CNN, so that the model can make up for the loss of context. Since the CNN based model can not capture the feature from long term context, Palang et al. [14] introduced the LSTM to overcome the problem.

Word Mover's Distance. Kusner et al. [7] proposed the document distance matrix called Word Mover's Distance(WMD), which can be cast as an instance of the Earth Mover's Distance(EMD). In statistics, the EMD is a measure of the distance between two probability distributions over a region D. If the distributions are interpreted as two different ways of piling up a certain amount of dirt over the region D, the EMD is the minimum cost of turning one pile into the other, where the cost is assumed to be the amount of dirt moved times the distance by which it is moved. The concept of EMD was first introduced by Gaspard Monge [13] in the context of transportation theory. The use of the EMD as a distance measure for monochromatic images was described by Peleg et al. [15]. Stolfi et al. [20] first proposed the name "Earth Mover's Distance". Rubner et al. [20] first used it on image retrieval task to measure the distance between images.

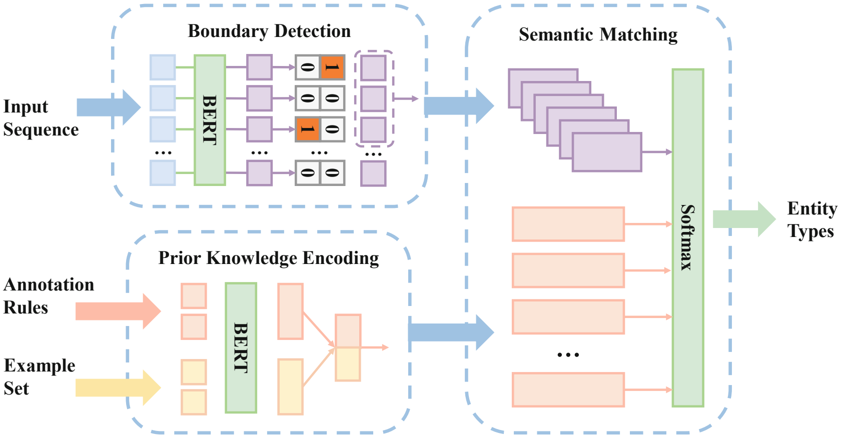


Fig. 2. Overview of deep semantic matching entity recognition framework(DSMER).

3 NER as Semantic Matching

Figure 2 shows the architecture of DSMER. Given an input sequence $X = \{x_1, x_2, ..., x_l\}$, where l denotes the length of the sequence, we need to extract every candidate entity span from X, and then assign a label $t \in T$ to it through semantic matching model, where T is the set of all entity types. The framework is a two-phase model composed of three modules. In the first phase, the representations of candidate spans are extracted, and entity types are encoded through prior knowledge like annotation rules, example set, etc. In the second phase, we separately measure the similarity of each candidate span and all entity types through the semantic matching module. BERT [3] is used as the encoder in each module of the first phase. The following subsections will describe the detail of different modules in DSMER.

3.1 Prior Knowledge Encoding

The prior knowledge encoding procedure is important for DSMER since the external text like annotation rules contains informative semantics and has a significant impact on the final result. Seyler et al. [21] discussed the importance of different categories of external knowledge for performing NER task, including Name-based, Knowledge-Base-based and Entity-based. Besides, Li et al. [11] encoded annotation guideline notes as reference queries and achieved a vast amount of performance boost over current SOTA models. In this paper, we take both annotation rules and example set of entity mentions as prior knowledge. Annotation rules are not only the guidelines provided to the annotators of the dataset but the Wikipedia definition and synonyms of entity type.

Assuming E_t is the representation of entity type t. Given a list of annotation rules $R = [r_1, r_2, ..., r_n]$ and a set of example mentions $S = s_1, s_2, ..., s_m$, where n and m denote the number of rules and mentions. We first encode the annotation rules and the example set separately, and then concatenate the hidden representations of them as E_t:

$$E_t = tanh(W_t[E_R, E_S] + b_t) \tag{1}$$

where E_R and E_S are both encoded by BERT, W_t and b_t is the trainable weight and bias:

$$\begin{aligned} E_R &= \frac{1}{n}\sum_{i=1}^{n} BERT(r_i) \\ E_S &= \frac{1}{m}\sum_{j=1}^{m} BERT(s_j) \end{aligned} \tag{2}$$

In particular, we only take the output context representation of [CLS] position to calculate the average representation of rules and mentions with different lengths.

3.2 Boundary Detection

The boundary detection module is designed to recognize all possible candidate span in the input sentence X. Previous work [23,28] simply set a maximum length of entity, and enumerated all possible spans as a candidate set, which caused the imbalance of positive and negative samples and the problem that the number of samples increased exponentially with the length of the input sequence. To tackle this problem, we use two binary classifiers: one to predict whether each token is the start index or not, the other to predict the end index. Figure 3 shows the architecture of boundary detection module.

Given the representation matrix E_X output from BERT,

$$E_X = BERT(X), \quad E \in R^{n \times d} \tag{3}$$

where d is the dimension size of the output layer of BERT. The module adopts two fully-connected layers to detect the start and end position indexed respectively by assigning each token a binary tag (0/1).

$$P^i_{start} = \sigma(W_{start}E_{x_i} + b_{start}) \tag{4}$$

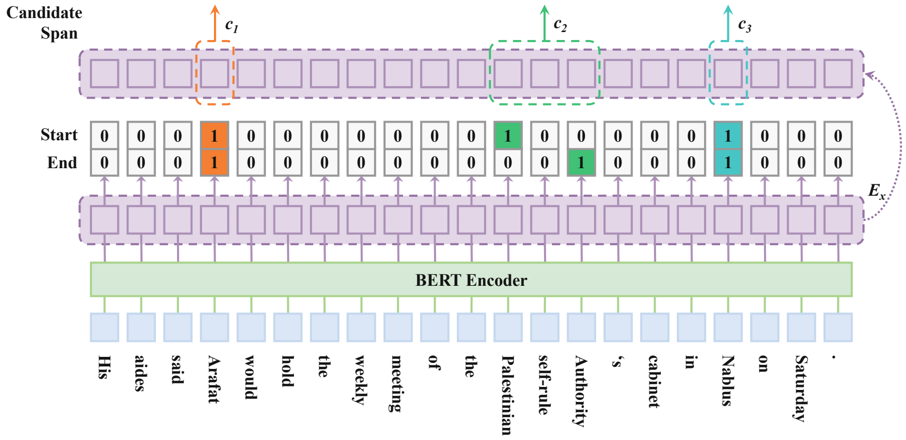


Fig. 3. The workflow of boundary detection module.

$$P^{i}_{end} = \sigma(W_{end}E_{x_i} + b_{end}) \tag{5}$$

where P_i^{start} and P_i^{end} represent the probability of identifying the i-th token in the input sequence X as the start and end position of a candidate span.

After predicting the start and end positions, we combine start index and each end index greater than it as a candidate span c, and extract the representation $E_c = \{E_{x_{start}}, E_{x_{end}}\}$ for semantic matching in next phase.

3.3 Semantic Matching

The semantic matching module is a deep neural network following DSSM [5] and CLSM [22]. Figure 4 shows the structure of this module. Considering the ground truth type $t^+ \in T$, which is closer to candidate span than other types in semantic space. We can simply use the deep semantic model to calculate the relevance of each pair of (c, t).

To directly measure the difference between two sequences of different lengths, we introduce the Word Mover's Distance. Considering the embedding of entity span E_c and the embedding of entity type E_t, the cost of WMD can be calculated by:

$$\begin{gathered}\min_{d_{i,j}\geq 0} \sum_{i,j} d_{i,j} \left\| e_i - e'_j \right\| \\ \text{s.t.} \sum_i d_{i,j} = \frac{1}{l_c}, \sum_j d_{i,j} = \frac{1}{l_t}\end{gathered} \tag{6}$$

where l_c and l_t are the length of candidate span and entity type vector, e_i and $e_j^{'}$ are i-th and j-th embedding vector in E_c and E_t. The semantic relevance score between a candidate c and a entity type t is then measured as:

$$M(c,t) = WMD(E_c, E_t) \tag{7}$$

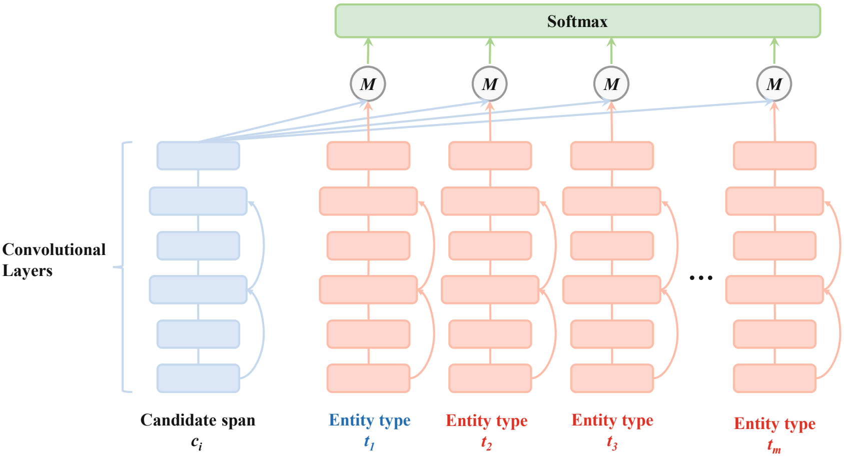


Fig. 4. The structure of deep semantic matching module. Let t_1 be the matched entity type of candidate span c_i, and all others are negative examples. Send their representations into the model, calculate the similarity of each pair, and finally output the posterior probability through softmax layer.

After obtaining the semantic relevance score, we compute the posterior probability through a softmax function:

$$P(t|c) = \frac{exp(M(c,t))}{\sum_{t' \in T} exp(M(c,t'))} \tag{8}$$

In particularly, we adopt shortcut connections every other layer parallel to linear transformation before the activation function, as in ResNet [4]. This helps the training of a deep neural network.

3.4 Loss Function

At the training time, X is paired with two label sequences Y_{start} and Y_{end} that represent the ground-truth label of each token x_i. We use the binary cross-entropy loss for the prediction of start and end index:

$$L_{start} = BCE(P_{start}, Y_{start}) \tag{9}$$

$$L_{end} = BCE(P_{end}, Y_{end}) \tag{10}$$

The parameters of semantic matching module are estimated to maximize the likelihood of t^+. Equivalently, we need to minimize the following loss function:

$$L_{match} = -log \prod_{(c,t^+)} P(t^+|c) \tag{11}$$

The overall training objective to be minimized is as follows:

$$L = \alpha L_{start} + \beta L_{end} + \gamma L_{match} \tag{12}$$

where $\alpha, \beta, \gamma \in [0, 1]$ are the hyper-parameters to control the contributions of different modules. The three losses from two phrase of DSMER are jointly trained with parameters shared at BERT.

At the test time, candidate spans are first extracted based on boundary detection module. Then the semantic matching model is used to measure the similarity of candidate span and entity types, leading to the final answers.

4 Experiments and Discussions

In this section, we conduct experiments on several public datasets and compare DSMER with models of different NER framework. The following subsections will describe the implementation details and ablation analysis in detail.

4.1 Datasets and Preprocessing

Datasets. We use corpora provided by CoNLL 2003 Shared Task [26] and OntoNotes 5.0 [17] to evaluate the model presented in this paper. CoNLL2003 is an English dataset with four types of named entities: Location, Organization, Person and Miscellaneous. And Ontonotes 5.0 includes 18 types of named entity, consisting of 11 types (Person, Organization, etc.) and 7 values (Date, Percent, etc.).

Data Reconstruction. Most NER corpora provide the labeled data for sequence labeling framework. Different from other NER frameworks, the DSMER needs to extract the rules from annotation document and random sampling part of entities for each type from raw dataset.

For each train set, we random choose 10% annotated entities as example set, and remain 90% as train set as usual. The statistical details are listed in Table 1. To further experiment, we also test the ratio of 5%, 15%, 20% and 40% in following experiments.

Table 1. The entity statistics of preprocessed datasets.

Corpus	Example set	Train set	Dev set	Test set
CoNLL 2003 [26]	2,350	21,149	5,942	5,648
OntoNotes 5.0 [17]	8,183	73,645	11,066	11,257

As for the boundary detection module, training data requires binary label for start and end indexes. The ground truth label of entities is converted into two lists for start and end, which are set to 1 only when the token belongs to the boundary of the entity.

4.2 Implementation Details

We use fastNLP[1] to implement the model and evaluate all experiments on datasets. The DSMER model uses BERT as the skeleton. In order to ensure the effectiveness of the semantic matching method, we only use BERT-base as a semantic encoder in all the comparison experiments below. All experiments are run on Nvidia Tesla V100 GPU, which has 32 GB memory to accommodate larger batch size.

Table 2. Hyper-parameter settings.

Parameters	Values
Optimizer	AdamW
Initial learning rate	2e–5
Gradient clipping value	1.0
Global dropout rate	0.5
Warmup rate	0.1
Batch size	64
Training epoch	20
Layer of DSM	5
Hidden dim of DSM input	300

We train the model using *AdamW* optimizer with an initial learning rate of 2e–5, and use warm-up mechanism with linear schedule to adjust the learning rate. To avoid gradient explosion problem, the gradient clip method is used as a callback in training. The semantic matching module of DSM follows the deep structured nerual network in [5], We use 5 fully connected layers, and the input dimension of candidate span and entity types is 300. All other details of hyperparameters are listed in Table 2.

4.3 Experimental Results

In order to verify the effectiveness of DSMER, we choose the classic and SOTA models under different NER frameworks for comparison. For sequence labeling framework, we change the encoder module connected to CRF in range of Bi-LSTM, IDCNN and Transformer. And BERT is also introduced for the pretrain+finetune framwork. Finally we use the MRC-BERT model to stand the machine reading comprehension framework. All comparison results on CoNLL2003 and Ontonotes 5.0 are listed in Table 3 and 4.

Because we use BERT-base as the model skeleton, we respectively give the experimental results without using the annotation rule and example set to verify the effectiveness of the semantic matching framework.

[1] https://github.com/fastnlp/fastNLP.

Table 3. Comparison with other NER models on Conll2003.

Framework	Model	Precision	Recall	F1
Sequence labeling	BiLSTM + CRF [6]	–	–	90.43
	IDCNN + CRF [24]	–	—	90.54
	TENER w/CNN-char [27]	–	—	91.45
	BERT-Tagger [3]	–	–	92.80
Reading comprehension	MRC-BERT [11]	92.33	94.61	93.04
Semantic matching	Ours w/o example set	91.75	90.13	90.93
	Ours w/o annotation rule	**92.75**	94.81	93.76
	Ours	92.74	**95.07**	**93.89**

Experimental results on CoNLL 2003 show a slight improvement by DSMER without example sets. However, significant improvement has been achieved under the conditions of only using the example set. At the same time, we observe that using example set and annotation rule can not improve all factors. This is because the example set can better represent the scope of the entity type in the semantic space, but the description text of the annotation rule may cause a certain offset, which makes the calculation of the semantic similarity also be affected.

Table 4. Comparison with other NER models on OntoNotes 5.0.

	Model	Precision	Recall	F1
Sequence labeling	LSTM + CRF [6]	–	–	86.99
	IDCNN + CRF [24]	–	–	86.84
	TENER w/CNN-char [27]	–	–	88.43
	BERT-Tagger [3]	–	–	89.16
Reading comprehension	MRC-BERT [11]	**92.98**	89.95	91.11
Semantic matching	Ours w/o example set	90.56	88.79	89.67
	Ours w/o annotation rule	92.90	90.27	91.57
	Ours	92.95	**90.47**	**91.69**

Similar results are also observed in the experiment on the OnteNotes 5.0 dataset. However, the use of annotation rule can still improve F1 score, so we think it is effective prior knowledge. Comparative experiments show that DSMER can handle NER problems. We continue to conduct more ablation experiments in Subsect. 4.4 to analyze the impact of different model designs on performance.

4.4 Ablation Studies

The Impact of Example Set. As shown in Table 3 and 4, whether to use example set has a great influence on model performance. In order to observe the impact of the size of the example set on the model, we split the data set according to the split ratio of Subsect. 4.1, and test it on the CoNLL 2003 dataset. The results are shown in Table 5:

Table 5. The impact of the percentage of example set, experiments on CoNLL 2003.

Percentage	Precision	Recall	F1
5%	91.67	94.23	92.93
10%	**92.75**	94.81	**93.76**
15%	92.60	**94.95**	93.76
20%	91.83	93.79	92.80
40%	91.43	91.88	91.65

It can be seen that the 10% and 15% split ratios have the best effect. And as the proportion of the example set increases, the overall effect decreases since the lack of training data. Since all entities in the example set are phrases that can express their entity type, a large number of entity examples can better express the position of the entity type in the high-dimensional semantic space. In this way, the calculation of the distance between candidate span and entity type is more accurate. But with the increase of the example set, the decrease of training data makes the model easy overfitting on the training data. This is a trade-off process for dataset segmentation. Comparing with other models, we choose 10% as the segmentation ratio.

The Impact of Annotation Rules. How to construct the annotation rule sentence also has a significant influence on the final results. In this subsection, we explore difference sources to construct annotation rules and their influence, including:

- **Annotation guideline:** the annotation rule from documents, like *"find organizations including companies, agencies and institutions"*.
- **Wikipedia:** the wikipedia definition of entity type, like *"an organization is an entity comprising multiple people, such as an institution or an association."*
- **Synonyms:** word or phrases that mean nearly the same as the entity type word from Dictionary, like *"association"*
- **All above:** encode above three concepts and use the average representation.

Table 6 shows the experimental results on CoNLL 2003. DSMER outperforms BERT-tagger by using different types of annotation rules. Among them,

Table 6. Results of different types of annotation rules on CoNLL 2003.

Model	F1
BERT-Tagger	89.16
Annotation guideline	90.21(+1.05)
Wikipedia	89.65(+0.49)
Synonyms	89.90(+0.74)
All above	**90.93(+1.77)**

the effect of using annotation guideline is the best among the three categories, because it is the closest text description to the entity annotation. At the same time, it can be seen that the combined usage of three different kind of rules can achieve better performance improvement.

5 Conclusion

In this paper, we introduce a novel framework for named entity recognition task which reflect the natural entity annotation process of human being. The proposed model obtain state of the art results on public datasets, which indicates the effectiveness of DSMER. The deep semantic matching based framework shows a possible new paradigm to tackle such problem. We would like to explore more variant of the framework in the future.

Acknowledgement. This work is supported by the National Key Research and Development Program of China (grant No. 2017YFB1402400 and No. 2017YFB1402401) and the Key Research Program of Chongqing Science and Technology Bureau (grant No. cstc2019jscx-mbdxX0012, No. cstc2019jscx-fxyd0142 and No. cstc2020jscx-msxmX0149).

References

1. Chiu, J.P., Nichols, E.: Named entity recognition with Bidirectional LSTM-CNNs. Trans. Assoc. Comput. Linguist. **4**, 357–370 (2016). https://www.aclweb.org/anthology/Q16-1026
2. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). 10.18653/v1/N19-1423. https://www.aclweb.org/anthology/N19-1423
4. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision - ECCV 2016, pp. 630–645. Springer International Publishing, Cham, Lecture Notes in Computer Science (2016)

5. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 2333–2338. CIKM 2013, Association for Computing Machinery, New York, NY, USA, October 2013. https://doi.org/10.1145/2505515.2505665
6. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF Models for Sequence Tagging. arXiv:1508.01991, August 2015. arXiv: 1508.01991
7. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings To document distances. In: International Conference on Machine Learning, pp. 957–966, June 2015. http://proceedings.mlr.press/v37/kusnerb15.html
8. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289. ICML 2001. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, June 2001
9. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 260–270. Association for Computational Linguistics, San Diego, California, June 2016. https://doi.org/10.18653/v1/N16-1030,https://www.aclweb.org/anthology/N16-1030
10. Lee, K.J., Hwang, Y.S., Rim, H.C.: Two-phase biomedical NE recognition based on SVMs. In: Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine, pp. 33–40. Association for Computational Linguistics, Sapporo, Japan, July 2003. https://doi.org/10.3115/1118958.1118963, https://www.aclweb.org/anthology/W03-1305
11. Li, X., Feng, J., Meng, Y., Han, Q., Wu, F., Li, J.: A unified MRC framework for named entity recognition. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5849–5859. Association for Computational Linguistics, July 2020. https://doi.org/10.18653/v1/2020.acl-main.519,https://www.aclweb.org/anthology/2020.acl-main.519
12. Ma, X., Hovy, E.: End-to-end sequence labeling via Bi-directional LSTM-CNNs-CRF. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1064–1074. Association for Computational Linguistics, Berlin, Germany, August 2016. https://doi.org/10.18653/v1/P16-1101, https://www.aclweb.org/anthology/P16-1101
13. Monge, G.: Mémoire sur la théorie des déblais et des remblais. Histoire de l'Académie Royale des Sciences de Paris (1781)
14. Palangi, H., et al.: Semantic Modelling with Long-Short-Term Memory for Information Retrieval. arXiv:1412.6629, Feburary 2015. http://arxiv.org/abs/1412.6629, arXiv: 1412.6629
15. Peleg, S., Werman, M., Rom, H.: A unified approach to the change of resolution: space and gray-level. IEEE Trans. Pattern Anal. Mach. Intell. **11**(7), 739–742 (1989). https://doi.org/10.1109/34.192468, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence
16. Peters, M., et al.: Deep contextualized word representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 2227–2237. Association for Computational Linguistics, New Orleans, Louisiana, June 2018. https://doi.org/10.18653/v1/N18-1202, https://www.aclweb.org/anthology/N18-1202

17. Pradhan, S., et al.: Towards Robust Linguistic Analysis using OntoNotes. In: Proceedings of the Seventeenth Conference on Computational Natural Language Learning, pp. 143–152. Association for Computational Linguistics, Sofia, Bulgaria, August 2013. https://www.aclweb.org/anthology/W13-3516
18. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving Language Understanding by Generative Pre-Training. OpenAI (2018)
19. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners, 24
20. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. Int. J. Comput. Vis. **40**(2), 99–121 (2000). https://doi.org/10.1023/A:1026543900054
21. Seyler, D., Dembelova, T., Del Corro, L., Hoffart, J., Weikum, G.: A study of the importance of external knowledge in the named entity recognition task. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 241–246. Association for Computational Linguistics, Melbourne, Australia, July 2018. https://doi.org/10.18653/v1/P18-2039,https://www.aclweb.org/anthology/P18-2039
22. Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: A latent semantic model with convolutional-pooling structure for information retrieval. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pp. 101–110. CIKM 2014, Association for Computing Machinery, New York, NY, USA, November 2014. https://doi.org/10.1145/2661829.2661935
23. Sohrab, M.G., Miwa, M.: Deep exhaustive model for nested named entity recognition. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2843–2849. Association for Computational Linguistics, Brussels, Belgium, October 2018. https://doi.org/10.18653/v1/D18-1309, https://www.aclweb.org/anthology/D18-1309
24. Strubell, E., Verga, P., Belanger, D., McCallum, A.: Fast and accurate entity recognition with iterated dilated convolutions. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2670–2680. Association for Computational Linguistics, Copenhagen, Denmark, September 2017. https://doi.org/10.18653/v1/D17-1283, https://www.aclweb.org/anthology/D17-1283
25. Sutton, C., McCallum, A., Rohanimanesh, K.: Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. J. Mach. Learn. Res. **8**, 693–723 (2007). https://www.jmlr.org/papers/v8/sutton07a.html
26. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pp. 142–147 (2003). https://www.aclweb.org/anthology/W03-0419
27. Yan, H., Deng, B., Li, X., Qiu, X.: TENER: Adapting Transformer Encoder for Named Entity Recognition. arXiv:1911.04474 [cs], December 2019
28. Zheng, C., Cai, Y., Xu, J., Leung, H.f., Xu, G.: A boundary-aware neural model for nested named entity recognition. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 357–366. Association for Computational Linguistics, Hong Kong, China, November 2019. 10.18653/v1/D19-1034, https://www.aclweb.org/anthology/D19-1034

Predicting User Engagement Status for Online Evaluation of Intelligent Assistants

Rui Meng[1(✉)], Zhen Yue[2], and Alyssa Glass[3]

[1] University of Pittsburgh, Pittsburgh, PA 15213, USA
rui.meng@pitt.edu
[2] Disney Streaming Service, CA, USA
zhen.yue@disney.com
[3] Apple Inc., CA, USA

Abstract. Evaluation of intelligent assistants in large-scale and online settings remains an open challenge. User behavior based online evaluation metrics have demonstrated great effectiveness for monitoring large-scale web search and recommender systems. Therefore, we consider predicting user engagement status as the very first and critical step to online evaluation for intelligent assistants. In this work, we first propose a novel framework for classifying user engagement status into four categories – fulfillment, continuation, reformulation and abandonment. We then demonstrate how to design simple but indicative metrics based on the framework to quantify user engagement. We also aim for automating user engagement prediction with machine learning methods. We compare various models and features for predicting engagement status using four real-world datasets. We conduct detailed analyses on features and failure cases to discuss the performance of current models as well as potential challenges.([1]Resources used in this study can be found at https://github.com/memray/dialog-engagement-prediction.)

Keywords: Intelligent assistant · User engagement · Online evaluation

1 Introduction

The increasing popularity of intelligent assistants such as Alexa, Siri and Google Home has attracted broad attention to human-machine dialogue systems, but also brought challenges for evaluating the performance of dialogue systems in online environments. Previous research demonstrated that the most effective way to improve any online system is to optimize it for end-user engagement [6]. For example, recommender systems can be optimized for user click and dwell time [43] and web search systems can be optimized for click-through rate [9]

R. Meng, Z. Yue and A. Glass—This work was done when the authors were at Yahoo Research.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 433–450, 2021.
https://doi.org/10.1007/978-3-030-72113-8_29

and reformulation rate [14]. Nevertheless, designing proper metrics to optimize online intelligent assistant systems remains a big challenge.

Previous studies seeking to evaluate dialogues systems mainly focus on the performance of individual system component rather than overall user engagement. The common practice in system-oriented evaluation is breaking down the dialogue system into parts, such as dialogue act classification and state tracking, and evaluating the performance of each component respectively. However, we cannot assess the performance of the whole system by simply aggregating the performance of each component. There were several methods developed to evaluate the overall system performance. For example, one can evaluate the quality of system responses by measuring their similarities to ground-truth responses with metrics like BLEU [39,40]. However, users' requests in online environment are very diverse and it is very expensive to build ground-truth datasets, which make the evaluation hard to scale up for online scenarios.

Research in web search has a long history of conducting large-scale online evaluation utilizing user engagement and behavior signals [8,13–15]. The idea was to regard possible user interaction outcomes as different engagement types, such as long-dwell click, query reformulation and abandonment. These engagement types can then be used to gauge search success and cost, thus making these measurements scalable for online evaluation. We think that the same idea can be adopted to the evaluation of intelligent assistants as well. For example, we can classify each user utterance in a dialogue system into success and failure requests. Previous research proposed a conceptual framework PARADISE [41] for evaluating dialogue systems. It pointed out that a successful dialogue system should maximize task success and minimize cost. Same for the online evaluation of intelligent assistant, we should not only focus on whether or not users' requests have been fulfilled but also measure how much effort it takes. We cannot simply use the conversation length as measurement for cost, since it might take multiple necessary turns to finish a complex user request. Instead, we should focus on whether or not the interaction is necessary for the intelligent assistant to fulfill the request. In order to solve the problem, we proposed a novel scheme categorizing users' utterances into different types of engagement status, with which we can design metrics to measure task success and cost for online evaluation of intelligent assistant.

Furthermore, we aim for a more challenging task, delivering an automatic method for predicting the user engagement status. In recommendation and search, researchers utilize behavior signals such as dwell time and query content features to predict user engagement. Similarly, we utilize interaction signals between users and intelligent assistants to predict users' engagement status. Comparing to the short queries in web search, the interaction between users and intelligent assistants contains rich contents, which can be used for creating sophisticated automatic methods. We investigate various machine learning models and feature settings for the engagement prediction task with four newly annotated datasets.

2 Related Work

2.1 Evaluation of Intelligent Assistants

There are several major methods being widely used for evaluating intelligent assistants: (1) Evaluation on specific components [10,22,33]. People have established several tasks to examine certain aspects of the systems, such as dialog state tracking and dialogue act classification, and evaluate them by metrics like precision and recall. While these evaluations are useful to identify problems in each component, the outcomes cannot reflect the overall performance of the dialogue system. (2) Evaluation by comparing system responses with ground-truth responses [28,36,39]. This type of approaches is broadly adopted for response generation. The basic idea is to measure the similarity between generated responses and ground-truth responses with metrics like BLEU [34]. However, a high degree of token matching may imply its readability, but does not mean it is a logical response, and such methods have been proved correlated poorly with the human judgment [29]. (3) There are a few tasks aiming to detect problematic system responses which share a similar motivation to our study, such as error detection [26,31] and breakdown detection [18]. But in these tasks, the cost of communication is not considered and task boundaries are presumably given. In the real world, both task success and cost affect users' experience considerably and users can move to a new task anytime, therefore our specially designed framework, detecting both system failures and user request boundaries, are more suitable for evaluating real-world systems.

2.2 User Engagement Prediction

User satisfaction rating in dialogue systems has been discussed for a long time [21,35,38]. A wide variety of techniques and features have been studied [4,11,42], as well as some recent efforts on the basis of deep neural networks [30]. Most of these studies output a holistic satisfaction rating for the entire dialogue, but it cannot offer any further information about how the system fails to satisfy users. Therefore it is not a reliable optimization target that can be used for improving the dialogue system.

PARADISE [41] framework tackles this problem by breaking down the measurement of user satisfaction into two parts: task success and dialogue cost. However, PARADISE was proposed more than twenty years ago and is more a conceptual framework than a practical solution: the two factors are too general to implement and they did not provide any automatic method practicable in a nowadays large-scale setting. For example, to measure task success it requires conversations to be represented as task-specific attribute-value pairs which are not flexible nor transferable. The measurements of dialogue cost using utterance length and number of dialogue turns are also arguable, because for many user requests, say the restaurant inquiry in DSTC2 (see Table 1), it can take several necessary turns to finish. Our work is also similar to [1,12,20,25] in the sense of using user behavior signals to predict short-term user engagement/satisfaction.

3 Classifying and Measuring User Engagement

3.1 Framework for Classifying User Engagement

Before introducing the classification framework, we would like to clarify several concepts that play important roles in it. We consider a continuous interaction between user and system within a small range of time as a *session*. And a *session* is comprised of a number of *utterances*, each of which is issued by either the user or the system. We can further group *utterances* into *tasks* basing upon user's potential information needs. For example, Table 1 shows a *session* sampled from the dataset DSTC2, in which a user (**User**) consults a dialogue system (**Bot**) about restaurants and the system asks the user to provide necessary information to narrow down the scope. It consists of 7 effective pairs of *user utterance* and *system utterance*, and 4 basic *tasks* (or *user requests*): requesting a restaurant (#1-#3), requesting its address (#4), and requesting its phone number (#5) and requesting its postcode (#6-#7).

With the goal of measuring both success and cost of user interaction with intelligent assistant, we propose a four-class utterance classification scheme. Each class represents the engagement status of a user after issuing an utterance:

Table 1. An annotated session from ***DSTC2***.

# Turn	Utterance	Label	Task
	Bot: Hello, welcome to the Cambridge restaurant system? You can ask for restaurants by area, price range or food type. How may I help you?		
1	**User:** Moderately priced	R	**Success**
	Bot: What part of town do you have in mind?		
2	**User:** North	C	
	Bot: Would you like something in the cheap, moderate, or expensive price range?		
3	**User:** Moderate	F	
	Bot: Golden wok is a nice restaurant in the north of town in the moderate price range		
4	**User:** Address	F	**Success**
	Bot: Sure , golden wok is on 191 Histon Road Chesterton		
5	**User:** Phone number	F	**Success**
	Bot: The phone number of golden wok is 01223 350688		
6	**User:** Postcode	R	**Failure**
	Bot: Golden wok is a nice restaurant in the north of town in the moderate price range		
7	**User:** Postcode	A	
	Bot: Golden wok is a nice restaurant in the north of town in the moderate price range		
8	**User:** Thank you good bye		

- ***Fulfillment*** **(F)**: current user request is understood and fulfilled by the system.
- ***Continuation*** **(C)**: current user request is understood by the system but more interactions are needed.
- ***Reformulation*** **(R)**: current user request is wrongly or incompletely understood by the system, and user repeats/rephrases this request in the next turn.
- ***Abandonment*** **(A)**: current user request is not understood or fulfilled by the system, so the user abandons this request by closing the conversation or starting a new request.

In Table 1, user utterances are annotated with the proposed classification scheme, as shown in the rightmost column. Specifically, the user told the system her desired price range and location (*Turn #1* and *#2*), but the system failed to catch the first price information. After the user repeated it (*Turn #3*), the system returned a restaurant that the user might be interested in. It is worth noting that the annotation of an utterance utt_i has to be one-turn delayed, determined after knowing the future responses (utt_{i+}) from both the system and the user side. Therefore the *Turn #1* utterance is annotated as '**R**'. The system replied correctly in both *Turn #4* and *#5*. The user requested the postcode in *Turn #6* and repeated it in the *Turn #7*, and in the end she terminated the conversation after an incorrect response. Thus #6 is labeled as '**R**' and #7 is '**A**'.

Table 2. Two dimensions along which the proposed classification scheme can be binarized.

	Ongoing	Ending
Correctly responded	Continuation	Fulfillment
Wrongly responded	Reformulation	Abandonment

From the definition of each type and the examples, we can see that the proposed classification scheme is clearly defined and highly explainable, because the four classes of user utterance are mutually exclusive and each depicts an explicit user behavior. As shown in Table 2, our scheme can be thought as two orthogonal binary classifications by checking (1) if the user continues or terminates the current task/request and (2) if the system gives a correct or wrong response. Based on the two conditions, one can assign labels much easier than giving a subjective score [30,42] or a sentiment class [4]. For example, we can split the session in Table 1 into four tasks and classify them in into **Success** or **Failure** using **F** or **A** as task boundary and satisfaction indicator.

3.2 Online Evaluation Metrics Based on User Engagement Status

In the context of industrial web services, ahead of optimizing any system to improve its performance for end users, it is common to first determine how to measure the user engagement with a system, i.e. creating engagement metrics that accurately reflect the user-end performance of a product. With the proposed classification scheme, not only are we able to understand the engagement status of a user after each request immediately, it also enables us to define a series of evaluation metrics to monitor the system performance in an online manner. Similar to PARADISE [41], we define two metrics to measure the user engagement, from the aspect of success and cost respectively.

Since *Fulfillment* or *Abandonment* indicates the boundary of a task as well as a good/bad user experience, we can split a session to several tasks, and then group them into successful/unsuccessful tasks. We define the **Success Rate** of a session $\mathbb{S}$ as the percentage of success tasks as in Eq. (1), where $\#(TASK_{success\in\mathbb{S}})/\#(TASK_{\in\mathbb{S}})$ denotes the number of successful/all tasks in the session $\mathbb{S}$:

$$\mathit{SuccessRate} = \frac{\#(TASK_{success\in\mathbb{S}})}{\#(TASK_{\in\mathbb{S}})} \tag{1}$$

Similarly, we would like a metric to represent how efficiently a system can respond to requests. Firstly, we can use a statistic of *Reformation* to represent the degree to which a user repeats in a task. We define **Reformulation Rate** of a session as the average percentage of reformulated utterances in each task as in Eq. (2), where $\#(UTT_{reform\in T})/\#(UTT_{\in T})$ denotes the number of *Reformulation*/all user utterances in the task **T**. Furthermore, we hope the final metric can also reflect the degree of user fatigue in the interaction. Though *Continuation* utterances are considered necessary in most cases, we think long dialogues should be avoided and better interaction models can be designed to shorten the length. To this end, we define **Fatigue Value** as the average thresholded length of a task as in Eq. (3) – if a task is longer than α turns (α is a preset parameter), we count its fatigue value as $\#(UTT_{\in T})-\alpha$ otherwise as 1. Then we define **Efficiency Rate** as in Eq. (4), which means the less reformulation or the shorter dialogue in each task, the more efficient we consider a session is.

$$\mathit{ReformRate} = \sum_{T\in\mathbb{S}} \frac{\#(UTT_{reform\in T})}{\#(UTT_{\in T})} \tag{2}$$

$$FatigueValue = \frac{\sum_{T\in\mathbb{S}} max(1, \#(UTT_{\in T}) - \alpha)}{\#(TASK_{\in\mathbb{S}})} \tag{3}$$

$$\mathit{EfficiencyRate} = \frac{1 - \mathit{ReformRate}}{FatigueValue} \tag{4}$$

Lastly, we can define a unified **User Engagement Score** representing the overall user experience of a session. Here we define it as a plain arithmetic mean of both Success Rate and Efficiency Rate (Eq. (5)), but it can be extended to more sophisticated forms to fit specific cases and applications.

$$UE_{SCORE} = \frac{\mathit{SuccessRate} + \mathit{EfficiencyRate}}{2} \tag{5}$$

Overall, this classification scheme and metrics are conducted at the utterance level, which is easy-to-run for real-time systems. Furthermore, as the proposed user engagement status can indicate a positive/negative experience explicitly, the corresponding metrics are explainable and instructive for troubleshooting potential system problems.

3.3 Datasets

Since there does not exist dataset available for our study, we collect data from four intelligent assistants – **DSTC2, DSTC3, Yahoo Captain (YCap), Google Home (GHome)** – and annotate them. All dialogues take place between a human and a real system , which fit our goal of evaluating real intelligent assistants. **DSTC2** [16] and **DSTC3** [17] are task-specific datasets, in which users call the system to inquire restaurant or tourist information. **YCap** is an SMS-based family assistant developed by Yahoo!. It supports functions like setting a reminder for family members, maintaining and sharing shopping list etc. **GHome** is collected from real users of Google Home, an intelligent home device powered by Google Assistant and responding to voice control with multiple functions. The **GHome** dataset is the most complicated among the four datasets. It not only covers a broad range of tasks including reminder, timer, search, in-house device control etc., but also supports open-domain chitchat. For **YCap** and **GHome**, as all the conversations are concatenated in a log file, we split dialogues by checking if the interval between two utterances is more than 10 min. Then we randomly select 1,000 anonymized dialogues from each dataset for annotation. We ask professional annotators to judge the engagement status of each user utterance. The first pass of annotation is done by two annotators independently and the conflicts are resolved by the third annotator. The inter-annotator agreement achieves a kappa of 0.790, indicating the proposed scheme is understandable and easy-to-annotate. Table 3 shows the statistics of each dataset. **#(user utt)** indicates the number of data examples used in the following study. Here we highlight several observations:

Table 3. Statistics of four annotated dialogue datasets

Dataset	#(task)	#(utt) per task	#(word) per utt	#(user utt)	C%/R%/F%/A%	Suc%/Effic%/Ref%/Fatigue	*UE*
DSTC2	2,825	4.36	3.87	5,700	28.6%/21.9%/47.1%/2.5%	93.8%/41.9%/17.0%/3.33	0.679
DSTC3	3,020	4.64	4.00	5,856	28.1%/20.4%/48.0%/3.6%	90.1%/45.1%/14.6%/4.01	0.676
YCap	2,733	2.37	4.49	3,530	7.6%/14.9%/70.8%/6.6%	91.8%/78.7%/12.4%/1.35	0.853
GHome	4,561	2.98	4.17	5,241	2.3%/10.6%/75.7%/11.4%	87.4%/73.3%/8.3%/1.80	0.804

1. By checking the average number of user utterances (#(utt) per task), dialogues of the text-based system (**YCap**) are averagely shorter than the ones of spoken systems. Also, since **YCap** takes the user typed input directly, though the data is intact from the error-prone ASR, it suffers from the typo errors of user inputs.
2. *Continuation* accounts for a large part in **DSTC2/3**. This is because, in order to inquire restaurants of interest, users have to interact with the system for

many turns. But in **YCap** and **GHome**, most user requests can be solved in one turn, such as “set up a reminder at 8pm” or “turn on the light”.

3. Utterances of *Fulfillment* and *Continuation* take the major part across all four datasets. By summing up these two types, we can see a basic success rate of each system at the utterance level (75.6%:75.9%:78.4%:78.0%).
4. *Abandonment* on task-specific systems is notably fewer than on more complicated systems such as **GHome**, which can be attributed to the fact that tasks in **GHome** are more diverse and difficult.
5. Overall, the class distribution is very skewed, and models may severely suffer from the data scarcity on minor classes.

3.4 Case Study of User Engagement Metrics

We compute the user engagement scores of each dataset as shown in Table 3. We also visualize the distribution of session scores in a 2-D scatter plot in Fig. 1. Here we set α to 2 for all datasets to discount tasks longer than 3 turns. From the table we can see that **YCap** and **GHome** perform overall better than the other two. All four assistants are able to achieve a satisfactory success rate, but **DSTC2** and **DSTC3** perform badly on efficiency. Specifically, among all the successful sessions ($SuccessRate = 1.0$), the ratio of tasks whose efficiency is less than 0.5 is more than 50% , but in **YCap** and **GHome** the percentage is less 20%. Since the system used in DSTC datasets is considerably dated, we think the high *Reformulation Rate* can be attributed to the poor ASR quality. What’s more, we can also use the metric to quickly identify problematic dialogues, i.e. the ones have low engagement scores. There are 35/63/9/13 sessions whose overall score is less than 0.2. After manually examining those sessions, we find the most prominent issues affecting **DSTC2** and **DSTC3** are poor ASR and language understanding ability. A user may repeat 5 times to make the system understand what the request is about. **YCap** only takes user commands matching particular templates and oftentimes users reform their request several times to make it accepted. In **GHome**, problems are more diverse since it supports various functions and users can ask open-domain questions to which the system cannot handle well yet.

The goal here is to demonstrate how metrics based on the proposed user engagement status could be used to evaluate system performance and troubleshoot failures, and these metrics can be easily adopted for online A/B testing.

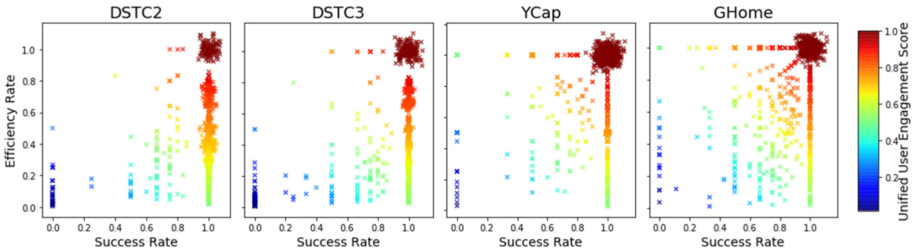


Fig. 1. 2-D scatter plot of user engagement metrics. A jitter is applied to show the size of clusters.

4 Automatic Prediction of User Engagement Status

Now we have defined a series of user engagement metrics for intelligent assistants, the next step is to automate the prediction of user engagement status so that the proposed metrics can be used in large-scale and online applications. We formalize this task as a four-class classification problem at the utterance-level.

4.1 Model Setting

We mainly examine two groups of models. The first group is classic classifiers, working together with hand-crafted features. We consider three models which are broadly used for text classification: Logistic Regression (**LR**), Support Vector Machine (**SVM**) and Random Forest (**RF**). The second group is convolutional neural networks (**CNN**), which learn continuous representations without manual feature engineering and allow us to leverage word vectors pretrained on a large corpus, with which a significant performance boost has been observed in various NLP studies. We use two variants of CNNs proposed by [23]: **CNN.Rand** and **CNN.MultiCh** (multi-channel). We have also tested a group of models based on recurrent neural networks, however they cannot converge well (may be due to the size of datasets). Thus their scores are not discussed.

4.2 Feature Setting

We think the status of user engagement is system-independent and identifiable by analyzing the dialogue contents. Therefore we only use features that can be extracted from transcriptions and ignore the other types of system-specific outputs (e.g., dialogue state, ASR output). From each utterance, we define six groups of features and use them to predict user engagement status. Besides, we notice that *Reformulation* implies a high semantic similarity between two user requests, thus we also define a set of *similarity features* for each feature group. We use '#feature_x'to denote the count of the feature.

- **Basic Features** Three subgroups of features indicating basic information of each utterance: (1) Utterance length: **utt_length**; (2) Time: **if_dialogue_start**, **if_dialogue_end**, **#utt_from_end**, **#utt_to_end**, **time_percent**; (3) Three features based on common user commands (e.g. "remind", "alarm", "add item"): **command_word** (one-hot vectors), **#command_word**, **jaccard_sim** (jaccard similarity between two adjacent user utterances).
- **Phrasal Features** We apply Stanford CoreNLP to extract 1) noun phrases (**noun_phrase**) and 2) entities (**entity**) from each utterance and represent them as one-hot vectors. We define three similarity features: 3) **repetition**: if any noun phrase/entity is repeated in two adjacent user utterances; 4) **#repetition**: number of repeated noun phrases/entities; 5) **jaccard_sim**.
- **Syntactic Features** The syntactic dependencies can help us understand the core components of utterances. From the dependency tree of each utterance, we can extract three types of syntactic features and represent them as one-hot vectors: 1) root word (**root_word**), 2) topmost subject word (**subject_word**) and 3) topmost object word (**object_word**). For similarity we only check if there is any repetition of these words between two user utterances: 4) **repeat_root_word**, 5) **repeat_subject_word** and 6) **repeat_object_word**.

- **N-grams Features** The n-grams is considered one of the most robust features for text classification. We extract 1-/2-/3-grams and represent them as one-hot vectors weighted by TfIdf. Two similarity features: 1) **edit_distance** (Levenshtein edit distance) and 2) **jaccard_sim**.
- **Topic Features** We apply the Latent Dirichlet Allocation (LDA) to capture the topical information in utterances (**lda_feature**). We train separate LDA model for each dataset and set its dimension to 50. We use the cosine similarity of LDA vectors between two user utterances (**lda_cosine**) as its similarity feature.
- **Distributed Representations** Previous studies [5,19,24,27] have demonstrated the efficacy of transferring language knowledge learned from rich resources to new tasks. Since we have only a limited amount of dialogues for training, we would like to know if we could utilize large text representation models to alleviate data shortage. Here we present three models to represent utterances: **Word2Vec** [32] (averaging word vectors in the utterance, dimension = 300), **Doc2Vec** [27] (treating each utterance as a document, dimension = 300) and **Skip-thought** [24] (using bi-skip model, dimension=2400).

4.3 Context Setting

The user engagement status greatly depends on the response from the system as well as the corresponding feedback of user. Previous studies have demonstrated the effects of contextual information in facilitating identification [3,22]. By comparing different settings of context, we are able to know which utterances are most effective for predicting user engagement. We denote five utterances in time order and define five settings of context as follows, covering different range of utterances in the dialogue:

- $user\_utt_{-1}$: previous user utterance,
- sys_utt_{-1}: previous system utterance,
- $user\_utt_0$: current user utterance,
- $user\_utt_{+1}$: next user utterance,
- sys_utt_{+1}: next system utterance.

- **CUR_UTT=**$\{user\_utt_0\}$,
- **CUR=**$\{user\_utt_0,\ sys\_utt_{+1}\}$,
- **NEXT=**$\{user\_utt_0,\ sys\_utt_{+1},\ user\_utt_{+1}\}$,
- **PREV=**$\{user\_utt_{-1},\ sys\_utt_{-1},\ user\_utt_0\}$,
- **ALL=**$\{user\_utt_{-1},\ sys\_utt_{-1},\ user\_utt_0,\ sys\_utt_{+1},\ user\_utt_{+1}\}$.

5 Results of Automatic Prediction

We conduct comprehensive experiments on four datasets to study the effects of different machine learning models, context ranges and feature settings. Specifically, we train and evaluate all models on each dataset using 10-fold cross-validation: 80%/10%/10% for training/validation/testing respectively. In order to perform significance tests on the relatively small datasets, we repeat the cross-validation five times, yielding 50 random splits and corresponding results. Unless otherwise stated, we report unweighted macro-average scores of 50 experiments on the testset. We apply two-sided paired T-test to examine the significance of changes. Besides, we also utilize the Bonferroni correction for T-test [2,37] to counteract the risk of using overlapping data partitions

5.1 Comparison of Models

We compare the performance of different models to get a general idea. We run experiments with the context range of **ALL** to include as many features as we can. All three classic classifiers are trained with *N-grams* features as well as similarity features. We report accuracy and F1-score, common metrics for classification tasks, of each model with optimal hyperparameters after a simple grid search, in Table 4.

Two simple baseline models are compared here, outputting the major class in the training set (Majority) or a random class uniformly (Random). Both simple baselines work poorly, and the F1-score of Majority is even lower due to the very skewed class distribution. The primary models perform fairly well. The two CNN models, without any human-designed feature, outperform all the other models in the current setting. The benefit of adopting pre-trained word vector is slight but significant ($p_{value}< 0.01$).

It shows comparative performance among three classic models. The SVM performs the best, but its advantage over LR is marginal ($p_{value} > 0.05$). Thus for the rest of this study, we only discuss the results of Logistic Regression, due to its advantage on interpretability of feature importance. Specifically, we use the LR with the L1 regularization ($\lambda = 1.0$), which performs robustly across different features and datasets.

Table 4. The averaged performance of user engagement classification with different models on four datasets (context=**All**, with similarity features). The underline indicates the maximum value in each column.

Model	Accuracy	F1-score
Majority	0.6020	0.1858
Random	0.2503	0.2029
SVM	0.8410	0.6440
LR	0.8398	0.6413
RF	0.8415	0.6192
CNN.Rand	0.8287	0.6549
CNN.MultiCh	0.8367	0.6674

Table 5. The comparison of user engagement classification without and with similarity features (context=**NEXT**). †/‡ indicates a significant change at p<0.05/p<0.01 between results with and without similarity features. The bold font/underline indicates the maximum value in the respective row/column.

Model	w\o Similarity	w\Similarity
Basic	0.3836	**0.4105** (+2.69%)
Phrasal	0.5913	**0.6316** (+4.03%)†‡
Syntactic	0.6078	**0.6280** (+2.02%)†‡
N-grams	0.6113	**0.6573** (+4.60%)†‡
Topic Model	0.5803	**0.6346** (+5.43%)†‡
Word2Vec	0.6162	**0.6521** (+3.59%)†‡
Doc2Vec	0.5858	**0.5968** (+1.10%)
Skip-thought	0.6063	**0.6216** (+1.53%)

Table 6. The performance (F1-score) comparison of user engagement classification with different context settings (without similarity features). †/‡ indicates a statistical significant difference at p<0.05/p<0.01 between **CUR_UTT** and **PREV** or between **NEXT** and **ALL**.

Model	CUR_UTT	CUR	NEXT	PREV	ALL
Basic	0.3425	0.3503	0.3836	0.3501†‡	**0.3963**†‡
Phrasal	0.3679	0.5521	**0.5913**	0.3709	0.5661†‡
Syntactic	0.3485	0.5530	**0.6078**	0.3671†‡	0.5867†‡
N-grams	0.3839	0.5694	**0.6113**	0.3788	0.5984†‡
Topic model	0.2982	0.5255	0.5803	0.3464†‡	**0.5829**
Word2Vec	0.3704	0.5723	**0.6162**	0.3827†‡	0.6032†‡
Doc2Vec	0.3427	0.5379	**0.5858**	0.3722†‡	0.5740†‡
Skip-thought	0.3648	0.5545	**0.6063**	0.3692	0.6008†
CNN.Rand	0.4252	0.5862	**0.6647**	0.4153	0.6549†
CNN.MultiCh	0.4207	0.5829	**0.6685**	0.4288	0.6674

5.2 Comparison of Context Settings

In this subsection, we investigate what context are most important for detecting user engagement status. We list the performance comparison with five context settings in Table 6. Note that, since there is no similarity feature for **CUR_UTT** and **CUR**, we exclude all similarity features for these experiments for a fair comparison.

Firstly, we see that, the score difference is consistent across different context settings, indicating that the context is a significant factor in engagement status prediction. **CUR_UTT** performs the worst among the five settings, since it includes only the content of the current user utterance and it provides very limited information. As for **CUR**, with one system utterance, the performance is remarkably better than the **CUR_UTT**. Furthermore, with the evident feedback from user ($user\_utt_{+1}$), **NEXT** performs generally the best among all context settings. This result conveys a clear message that, the following utterances from both system and user are critical in determining whether the next system response is relevant or not and whether the user is satisfied or not. As for **PREV** and **ALL**, which include the historical information of user requests, the performances are generally no better than the **CUR_UTT** and **NEXT** respectively. But this gap is smaller on distributed representations and models, especially for CNN. We speculate this is because most user requests can be satisfied within a few turns and do not require much historical information, thus the features from previous utterances rarely have an effect and even become detrimental.

5.3 Effects of Similarity Features

Based on the comparison of context settings, here we focus on analyzing the models with **NEXT** setting. We show the performances of Logistic Regression with and without similarity features in Table 5. By adding similarity features, which are just one or two additional features, the scores on different feature groups increase significantly. The similarity features are devised to facilitate detecting the reformulated utterances, and we observe that the average improvement on the *Reformulation* (8.06%) is much more salient than other three classes (2.99%, 1.74% and 0.51%). Feature importance analysis based on one-way ANOVA shows that similarities on *N-gram*, *LDA* and *Phrasal* features are most significant, which is consistent with the improvement in Table 5.

5.4 Analysis on Feature Groups

Furthermore, we apply another two techniques to explore better model performance: feature combination and feature selection. On one hand, the first four feature groups are discrete and capture various local linguistic information, while the rest four groups give continuous representations with regard to the whole utterance. Thus we consider combining these two sets of features and expect further improvement with the advantages of both. On the other hand, feature selection has been proved helpful in reducing noisy features. Here we apply *Chi-square statistic* to discrete feature groups and *Principal Component Analysis (PCA)* to continuous feature groups. We report the best performance of each setting in Table 7, after a simple grid search of hyperparameters.

Table 7. Scores of user engagement prediction with different features (context=**NEXT**, with similarity features). The right part presents F1-score of best models on each dataset. The underline indicates the best score in each column. The bold indicates the better score between models with and without feature selection. †/‡ indicates a statistical significant difference at $p < 0.05$/$p < 0.01$.

Model	w\o FeatSelect	w\FeatSelect		DSTC2	DSTC3	YCap	GHome
	w\Sim	w**o** Sim	w\Sim				
(a) Basic	0.4105	–	0.4105	0.5411	0.5044	0.3079	0.2886
(b) Phrasal	0.6316	–	**0.6318**	0.6470	0.6703	0.6593	0.5508
(c) Syntactic	0.6280	–	**0.6402†‡**	0.6567	0.6469	0.7005	0.5566
(d) N-grams	0.6573	–	**0.6770†‡**	0.7078	0.6905	0.6851	0.6248
(e) Topic model	0.6346	–	**0.6358**	0.6774	0.6384	0.6397	0.5877
(f) Word2Vec	0.6521	–	**0.6523**	0.6919	0.6919	0.6209	0.6043
(g) Doc2Vec	0.5968	—	**0.5969**	0.6325	0.6335	0.5730	0.5486
(h) Skip-thought	0.6216	–	**0.6216**	0.6654	0.6414	0.6020	0.5775
(i) (a) + (b) + (c) + (d)	0.6694	0.6511	**0.7085†‡**	0.7360	0.7151	0.7218	0.6613
(j) + Topic Model	0.6720	0.6617	**0.7152†‡**	0.7438	0.7161	0.7314	0.6699
(k) + Word2Vec	0.6790	0.6617	**0.7135†‡**	0.7514	0.7194	0.7180	0.6651
(l) + Doc2Vec	0.6713	0.6631	**0.7100†‡**	0.7390	0.7149	0.7269	0.6592
(m) + Skip-thought	0.6747	0.6666	**0.7124†‡**	0.7412	0.7181	0.7209	0.6696
(n) All	0.6825	0.6589	**0.7140†‡**	0.7490	0.7213	0.7202	0.6655
(o) CNN.Rand	0.6647	–	–	0.6798	0.6669	0.6943	0.6176
(p) CNN.MultiCh	0.6685	–	–	0.6880	0.6612	0.7054	0.6196

Overall, we observe that most models with feature selection outperform the original ones significantly. The feature selection works more significantly on groups having a large number of features such as *N-grams*, *Syntactic* and combined feature groups, indicating that only a small proportion of discrete features is actually in effect. Also the performances on combined feature groups (row **i** to **n**) are much better than on any of individual groups. But we observe that the continuous representations (**j-n**) contribute marginally on the top of the combined discrete features (**i**).

With the help of feature combination and selection, the Logistic Regression outpaces the previous best model CNN by a large margin. But if we exclude the similarity features (3rd column), we find that CNN still works on a par with the best LR models. Since the CNN models do not take any explicit input about similarity, the best LR models with similarity features beat CNN soundly. In order to let the CNN be aware of the user reformulation, we think it might be helpful to leverage a submodule for similarity calculation: train the submodule separately in a way like paraphrase identification [44], and take the similarity vector as additional input for classification.

Table 7 also presents detailed scores on each dataset after feature selection. One trend emerging among most LR results is that, the scores decrease gradually from **DSTC2** to **GHome**, implying the difficulty of each dataset. *LR+Basic* works well on **DSTC2** and **DSTC3** but poorly on the other two datasets. As we know, the *command_word* in *Basic* covers the most common user commands, and therefore it performs adequately in simple dialogues. But in more complicated cases, general words or linguistic components from both user and system sides become necessary, such as confirmations (ok, sure, yeah, etc.), success and failure signals (discard, sorry, don't understand, etc.), function-related words, and they are captured in different feature groups.

5.5 Analysis on Failure Cases

To understand better what shortcomings our models suffer from, we manually examine 50 random wrongly-predicted examples from **GHome** dataset and try to understand the reasons behind: *Reformulation* - 22 (examples), *Abandonment* - 21, *Fulfillment* - 4, *Continuation* - 3. The highly skewed class distribution might be one major reason. The model is trained with very few examples of *Reformulation* and *Abandonment*, therefore it is more prone to make more mistakes on them. We also notice some issues that are general to all dialogue related tasks, which might be difficult to overcome with NLP techniques used in this study: (1) A common error (16 times) is that the model cannot distinguish whether a system response is relevant to a user's request or not. Our models can only determine the relevance by feature matching instead of understanding the actual semantics, particularly when the user request is long or task-general. (2) 15 examples that require considering contextual and historical information. For example, a user asks Google Home to "Turn the Christmas tree off" and "Turn it on", our model does not recognize "it" refers to the previous "Christmas tree". Another long-dependency case is, the system confirms a similar question after a few turns, which should be treated as *Reformulation*, but this can be hardly addressed by current models. (3) The third common mistake is specific to *Reformulation*, which occurs 9 times. On one hand, a user may paraphrase an utterance in a different way to help the system understand, such as from “I want the stair lights” to “turn on the stair lights”. On the other hand, a user can also issue two apparently similar but different requests, say “how skinny is my husband” and “how old is my husband”. A more powerful semantic encoder [7] might be helpful in this case.

6 Conclusion and Future Work

In a preliminary effort to solve the challenging problem of online evaluation for large-scale intelligent assistants, we provide a practicable solution, by converting the problem into a more tractable classification task and automating it with various machine learning methods. We admit there is still a long way to improve our model to work well in real environments. Also, more research is in urgent need to bridge the gap between utterance-level user engagement status and task-level user experience. Thus, for future research, we will first apply online A/B testing to validate whether any of proposed utterance-level user engagement status and metrics correlates well with the real long-term success. We believe with insights from these studies, we can understand user experience with intelligent assistants better and design better evaluation methods.

References

1. Aliannejadi, M., Zamani, H., Crestani, F., Croft, W.B.: Asking clarifying questions in open-domain information-seeking conversations. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 475–484 (2019)
2. Armstrong, R.A.: When to use the bonferroni correction. Ophthalmic Physiol. Optics **34**(5), 502–508 (2014)
3. Bangalore, S., Di Fabbrizio, G., Stent, A.: Learning the structure of task-driven human-human dialogs. IEEE Trans. Audio, Speech Lang. Process. **16**(7), 1249–1259 (2008)
4. Chowdhury, S.A., Stepanov, E.A., Riccardi, G.: Predicting user satisfaction from turn-taking in spoken conversations. Interspeech **2016**, 2910–2914 (2016)
5. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011)
6. Deng, A., Shi, X.: Data-driven metric development for online controlled experiments: seven lessons learned. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 77–86 (2016)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)

8. Diriye, A., White, R., Buscher, G., Dumais, S.: Leaving so soon?: understanding and predicting web search abandonment rationales. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 1025–1034. ACM (2012)
9. Graepel, T., Candela, J.Q., Borchert, T., Herbrich, R.: Web-scale Bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In: Proceedings of the 27th International Conference on International Conference on Machine Learning, pp. 13–20 (2010)
10. Griol, D., Callejas, Z.: A neural network approach to intention modeling for user-adapted conversational agents. Comput. Intell. Neurosci. **2016**, 44 (2016)
11. Hara, S., Kitaoka, N., Takeda, K.: Estimation method of user satisfaction using n-gram-based dialog history model for spoken dialog system. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010) (2010)
12. Hashemi, S.H., Williams, K., El Kholy, A., Zitouni, I., Crook, P.A.: Measuring user satisfaction on smart speaker intelligent assistants using intent sensitive query embeddings. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 1183–1192 (2018)
13. Hassan, A., Jones, R., Klinkner, K.L.: Beyond DCG: user behavior as a predictor of a successful search. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 221–230. ACM (2010)
14. Hassan, A., Shi, X., Craswell, N., Ramsey, B.: Beyond clicks: query reformulation as a predictor of search satisfaction. In: Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, pp. 2019–2028. ACM (2013)
15. Hassan, A., Song, Y., He, L.W.: A task level metric for measuring web search satisfaction and its application on improving relevance estimation. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, pp. 125–134. ACM (2011)
16. Henderson, M., Thomson, B., Williams, J.D.: The second dialog state tracking challenge. In: Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pp. 263–272 (2014)
17. Henderson, M., Thomson, B., Williams, J.D.: The third dialog state tracking challenge. In: 2014 IEEE Spoken Language Technology Workshop (SLT), pp. 324–329. IEEE (2014)
18. Higashinaka, R., Funakoshi, K., Kobayashi, Y., Inaba, M.: The dialogue breakdown detection challenge: task description, datasets, and evaluation metrics. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), pp. 3146–3150 (2016)
19. Hill, F., Cho, K., Korhonen, A.: Learning distributed representations of sentences from unlabelled data. In: 15th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016, pp. 1367–1377. Association for Computational Linguistics (ACL) (2016)
20. Jiang, J.E.A.: Automatic online evaluation of intelligent assistants. In: Proceedings of the 24th WWW, pp. 506–516. International World Wide Web Conferences Steering Committee (2015)
21. Kamm, C.: User interfaces for voice applications. Proc. Natl. Acad. Sci. **92**(22), 10031–10037 (1995)

22. Kim, S.N., Cavedon, L., Baldwin, T.: Classifying dialogue acts in one-on-one live chats. In: Proceedings of the 2010 Conference on EMNLP, pp. 862–871. Association for Computational Linguistics (2010)
23. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751 (2014)
24. Kiros, R.E.A.: Skip-thought vectors. In: Advances in Neural Information Processing Systems, pp. 3294–3302 (2015)
25. Kiseleva, J., Williams, K., Hassan Awadallah, A., Crook, A.C., Zitouni, I., Anastasakos, T.: Predicting user satisfaction with intelligent assistants. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 45–54. ACM (2016)
26. Krahmer, E., Swerts, M., Theune, M., Weegels, M.: Error detection in spoken human-machine interaction. Int. J. Speech Technol. **4**(1), 19–30 (2001)
27. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31st International Conference on Machine Learning (ICML-14), pp. 1188–1196 (2014)
28. Li, J., Galley, M., Brockett, C., Gao, J., Dolan, B.: A diversity-promoting objective function for neural conversation models. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 110–119 (2016)
29. Liu, C.W., Lowe, R., Serban, I.V., Noseworthy, M., Charlin, L., Pineau, J.: How not to evaluate your dialogue system: an empirical study of unsupervised evaluation metrics for dialogue response generation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2122–2132 (2016)
30. Lowe, R., Noseworthy, M., Serban, I.V., Angelard-Gontier, N., Bengio, Y., Pineau, J.: Towards an automatic turing test: learning to evaluate dialogue responses. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1116–1126 (2017)
31. Meena, R., Lopes, J., Skantze, G., Gustafson, J.: Automatic detection of miscommunication in spoken dialogue systems. In: Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 354–363 (2015)
32. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
33. Ohtake, K.: Unsupervised approach for dialogue act classification. In: Proceedings of the 22nd Pacific Asia Conference on Language, Information and Computation, pp. 445–451 (2008)
34. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 311–318 (2002)
35. Polifroni, J., Hirschman, L., Seneff, S., Zue, V.: Experiments in evaluating interactive spoken language systems. In: Proceedings of the workshop on Speech and Natural Language, pp. 28–33. Association for Computational Linguistics (1992)
36. Ritter, A., Cherry, C., Dolan, W.B.: Data-driven response generation in social media. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 583–593 (2011)
37. Salzberg, S.L.: On comparing classifiers: pitfalls to avoid and a recommended approach. Data Mining Knowl. Discov. **1**(3), 317–328 (1997)

38. Shriberg, E., Wade, E., Price, P.: Human-machine problem solving using spoken language systems (sls): factors affecting performance and user satisfaction. In: Proceedings of the Workshop on Speech and Natural Language, pp. 49–54. Association for Computational Linguistics (1992)
39. Sordoni, A., et al.: A neural network approach to context-sensitive generation of conversational responses. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 196–205 (2015)
40. Vinyals, O., Le, Q.V.: A neural conversational model. In: ICML Deep Learning Workshop (2015). http://arxiv.org/pdf/1506.05869v3.pdf
41. Walker, M.A., Litman, D.J., Kamm, C.A., Abella, A.: Paradise: a framework for evaluating spoken dialogue agents. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, pp. 271–280 (1997)
42. Yang, Z., Li, B., Zhu, Y., King, I., Levow, G., Meng, H.: Collaborative filtering model for user satisfaction prediction in spoken dialog system evaluation. In: Spoken Language Technology Workshop (SLT), 2010 IEEE, pp. 472–477. IEEE (2010)
43. Yi, X., Hong, L., Zhong, E., Liu, N.N., Rajan, S.: Beyond clicks: dwell time for personalization. In: Proceedings of the 8th ACM Conference on Recommender Systems, pp. 113–120. ACM (2014)
44. Yin, W., Schütze, H.: Convolutional neural network for paraphrase identification. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 901–911 (2015)

Drug and Disease Interpretation Learning with Biomedical Entity Representation Transformer

Zulfat Miftahutdinov^(✉), Artur Kadurin, Roman Kudrin,
and Elena Tutubalina

Insilico Medicine Hong Kong, Pak Shek Kok, Hong Kong
{zulfat,artur,kudrin,elena}@insilico.com
https://insilico.com/

Abstract. Concept normalization in free-form texts is a crucial step in every text-mining pipeline. Neural architectures based on Bidirectional Encoder Representations from Transformers (BERT) have achieved state-of-the-art results in the biomedical domain. In the context of drug discovery and development, clinical trials are necessary to establish the efficacy and safety of drugs. We investigate the effectiveness of transferring concept normalization from the general biomedical domain to the clinical trials domain in a zero-shot setting with an absence of labeled data. We propose a simple and effective two-stage neural approach based on fine-tuned BERT architectures. In the first stage, we train a metric learning model that optimizes relative similarity of mentions and concepts via triplet loss. The model is trained on available labeled corpora of scientific abstracts to obtain vector embeddings of concept names and entity mentions from texts. In the second stage, we find the closest concept name representation in an embedding space to a given clinical mention. We evaluated several models, including state-of-the-art architectures, on a dataset of abstracts and a real-world dataset of trial records with interventions and conditions mapped to drug and disease terminologies. Extensive experiments validate the effectiveness of our approach in knowledge transfer from the scientific literature to clinical trials.

Keywords: Clinical trials · Natural language processing · Neural networks · Entity linking · Medical concept normalization · Metric learning · Negative sampling · Bert

1 Introduction

The emerging use of neural network architectures in the early-stage of drug discovery has recently resulted in several breakthroughs [20,50]. Later stages of drug development are much more conservative due to the complicated process of clinical trials. The use of state-of-the-art neural network approaches in clinical trials could dramatically speed up the overall drug development process and increase its success rate, thus saving lives.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 451–466, 2021.
https://doi.org/10.1007/978-3-030-72113-8_30

Clinical trial registers (e.g., ClinicalTrials.gov) contain vast amounts of structured information on how standardized interventions work in a clinical setting. Despite the existing structure, these registers remain very difficult to harmonize with drug and disease databases using current techniques. This very often results in substantial information losses. The primary cause for this inaccurate harmonization is that in a clinical trial record diseases and interventions are not described with a centralized standardized taxonomy but with a free text. The automatic natural language processing (NLP) methods are promising approaches for the semantic annotation of large volumes of clinical records and for the integration and standardization of biomedical entity mentions to formal concepts. In biomedical research and healthcare, the entity linking problem is known as medical concept normalization (MCN). A source as a knowledge base (KB) contains further information about the concept, such as its preferred name and synonyms, pharmacological profile, and its relationships with other concepts.

Neural architectures have been widely used in recent state-of-the-art models for MCN from user reviews and social media texts [22,25,31,44,49,51]. These studies mostly share limitations regarding a supervised classification framework: binary or multiclass classifiers are trained on a dataset with a narrow subsample of concepts from a specific terminology. In particular, recent models [22,49,51] learn a scoring function measuring the similarity between an entity mention and a concept. The difficulty with these methods is that it is not possible to extract representations describing mentions and concepts separately. In this setup, to retrieve concepts from a particular terminology for a given entity mention, we have to compute all the similarities through the ranking function and sort these scores in descending order. This is impractical if we need to process large corpora of free-form clinical trials, scientific literature, patents in days.

Inspired by metric learning [16,18,38], its usage for multimodal and sentence representation learning [28,37], negative sampling [32], and Bidirectional Encoder Representations from Transformers (BERT) [10], we present a BERT-based neural model for medical concept normalization that directly optimizes the BioBERT representations [23] of entity mentions and concept names itself, rather than classification or ranking layer. We use triplets of free-form entity mention, positive concept names, and randomly sampled concept names as negative examples to train our model. In this work, we consider the zero-shot scenario because it is often the case in the biomedical domain, where there are dozens of concept categories and terminologies. We trained models on annotated pairs of disease or chemical mentions with the corresponding concepts and evaluated on a novel dataset of condition and intervention concepts from clinical trials.

The contributions of this paper can be summarized as follows:

1. We develop a simple and effective model that uses metric learning and negative sampling to obtain entity and concept embeddings. These embeddings were utilized for knowledge transfer between different terminologies. We explore several strategies to select positive and negative samples.

2. We perform extensive experiments of several BERT-based models on a newly annotated dataset of clinical trials in two setups, where each mention is associated with one or more concepts (in-KB) or zero (out-of-KB).

2 Related Work

Our work most closely relates to research in information extraction and semantic textual similarity by directly linking a set of entity mentions and a large set of medical concept names using triplet structures to derive embeddings of entity mentions and concept names that can be compared using semantic similarity. Entity linking of mentions to entries in a knowledge base (KB) is a well-studied area; see a good survey [40]. Research studies in this area assume that there is one knowledge base, such as Wikipedia or Freebase. The KB contains rich text descriptions (from an entity page, for example), hyperlink statistics, and metadata. This assumption holds for the general domain, but not for the biomedical domain, where diverse terminologies exist for numerous purposes.

2.1 Medical Concept Normalization

Medical concept normalization is usually formulated as a classification or ranking problem with a wide variety of features – syntactic and morphological parsing, dictionaries of medical concepts and their synonyms, distances between raw entity mentions and formal concept names in terms of TF-IDF or word2vec representations [1,9,12,22,45]. MetaMap is one of the most well-known knowledge-based systems for mapping texts to concepts from Unified Medical Language System (UMLS) [3] developed by the US National Library of Medicine (NLM) [1]. This system is based on a linguistic approach using lexical lookup and variants by associating a score with phrases in a sentence. The NLM provides automatic indexing of clinical trials to Medical Subject Headings (MeSH) [6] via the Medical Text Indexer (MTI) [33] based on MetaMap. MTI achieves an F1 measure around 0.55 on the indexing of PubMed abstracts. The most popular open-source supervised system maintained by the NLM is TaggerOne [22]. TaggerOne utilizes semi-Markov models with features and dictionaries to jointly perform entity extraction and normalization tasks.

The works that are the closest to ours and consider synonyms during entity and concept representation learning is Biomedical Named Encoder (BNE) [35] and BioSyn [41]. Sung et al. proposed a BioBERT-based model named BioSyn that maximizes the probability of all synonym representations in the top 20 candidates [41]. BioSyn uses a combination of two scores, sparse and dense, as a similarity function. Sparse scores are calculated on character-level TF-IDF representations to encode morphological information of given strings. Dense scores are defined by the similarity between `CLS` tokens of a single vector of input in BioBERT. This model achieves state-of-the-art results in disease and chemical mapping over previous works [22,35,47]. Phan et al. presented an encoding framework with new context, concept, and synonym-based objectives [35].

Synonym-based objective enforces similar representations between synonymous names, while concept-based objective pulls the name's representations closer to its concept's centroid. However, ranking on these embeddings shows worse results on three sets than TaggerOne.

Our work differs from the studies discussed above in the following important aspects. First, none of these methods have been applied to free-form descriptions of conditions and interventions from clinical trials. Second, evaluation strategies in the mentioned papers are based on train/test splits provided by datasets' authors. We follow the recent *refined* evaluation strategy from [43] on the creation of test sets without duplicates or exact overlaps between the train and test sets. Finally, our dataset includes entity mentions for both in-KB and out-of-KB linking.

2.2 NLP in Clinical Trials Research

While the majority of biomedical research on information extraction primarily focused on scientific literature [17], much less work had been used NLP methods to conduct curation of clinical trial records' fields to advance downstream tasks [2,4,5,11,15,39]. Gayvert et al. [11] proposed an approach for the prediction of the likelihood of toxicity in clinical trials. They selected 108 clinical trials of any phase that were annotated as having failed for toxicity reasons. Then intervention names of each trial were manually mapped to DrugBank [46] concepts to collect molecular weight, polar surface area, and other compounds' properties. In [2], Atal et al. developed a knowledge-based approach to classify entity mentions to disease categories from a Global Burden of Diseases (GBD) cause list. The proposed method uses MetaMap to extract UMLS concepts from trial fields (health condition, public title, and scientific title), link UMLS concepts with ICD10 codes, and classify ICD10 codes to candidate GBD categories. The developed classifier identified GBD categories for 78% of the trials. Li and Lu [26] identified clinical pharmacogenomics (PGx) information from clinical trial records based on dictionaries from a pharmacogenomics knowledge base PharmGKB. Previous studies on clinical trial records, however, have not analyzed the performance of linking of clinical trials to disease and drug concepts, but rather across eligibility criteria (e.g., patient's demographic, disease category) [2,4,15,24,39].

3 Dataset of Clinical Trials

NLM maintains a clinical trial registry data bank ClinicalTrials.gov[1] that contains over 340,000 trials from 214 countries. This database includes comprehensive scientific and clinical investigations in biomedicine [13]. Each trial record provides information about a trial's title, purpose, description, condition, intervention, eligibility, sponsors, etc. Most information from records is described in

[1] https://clinicaltrials.gov/.

natural language. In our study, we use publicly available American Association of Clinical Trials (AACT) Database[2], v. 20200201.

Since there is no off-the-shelf manually annotated dataset for biomedical concept normalization of clinical trials, we built one by selecting 500 trials using the following criteria:

1. A type of clinical study is an interventional study. Participants of interventional studies receive intervention/treatment so that researchers can evaluate the effects of the interventions on biomedical or health-related outcomes [29].
2. Phase of clinical study is defined by U.S. Food and Drug Administration (FDA). There are five phases: Early Phase 1, Phase 1, Phase 2, Phase 3, and Phase 4.
3. Clinical study is associated with one or more interventions of the following types: Biological, Combination Product, Drug.

As a drug terminology source, we use an internal knowledge base that contains 15,532 concept unique identifiers (CUIs), including small molecule drugs, biologics, nutraceuticals, and experimental drugs. As a condition terminology source, we use MeSH v. 20200101. 500 selected trials contain 1075 and 819 entries in the 'Intervention' and 'Condition' fields respectively. Two annotators with a background in bioinformatics manually annotated each entry. The calculated inter-annotator agreement (IAA) using Kappa was 92.32% for the entire dataset. The disagreement was resolved through mutual consent.

Statistics of annotated texts are summarized in Table 1. 794 out of 1075 non-unique mentions (73.9%) were mapped to one or more drug concepts. 838 (80%) of lower-cased interventions are unique. 804 out of 819 non-unique mentions (98.2%) were mapped to one or more concepts, while there are 638 (78%) lower-cased unique mentions. Interestingly, MeSH concepts linked to conditions belong to several MeSH categories including Diseases [C], Psychiatry and Psychology [F], and Analytical, Diagnostic and Therapeutic Techniques, and Equipment [E]. We note that NLM provided automatically assigned MeSH terms to trials' interventions. 716 out of 1075 entries (66.6%) were mapped to MeSH terms. Our analysis revealed that mapping from NLM does not include investigational drugs, which are essential for developing new pharmaceutical drugs. Table 2 contains a sample of annotated texts.

4 Model

In this section, we present a neural model for Drug and disease Interpretation Learning with Biomedical Entity Representation Transformer (DILBERT). We address MCN as a retrieval task by fine-tuning the BERT-based network using metric learning [16,18,38], negative sampling [32], specifically, triplet constraints. This idea was successfully applied to learn multimodal embeddings [28,48] and recent sentence embeddings via a sentence-BERT model [37]. Compared to a pair

[2] https://www.ctti-clinicaltrials.org/aact-database.

Table 1. Statistics of annotated texts.

Mention	#texts	#texts with CUIs	#unique texts	#unique texts with CUIs
Intervention types				
Drug	850	693	671	585
Biological	118	90	102	79
Other	57	4	27	4
Procedure	19	1	16	1
Radiation	11	0	9	0
Device	11	1	11	1
Combination product	5	3	5	3
Dietary supplement	2	2	2	2
Diagnostic test	1	0	1	0
Behavioral	1	0	1	0
Total				
Intervention	1075	794	838	671
Condition	819	804	638	638

of independent sentences or images, two concept names can have relationships as synonyms, hypernyms, hyponyms, etc., that we consider during the training phase to facilitate the concept ranking task at the retrieval phase.

Let us first recall two terms: *concept* and *concept name*. Following the UMLS Glossary [34], the concept is the fundamental unit of meaning in terminology. It represents a single meaning in any way, whether formal or casual, verbose or abbreviated. Every concept is assigned a unique identifier (CUI). A concept consists of atoms, which are the smallest units of naming. All of the atoms within a concept are synonymous. The concept name is a string chosen to represent the concept as a whole. It is linked to atoms. Formally, the medical concept normalization task aims to assign each entity mention m a CUI (or predicts that there is no corresponding concept).

Architecture Following denotations proposed by [19], we encode both entity mention m and candidate concept name c into vectors:

$$y_m = red(T(m)); y_c = red(T(c)) \tag{1}$$

where T is the transformer that is allowed to update during fine-tuning. $red(\cdot)$ is a function that reduces that sequence of vectors into one vector. There are two main ways of reducing the output into one representation via $red(\cdot)$: choose the first output of T (corresponding to the token CLS) or compute the elementwise average over all output vectors to obtain a fixed-size vector. As a pretrained transformer model, we use BioBERT base v1.1. [23]

Scoring The score of a candidate c_i for an entity mention m is given by a distance metric, e.g. Euclidean distance:

$$s(m, c_i) = ||y_m - y_{c_i}|| \tag{2}$$

A noteworthy aspect of the proposed model is its scope: by design, it aims at the cross-terminology mapping of entity mentions to a given lexicon without additional re-training. This approach allows for fast, real-time inference, as all concept names from a terminology can be cached. This is a necessary requirement for processing biomedical documents of different subdomains such as clinical trials, scientific literature and drug labels.

Table 2. Sample of manually annotated trials' texts.

NCT/Type	Text	Concept
Intervention (with DrugBank CUIs)		
NCT00559975/Biological	Adjuvanted influenza vaccine combine with CpG7909	Agatolimod sodium (DB15018)
NCT01575756/Biological	Haemocomplettan® P or RiaSTAPTM	Fibrinogen human (DB09222)
NCT00081484/Drug	epoetin alfa or beta	Erythropoietin (DB00016)
NCT03375593/Drug	Ibuprofen 600 mg tab	Ibuprofen (DB01050)
NCT01170442/Drug	vitamin D3 5000 IU	Calcitriol (DB00136)
NCT02493335/Drug	Placebo orodispersible tablet twice daily	*nil (no concept)*
Condition (with MeSH CUIs)		
NCT02009605	Squamous Cell Carcinoma of Lung	Carcinoma, Non-Small-Cell Lung (D002289)
NCT04169763	Stage IIIC Vulvar Cancer AJCC v8	Vulvar Neoplasms (D014846)

Optimization The network is trained using a triplet objective function. Given a user-generated entity mention m, a positive concept name c_g and a negative concept name c_n, triplet loss tunes the network such that the distance between m and c_g is smaller than the distance between m and c_n. Mathematically, we minimize the following loss function:

$$max(s(m, c_g) - s(m, c_n) + \epsilon, 0) \tag{3}$$

where ϵ is margin that ensures that c_g is at least ϵ closer to m than c_n. As a scoring metric, we use Euclidean distance or cosine similarity and we set $\epsilon = 1$ in our experiments.

Positive and Negative Sampling Suppose that a pair of the entity mention with the corresponding CUI is given as well as the vocabulary. For positive examples, vocabulary is restricted to the concepts that have the same CUI as a mention. Multiple positive concept names could be explained by the presence of synonyms in the vocabulary. Negative sampling [32] uses the rest part of the vocabulary. We explore several strategies to select positive and negative samples for a training pair (entity mention, CUI):

1. **random sampling**: we sample several concept names with the same CUI as positive examples and random negatives from the rest of the vocabulary;
2. **random + parents**: we sample k concept names from the concept's parents in addition to positive and negative names gathered with the random sampling strategy;
3. **re-sampling**: using a model trained with random sampling, we identify positives and *hard* negatives via the following steps: (i) encode all mentions and concept names found in training pairs using the current model (ii) select positives with the same CUI, which are closest to a mention, (iii) for each mention, retrieve the most similar k concept names (i.e., its nearest neighbors) and select all names that are ranked above the correct one for the mention as negative examples. We follow this strategy from [14];
4. **re-sampling + siblings**: we modify the re-sampling strategy by using k concept names from the concept's siblings as negatives.

Inference At inference time, the representation for all concept names can be precomputed and cached. The inference task is then reduced to finding the closest concept name representation to entity mention representation in a common embedding space.

5 Experiments

We evaluate our model DILBERT and compare it to the state-of-the-art methods using (i) a publicly available benchmark BioCreative V CDR Disease & Chemical [27], (ii) our dataset of clinical trials named CT Condition & Intervention. The statistics of the two datasets are summarized in Table 3.

5.1 Datasets

BioCreative V CDR [27] introduces a challenging task for the extraction of chemical-disease relations (CDR) from PubMed abstracts. Disease and chemical mentions are linked to the MEDIC [8] and CTD [7] dictionaries, respectively. We utilize the CTD chemical dictionary (v. November 4, 2019) that consists of pf 171,203 CUIs and 407,247 synonyms, and the MEDIC lexicon (v. July 6, 2012) that contains 11,915 CUIs and 71,923 synonyms.

According to the BioCreative V CDR annotation guidelines, the annotators used two MeSH branches to annotate entities: (i) "Diseases" [C], including signs and symptoms, (ii) "Drugs and Chemicals" [D]. The terms "drugs" and "chemicals" are often used interchangeably. Annotators annotated chemical nouns convertible to single atoms, ions, isotopes, pure elements and molecules (e.g., calcium, lithium), class names (e.g., steroids, fatty acids), small biochemicals, synthetic polymers.

As shown in [43], the CDR dataset contains a high amount of mention duplicates and overlaps between official sets. In order to obtain more realistic results, we evaluate models on preprocessed official and *refined* CDR test sets from [43].

For the preprocessing of the clinical trial data, we use heuristic rules to split the composite mentions into separate mentions (e.g., *combination of ribociclib + capecitabine* into *ribociclib* and *capecitabine*) by considering each mention containing "combination", "combine", "combined", "plus", "vs" or "+" as composite. We process all characters to lowercase forms and remove the punctuation for both mentions and synonyms.

Table 3. Statistics of the datasets used in the experiments. Two sets of annotated clinical trials' fields are marked with 'CT'.

	CDR Disease	CDR Chem	CT Condition	CT Intervention
Domain	Abstracts	Abstracts	Clinical trials	Clinical trials
Entity type	Disease	Chemicals	Conditions	Drugs
Terminology	MEDIC	CTD Chemicals	MeSH	In-house dict.
entity level statistics				
% numerals	0.11%	7.32%	7.69%	25.3%
% punctuation	1.21%	0.07%	14.28%	24.83%
avg. len	14.88	11.27	17.92	21.68
number of pre-processed entity mentions				
Train set	4,182	5,203	—	–
Dev set	4,244	5,347	100	100
test set	4,424	5,385	719	975
number of pre-processed entity mentions after removal of duplicates from test set				
refined test	657 (14.9%)	425 (7.9%)	638 (77.89%)	838 (77.95%)

It is assumed that each entity mention in the CDR corpus has a valid concept in the terminology, which is referred as in-KB evaluation in the entity linking task. In contrast with the CDR sets, 26% and 1.8% of intervention and condition mentions in the CT dataset are not appeared in terminologies, respectively. In Sect. 5.4, we investigate different strategies for the out-of-KB prediction (i.e. *nil* prediction) on clinical trials' texts.

5.2 Baseline Methods

We compare our proposed method with the following methods.

BioBERT ranking This is a baseline model that used the BioBERT model for encoding mention and concept representations. Each entity mention or concept name is firstly passed through BioBERT (we use the average over all outputs of BERT) and then through a mean pooling layer to yield a fixed-sized vector. The inference task is then reduced to finding the closest concept name representation to entity mention representation in a common embedding space. We use the

Euclidean distance as the distance metric. The nearest concept names are chosen as top-k concepts for entities. We use the publicly available code provided by [43] at https://github.com/insilicomedicine/Fair-Evaluation-BERT.

BioSyn BioSyn [41] is a recent state-of-the-art model that utilizes the synonym marginalization technique and the iterative candidate retrieval. The model uses two similarity functions based on sparse and dense representations, respectively. The sparse representation encodes the morphological information of given strings via TF-IDF, the dense representation encodes the semantic information gathered from BioBERT. For reproducibility, we use the publicly available code provided by the authors at https://github.com/dmis-lab/BioSyn. We follow the default parameters of BioSyn as in [41]: the number of top candidates k is 20, the mini-batch size is 16, the learning rate is 1e–5, the dense ratio for the candidate retrieval is 0.5, 20 epochs for training.

5.3 Experimental Setup

We experiment with BioBERT$_{base}$ v1.1 with 12 heads, 12 layers, 768 hidden units per layer, and a total of 110M parameters. Epsilon, the number of positive and negative examples, and distance metric were chosen optimally on dev sets. We choose $red(\cdot)$ to be the average over all outputs of BERT. We have evaluated different epsilons starting from 0.5 up to 4.0 with 0.5 step for Euclidean distance metric, for cosine distance from 0.05 up to 0.3 with 0.05 step. These experiments have quite similar results. We have evaluated a number of positive and negative examples. For positives, we iterated over values from 15 to 35, for negatives from 5 to 15. We found that the optimal is to sample 30 positive examples and 5 negative examples per mention. For the random + parents strategy, we evaluated the number of names of concept's parents from 1 to 5. Similarly, we evaluated the number of names of concept's siblings from 1 to 5. We found that hard negative sampling (with siblings) achieves the same optima as random negative sampling. The highest metrics are achieved at 5 concept names of the concept's parents on the CT Condition and CDR Chemical sets. The highest accuracy is achieved at 2 names of the concept's parents on other sets. As a result, we trained the DILBERT model with Euclidean distance and the following parameters: batch size is equal to 48, learning rate was set to 1e–5, epsilon to 1.0.

We evaluate this solution in information retrieval (IR) scenario, where the goal is to find within a dictionary of concept names and their identifiers the top-k concepts for every entity mention in texts. In particular, we use the top-k accuracy as an evaluation metric, following the previous works [35,36,41–43,47]. Let Acc@k be 1 if a right CUI is retrieved at rank k, otherwise 0. All models are evaluated with Acc@1. For composite entities, we define Acc@k as 1 if each prediction for a single mention is correct.

5.4 Out-of-KB Cases in Clinical Trials

To deal with *nil* predictions in clinical trials, we apply three different strategies for the selection of a threshold value. Namely, the intervention or condition mention is considered out of KB if the nearest candidate has a larger distance than a threshold value. Our first strategy is to set the threshold equal to the minimum distance of false-positive (FP) cases. In this case, we consider a mention mapped to a concept by our model but having no appearance in the terminology. Our second strategy set the threshold to the maximum distance of true-positive (TP) cases. The third strategy uses a weighted average of the first two threshold values. The proportion of FP cases used as a weight for the first strategy's threshold, the proportion of TP cases used as a weight for the second strategy's threshold. We tested three strategies on the dev set which containing 100 randomly selected mentions and evaluated the selected threshold values on the test set. This procedure was repeated 20 times. For intervention normalization, the first strategy showed an average accuracy of 79.41 with std of 3.5; second – accuracy of 71.77 and std of 3.5; third – accuracy of 85.73, std of 1.3.

Table 4. Out-of-domain performance of the proposed DILBERT model and baselines in terms of Acc@1 on the *refined* test set of clinical trials (CT).

Model	CT Condition		CT Intervention	
	Single concept	Full set	Single concept	Full set
BioBERT ranking	72.60	71.74	78.67	74.57
BioSyn	86.36	–	86.29	–
DILBERT, random sampling	85.73	84.85	90.23	**88.37**
DILBERT, random + 2 parents	86.74	86.36	**90.53**	87.94
DILBERT, random + 5 parents	**87.12**	**86.74**	89.54	87.15
DILBERT, resampling	85.22	84.63	89.83	87.28
DILBERT, resampling + 5 siblings	84.84	84.26	89.26	86.23

Table 5. In-domain performance of the proposed DILBERT model in terms of Acc@1 on the *refined* test set of the Biocreative V CDR corpus.

Model	CDR Disease	CDR Chemical
BioBERT ranking	66.4	80.7
BioSyn	74.1	**83.8**
DILBERT, random sampling	75.5	81.4
DILBERT, random + 2 parents	75.0	81.2
DILBERT, random + 5 parents	73.5	81.4
DILBERT, resampling	**75.8**	83.3
DILBERT, resampling + 5 siblings	75.3	82.1

5.5 Results and Discussion

We investigate the effectiveness of transferring concept normalization from the general biomedical domain to the clinical trial domain. We trained DILBERT and BioSyn models on the CDR Disease and CDR Chemical train sets, respectively, for linking clinical conditions and interventions.

Table 4 presents the performance of the DILBERT models compared to BioSyn and BioBERT ranking on the datasets of clinical trials. We test the DILBERT model's transferability on two sets of interventions and conditions where each mention is associated with one concept only (see 'single concept' columns). We evaluate the model on test sets with all mentions, including single concepts, composite mentions, and out-of-KB cases (see 'full set' columns). In Table 5, we present in-domain results of models evaluated on the CDR data. In all our experiments when comparing DILBERT and BioSyn models, we use paired McNemar's test [30] with a confidence level at 0.05 to measure statistical significance.

Several observations can be made based on Tables 4 and 5. First, DILBERT outperformed BioSyn and BioBERT ranking on three sets staying on par with BioSyn on the CDR Chemical test set. Adding randomly sampled positive examples from parent-child relationships gives a statistically significant improvement in 1–2% on the CT Condition set while staying on par with random sampling on interventions. To our surprise, hard negative mining produces performance gains on one of four sets only, which includes chemicals. Second, we compare results on refined test sets with results on the CDR corpus's official test set. We observe the significant decrease of Acc@1 from 93.6% to 75.8% and from 95.8% to 83.8% for DILBERT on disease and chemical mentions, respectively. Third, DILBERT models obtained higher results on test sets with single concepts. Models achieve much higher performance for the normalization of interventions rather than conditions. The DILBERT model achieves a statistically significant improvement compared to the BioSyn model on the interventions dataset. The error analysis on the CDR Disease set showed that models with random negative sampling incorrectly maps 39 out of 147 mentions to the correct concept's parent. We observe that some mentions are mapped to the gold concept' child for the models trained by re-sampling+siblings sampling.

Inference Time Efficiency and Deployment Our model uses the FAISS library [21] with GPU support for fast nearest neighbor search by comparing vectors with Euclidean distance. Embeddings of all terminologies' concepts are indexed. We profiled retrieval speed on a server with Intel Xeon CPU E5-2660 2.00 GHz and 256 GB memory. First, we precomputed all embeddings for all concepts (500 thousand). On a single Nvidia TITAN X GPU, it takes about 7 min to compute all embeddings. Given that all embeddings are indexed on Nvidia TITAN X GPU using IndexFlatL2 index type. To obtain top candidates for 10 million queries, it requires approximately 3 h.

6 Conclusion

We studied the task of drug and disease normalization for clinical trials, using a newly created dataset of 500 interventional studies with 1075 intervention mentions and 819 condition mentions. We designed a triplet-based metric learning model named DILBERT that optimizes to pull pairs of mention and concept BioBERT representations closer than negative samples. We investigated strategies to obtain random and hard positive and negative examples using parent-child (i.e., broader-narrower) relationships between biomedical concepts. We performed experiments on in-KB and out-of-KB (*nil*) linking of mentions from the scientific domain to the clinical domain in a zero-shot setting. DILBERT shows better transfer capabilities for disease- and drug-related mentions compared to other state-of-the-art models. In future work, we plan to investigate taxonomy induction evaluation metrics and the normalization of protein/gene mentions.

Acknowledgements. Research on academic corpora was carried out by Z.M. and supported by RFBR, project no. 19–37-90074.

References

1. Aronson, A.R.: Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. In: Proceedings of the AMIA Symposium, p. 17. American Medical Informatics Association (2001)
2. Atal, I., Zeitoun, J.D., Névéol, A., Ravaud, P., Porcher, R., Trinquart, L.: Automatic classification of registered clinical trials towards the global burden of diseases taxonomy of diseases and injuries. BMC Bioinform. **17**(1), 392 (2016)
3. Bodenreider, O.: The unified medical language system (UMLS): integrating biomedical terminology. Nucleic Acids Res. **32**(suppl_1), D267–D270 (2004)
4. Boland, M.R., Miotto, R., Gao, J., Weng, C.: Feasibility of feature-based indexing, clustering, and search of clinical trials. Meth. Inf. Med. **52**(05), 382–394 (2013)
5. Brown, A.S., Patel, C.J.: A standard database for drug repositioning. Sci. Data **4**(1), 1–7 (2017)
6. Coletti, M.H., Bleich, H.L.: Medical subject headings used to search the biomedical literature. J. Am. Med. Inform. Assoc. **8**(4), 317–323 (2001)
7. Davis, A.P., et al.: The comparative toxicogenomics database: update 2019. Nucleic Acids Res. **47**(D1), D948–D954 (2019)
8. Davis, A.P., Wiegers, T.C., Rosenstein, M.C., Mattingly, C.J.: Medic: a practical disease vocabulary used at the comparative toxicogenomics database. Database **2012** (2012)
9. Dermouche, M., Looten, V., Flicoteaux, R., Chevret, S., Velcin, J., Taright, N.: ECSTRA-INSERM@ CLEF eHealth2016-task 2: ICD10 code extraction from death certificates. CLEF (2016)
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)

11. Gayvert, K.M., Madhukar, N.S., Elemento, O.: A data-driven approach to predicting successes and failures of clinical trials. Cell Chem. Biol. **23**(10), 1294–1301 (2016)
12. Ghiasvand, O., Kate, R.J.: UWM: Disorder mention extraction from clinical text using CRFs and normalization using learned edit distance patterns. In: SemEval@ COLING, pp. 828–832 (2014)
13. Gill, S.K., Christopher, A.F., Gupta, V., Bansal, P.: Emerging role of bioinformatics tools and software in evolution of clinical research. Perspect. Clin. Res. **7**(3), 115 (2016)
14. Gillick, D., et al.: Learning dense representations for entity retrieval. In: Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pp. 528–537 (2019)
15. Hao, T., Rusanov, A., Boland, M.R., Weng, C.: Clustering clinical trials with similar eligibility criteria features. J. Biomed. Inform. **52**, 112–120 (2014)
16. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Feragen, A., Pelillo, M., Loog, M. (eds.) SIMBAD 2015. LNCS, vol. 9370, pp. 84–92. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24261-3_7
17. Huang, C.C., Lu, Z.: Community challenges in biomedical text mining over 10 years: success, failure and the future. Briefings Bioinform. **17**(1), 132–144 (2015)
18. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 2333–2338 (2013)
19. Humeau, S., Shuster, K., Lachaux, M.A., Weston, J.: Poly-encoders: transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. CoRR abs/1905.01969. External Links: Link Cited by 2, 2–2 (2019)
20. Ivanenkov, Y., et al.: Identification of novel antibacterials using machine-learning techniques. Front. Pharmacol. **10**, 913 (2019)
21. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUS. arXiv preprint arXiv:1702.08734 (2017)
22. Leaman, R., Lu, Z.: Taggerone: joint named entity recognition and normalization with semi-markov models. Bioinformatics **32**(18), 2839–2846 (2016)
23. Lee, J., et al.: Biobert: pre-trained biomedical language representation model for biomedical text mining. Bioinformatics (2019)
24. Leveling, J.: Patient selection for clinical trials based on concept-based retrieval and result filtering and ranking. In: TREC (2017)
25. Li, H., et al.: CNN-based ranking for biomedical entity normalization. BMC Bioinform. **18**(11), 79–86 (2017)
26. Li, J., Lu, Z.: Systematic identification of pharmacogenomics information from clinical trials. J. Biomed. Inform. **45**(5), 870–878 (2012)
27. Li, J., et al.: Biocreative V CDR task corpus: a resource for chemical disease relation extraction. Database **2016** (2016)
28. Liu, Y., Guo, Y., Bakker, E.M., Lew, M.S.: Learning a recurrent residual fusion network for multimodal matching. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4107–4116 (2017)
29. Lo, B.: Sharing clinical trial data: maximizing benefits, minimizing risk. Jama **313**(8), 793–794 (2015)
30. McNemar, Q.: Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika **12**(2), 153–157 (1947)

31. Miftahutdinov, Z., Tutubalina, E.: Deep neural models for medical concept normalization in user-generated texts. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pp. 393–399 (2019)
32. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
33. Mork, J.G., Jimeno-Yepes, A., Aronson, A.R.: The NLM medical text indexer system for indexing biomedical literature. In: BioASQ@ CLEF (2013)
34. NLM: Umls glossary (2016). http://www.nlm.nih.gov/research/umls/new_users/glossary.html
35. Phan, M.C., Sun, A., Tay, Y.: Robust representation learning of biomedical names. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3275–3285 (2019)
36. Pradhan, S., Elhadad, N., Chapman, W.W., Manandhar, S., Savova, G.: Semeval-2014 task 7: Analysis of clinical text. In: SemEval@ COLING, pp. 54–62 (2014)
37. Reimers, N., Gurevych, I.: Sentence-bert: sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3973–3983 (2019)
38. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815–823 (2015)
39. Sen, A., et al.: The representativeness of eligible patients in type 2 diabetes trials: a case study using gist 2.0. J. Am. Med. Inform. Assoc. **25**(3), 239–247 (2018)
40. Shen, W., Wang, J., Han, J.: Entity linking with a knowledge base: issues, techniques, and solutions. IEEE Trans. Knowl. Data Eng. **27**(2), 443–460 (2014)
41. Sung, M., Jeon, H., Lee, J., Kang, J.: Biomedical entity representations with synonym marginalization. arXiv preprint arXiv:2005.00239 (2020)
42. Suominen, H., et al.: Overview of the ShARe/CLEF eHealth evaluation lab 2013. In: Forner, P., Müller, H., Paredes, R., Rosso, P., Stein, B. (eds.) CLEF 2013. LNCS, vol. 8138, pp. 212–231. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40802-1_24
43. Tutubalina, E., Kadurin, A., Miftahutdinov, Z.: Fair evaluation in concept normalization: a large-scale comparative analysis for bert-based models. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 6710–6716 (2020)
44. Tutubalina, E., Miftahutdinov, Z., Nikolenko, S., Malykh, V.: Medical concept normalization in social media posts with recurrent neural networks. J. Biomed. Inform. **84**, 93–102 (2018)
45. Van Mulligen, E., Afzal, Z., Akhondi, S.A., Vo, D., Kors, J.A.: Erasmus MC at CLEF eHealth 2016: Concept recognition and coding in French texts. CLEF (2016)
46. Wishart, D.S., et al.: Drugbank: a comprehensive resource for in silico drug discovery and exploration. Nucleic Acids Res. **34**(suppl_1), D668–D672 (2006)
47. Wright, D., Katsis, Y., Mehta, R., Hsu, C.N.: Normco: deep disease normalization for biomedical knowledge base construction. In: Automated Knowledge Base Construction (2019). https://openreview.net/forum?id=BJerQWcp6Q
48. Wu, P., Hoi, S.C., Xia, H., Zhao, P., Wang, D., Miao, C.: Online multimodal deep similarity learning with application to image retrieval. In: Proceedings of the 21st ACM International Conference on Multimedia, pp. 153–162 (2013)

49. Zhao, S., Liu, T., Zhao, S., Wang, F.: A neural multi-task learning framework to jointly model medical named entity recognition and normalization. Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 817–824 (2019)
50. Zhavoronkov, A., et al.: Deep learning enables rapid identification of potent DDR1 kinase inhibitors. Nat. Biotechnol. **37**(9), 1038–1040 (2019)
51. Zhu, M., Celikkaya, B., Bhatia, P., Reddy, C.K.: Latte: Latent type modeling for biomedical entity linking. arXiv preprint arXiv:1911.09787 (2019)

CEQE: Contextualized Embeddings for Query Expansion

Shahrzad Naseri[1](✉), Jeffrey Dalton[2], Andrew Yates[3], and James Allan[1]

[1] University of Massachusetts Amherst, Amherst, USA
{shnaseri,allan}@cs.umass.edu
[2] University of Glasgow, Glasgow, UK
jeff.dalton@glasgow.ac.uk
[3] Max Planck Institute for Informatics, Saarbrücken, Germany
ayates@mpi-inf.mpg.de

Abstract. In this work we leverage recent advances in context-sensitive language models to improve the task of query expansion. Contextualized word representation models, such as ELMo and BERT, are rapidly replacing static embedding models. We propose a new model, Contextualized Embeddings for Query Expansion (CEQE), that utilizes query-focused contextualized embedding vectors. We study the behavior of contextual representations generated for query expansion in ad-hoc document retrieval. We conduct our experiments on probabilistic retrieval models as well as in combination with neural ranking models. We evaluate CEQE on two standard TREC collections: Robust and Deep Learning. We find that CEQE outperforms static embedding-based expansion methods on multiple collections (by up to 18% on Robust and 31% on Deep Learning on average precision) and also improves over proven probabilistic pseudo-relevance feedback (PRF) models. We further find that multiple passes of expansion and reranking result in continued gains in effectiveness with CEQE-based approaches outperforming other approaches. The final model incorporating neural and CEQE-based expansion score achieves gains of up to 5% in P@20 and 2% in AP on Robust over the state-of-the-art transformer-based re-ranking model, Birch.

1 Introduction

Recently there is a significant shift in text processing from high-dimensional word-based representations to ones based on continuous low-dimensional vectors. However, fundamentally both are static – each word has a *context-independent* or static representation. The fundamental challenge of polysemy remains. Recent approaches aim to address this, namely ELMo [27] and BERT [6], by creating *context-dependent* representations that depend on the surrounding context in which they occur. The power of contextualized models comes from this ability to disambiguate and generate distinctive representations for terms with the same lexical form. Contextualized representation models provide significant improvements across a range of diverse tasks. To our knowledge this is the first work to

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 467–482, 2021.
https://doi.org/10.1007/978-3-030-72113-8_31

develop an unsupervised contextualized query expansion model based on pseudo-relevance feedback. This represents an advancement over previous context-free expansion models based on lexical matching. Our proposed approach leverages contextual word similarity with an unsupervised expansion model.

Contextualized representations from BERT and similar models are rapidly being adopted for retrieval and NLP, because they transfer well to new domains with limited training data. Supervised ranking models derived from them, such as CEDR [18] and T5 [22], are the top-ranked learning-to-rank methods for a wide range of retrieval and QA benchmarks. In this work we leverage these contextualized word representations not for supervised re-ranking, but instead to improve core document matching. We address the fundamental problem that for many queries the core matching algorithms fails to identify many (or even all) relevant results in the candidate pool. Advancements in retrieval require more effective core matching algorithms to improve recall for neural ranking methods. No amount of reranking irrelevant results will provide relevance gains.

We propose a new contextualized expansion method to address the task of core matching building on proven pseudo-relevance feedback (PRF) techniques from probabilistic Language Modeling and extending them to effectively leverage contextual word representations. Further, we investigate the effect of applying CEQE in combination with state-of-the-art neural re-ranking models. Our work addresses core research questions (RQ) in contextualized query expansion:

- **RQ1** How can contextualized representations be effectively leveraged to improve state-of-the-art unsupervised query expansion methods?
- **RQ2** How effective are neural reranking methods when performed after query expansion?
- **RQ3** How effective are query expansion methods after a first pass of high-precision neural re-ranking?

We study these questions with empirical experiments on standard TREC test collections: Robust and Deep Learning 2019. The results on these test collections demonstrate that variations of CEQE significantly outperform previous static embedding models (based on GLoVe) in extrinsic retrieval effectiveness by approximately 18% MAP on Robust04 and 31% on TREC Deep Learning 2019 and 6–9% for recall@1000 across all datasets.

This work makes several new contributions to methods and understanding of contextualized representations for query expansion and relevance feedback:

- We develop a new contextualized query expansion method, CEQE, that shifts from word-count approaches to contextualized query similarity.
- We demonstrate through experimental evaluation that the proposed approach outperforms static embedding methods and performs at least as well as state-of-the-art word-based feedback models on multiple collections.
- We demonstrate that neural reranking combined with CEQE results in state-of-the-art effectiveness that outperforms previous approaches.

2 Background and Related Work

Query Expansion. A widely used approach to improve recall uses query expansion from relevance feedback that takes a user judgment of a result's relevance and uses it to build an updated query model [30]. Pseudo-relevance feedback (PRF) [13,16,38] approaches perform this task automatically, *assuming* the top documents are relevant. We build on these proven approaches based on static representations and extend them to contextualized representations. Padaki et al. [24] investigate BERT's performance when using expanded queries and find that expansion that preserves some linguistic structure is preferrable to expanding with keywords.

Embedding-based Expansion. Another approach for query expansion incorporates static embeddings [19,26] to find the relevant terms to the query, because embeddings promise to capture the semantic similarity between terms and are used in different ways to expand queries [5,7,12,20,31,36,37]. These word embeddings, such as Word2Vec, GloVe, and others, learn a static word embedding for each term regardless of the context. Most basic models fail to address polysemy and the contextual characteristics of terms. All of the previous approaches use static representations that have fundamental limitations addressed by the use of contextualized representations.

Supervised Expansion. There is a vein of work using supervised learning to perform pseudo-relevance feedback. Cao et al. [2] and Imani et al. [10] use feature-based models to try to predict what terms should be used for expansion. A common practice is to classify terms as positive, negative, or neutral and use classification methods to maximize the number of predicted positive terms. We use this labeling method to intrinsically evaluate the utility of our unsupervised approach. An end-to-end neural PRF model (NPRF) proposed by Li et al. [14] uses a combination of models to compare document summaries and compute document relevance scores for feedback and achieves limited improvement while only using bag-of-words neural models. Later work combining BERT with a NPRF framework [41] illustrated the importance of an effective first-stage ranking method. A complementary vein of work [23] uses generative approaches to perform *document expansion* by predicting questions to add to document. In contrast, we focus on query expansion approaches.

Neural Ranking. Contextualized Transformer-based models are now widely used for ranking tasks [1,4,15,18,21,22,25,29,40]. MacAvaney et al. [18] propose incorporating contextualized language models into existing neural ranking architectures by considering each layer of contextualized language models as one channel and integrating the similarity matrices of each layer in the neural ranking architecture. Recent research [8,11,17,33,39] uses Transformer models to produce query and document representations that can be used for (relatively) efficient first-stage retrieval. In this context, Gao et al. [8] find that combining a representation-based model with a lexical matching component improves effectiveness. In contrast, we focus on representations solely as a contextualized word representation model for the task of unsupervised query expansion.

3 Methodology

In this section we introduce our proposed Contextualized Embedding for Query Expansion (CEQE) method that utilizes contextualized representations for the task of query expansion. The method below applies to many widely used contextualized embedding representation models, including BERT and its variants.

3.1 Word and WordPiece Representations

In contextualized models, to address the problem of out-of-the-vocabulary terms, subword representation such as WordPieces [32] are used. For backwards compatibility with existing word-based retrieval systems (as well as comparison with previous methods) we use words as the matching unit. We first aggregate WordPiece tokens into a contextualized vector for words. We compute the average embedding vector of word w by $\overrightarrow{w} \triangleq \frac{1}{|w|} \sum_{p_i \in w} \overrightarrow{p_i}$, where p_i is a WordPiece of word w and $|w|$ is the number of WordPieces in the word w.

3.2 Contextualized Embeddings for Query Expansion (CEQE)

In this section we describe the core of the CEQE model. It follows in the vein of principled probabilistic language modeling approaches, such as the Relevance Model formulation of pseudo-relevance feedback [13]. In contrast to these approaches that are based on static lexical matching, we formulate relevance based on contextualized vector representations. We build the contextualized feedback model based upon the core Relevance Model (RM) formulation:

$$p(w|\theta_R) \propto \sum_{D \in R} p(w, Q, D) \tag{1}$$

where θ_R and R respectively denote the feedback language model and the set of pseudo-relevant documents, i.e., the top retrieved documents. In the original RM formulation, the joint probability of $p(Q, w, D)$ is broken down as follows:

$$\sum_{D \in R} p(w, Q, D) = \sum_{D \in R} p(w, Q|D)p(D) \tag{2}$$

$$= \sum_{D \in R} p(w|D)p(Q|D)p(D) \tag{3}$$

where Eq. 3 is derived from the simplifying independence assumption between the query Q and term w. This assumption results in a static representation based on simple word counts and ignores the query explicitly. It only incorporates evidence indirectly through $P(Q|D)$. In contrast, the proposed CEQE parameterization doesn't assume term w is independent of query Q and explicitly incorporates the query focus based on similarity with contextualized vector representations. More formally:

$$\sum_{D \in R} p(w, Q, D) = \sum_{D \in R} p(w|Q, D)p(Q|D)p(D) \tag{4}$$

With a contextualized model it is no longer possible to simply count document terms – they must be grouped, simplified, or compared against a query representation. We explicitly incorporate contextualized query similarity for each word occurrence. We now break down each of the elements in Eq. 4 in more detail. Following common practice we assume a uniform probability for $p(D)$. $p(Q|D)$ is the posterior probability of the query given a document from the retrieval model. We propose several methods to calculate $p(w|Q, D)$ below.

Centroid Representation. In this approach we create a model of the whole query and then compare it to the contextualized representation of each word mention (occurrence), m_w. In the centroid representation we define $\sigma(Q)$, the aggregation of all WordPieces of the query. Note that a representation of a query also includes special delimiter tokens. For example, in BERT this would include [CLS] and [SEP] tokens that we find carry contextual importance. We include the [CLS] token in particular because it is often used as a representation of the input with respect to the target task. For the query centroid representation we define σ as the mean of its individual component contextual vectors: we represent query $\sigma(Q)$ by $\overrightarrow{Q} \triangleq \frac{1}{|Q|} \sum_{q_i \in Q} \overrightarrow{q}$, where q_i is a WordPiece token and $|Q|$ is the length of the query in WordPiece tokens.

We then define $p(w|Q, D)$ by comparing the similarity of individual word mentions to the query centroid representation based on a similarity function δ (e.g., cosine). If m_w^D is a mention of word w in a document D and M_w^D is the complete set of mentions of w:

$$p(w|Q, D) \triangleq \frac{\sum_{m_w^D \in M_w^D} \delta(\overrightarrow{Q}, \overrightarrow{m_w^D})}{\sum_{m^D \in M_*^D} \delta(\overrightarrow{Q}, \overrightarrow{m^D})} \tag{5}$$

The denominator is a normalization constant that considers all word mentions across the entire document to form a probability. This approach is novel because the contextualized vector m_w^D will be different for every occurrence in D because the context surrounding each mention of word w varies.

Term-based Representation. In this section we propose an alternative parameterization for $p(w|Q, D)$. Instead of using the centroid of the query to compute a term's similarity to the entire query, we compute the similarity for each query term separately. If q is a query term and $\overrightarrow{q}$ is its corresponding contextualized embedding vector, this can be formulated as:

$$p(w|q, D) \triangleq \frac{\sum_{m_w^D \in M_w^D} \delta(\overrightarrow{q}, \overrightarrow{m_w^D})}{\sum_{m^D \in M_*^D} \delta(\overrightarrow{q}, \overrightarrow{m^D})} \tag{6}$$

To select a term for expansion for the query overall we perform an extra step of pooling across the similarities of individual words. This step combines the contextualized word vectors. Function f calculate the semantic similarity of word w with the whole query by combining the semantic similarity of it with each query term q. We define $f_{\max}(w, Q, D) = \max_{q \in Q} p(w|q, D)$ and $f_{\text{prod}}(w, Q, D) = \prod_{q \in Q} p(w|q, D)$ as MaxPool and MulPool, respectively. If Z' is a normalization factor that is the sum over the terms in document D, which is less computationally expensive than summing over all vocabulary terms, these can be defined as:

$$p(w|Q, D) \triangleq \frac{f_{\text{max/prod}}(w, Q, D)}{Z'} \tag{7}$$

The final result of all of these methods is a relevance distribution over terms derived from the contextualized representations in top retrieved documents. The result is an updated query language model that can be used on its own or combined with other representations.

4 Experimental Setup

4.1 Datasets

We evaluate our model on two standard TREC datasets: Robust and Deep Learning.

Robust. The corpus consists of Tipster disks 4 and 5 containing approximately 528 K newswire articles. The evaluation topics are the 250 Robust topics (301–450, 601–700). We use the titles as queries.

TREC Deep Learning. The 2019 TREC Deep Learning (DL) Track created large labeled datasets for ad-hoc search. We perform the full document ranking task with the goal of testing new expansion methods to improve effectiveness. The evaluation has 43 test queries from Bing, and the corpus consists of 3.2 million web documents. Documents are rated on a four point graded relevance scale. The primary measure is NDCG@10.

Evaluation Metrics. Since we focus on introducing relevant documents to a candidate pool for downstream ranking, we consider both recall-focused metrics (Recall@100, Recall@1000, MAP) as well as precision-based measures (P@10/20, NDCG@10/20). For Robust, in order to compare with previous works we report precision and NDCG at cut-off 20. We report the official primary measure for DL, NDCG@10. For significance testing, we use a paired t-test with significance at the 95% confidence interval.

4.2 Baselines

We study the behavior of the CEQE model in comparison with standard models from probabilistic language modeling. For the baseline retrieval we use BM25 because it is the most widely used first-pass unsupervised ranker used to generate candidate pools. We compare to two static expansion models [12] and a proven pseudo-relevance feedback model, the Relevance Model [13]. We use the standard relevance model (RM3 variant) that performs linear interpolation of the RM expansion terms with the original query using the Query Likelihood score.

Static Embeddings. For static word embeddings we use GloVe [26] embeddings. The pre-trained 300 dimensional Glove word embeddings are extracted from a 6 billion token collection (Wikipedia dump 2014 plus Gigawords 5). These embeddings are the most effective static embeddings for a variety of tasks, including previous work [7] on query expansion. We use the static embeddings with two variations. The *Static-Embed model* [12] is a global expansion model using GloVe expansion on the target collection vocabulary. For a fair comparison with CEQE, we additionally consider a *Static-Embed-PRF* variant that has its vocabulary limited to terms appearing in the PRF documents.

4.3 Intrinsic Expansion Judgments

Beyond direct retrieval, we also assess term selection quality intrinsically. We directly measure the utility of individual expansion terms. Following previous work from Imani et al., we generate this term utility by performing expansion one word at a time [10]. Retrieval effectiveness assesses whether a term is good (helps retrieval), bad (hurts retrieval), or neutral (has no effect). We pool the top thousand candidate expansion terms from all candidate expansion methods. These are issued to the retrieval system with the original query (each with a default weight of 0.5, the default relevance model expansion weight). This approach follows standard relevance model interpolation practice and removes the dependence on the original query length (instead of simply appending a word). We measure improvement based on recall@1000 with a threshold of 0.001. For Robust this results in approximately 500k candidate terms. For the intrinsic evaluation only queries with at least one positive expansion term are used. This is 181 queries for Robust with 10,068 positive terms.

4.4 System Details

All collections are indexed with the Galago[1] open-source retrieval system for research. The query models and feedback expansion models are all implemented using the Galago query language. We perform stopword removal and stemming using Galago's stopword list and Krovetz stemmer, respectively.

Contextualized Embedding Model. We use BERT because it is the most widely used contextual representation model. We use the pre-trained BERT (BERT-Base, Uncased) model with maximum sequence length of 128 for calculating the contextualized embedding vectors. Since the documents in Robust are longer than 128 tokens we split the documents into chunks with maximum size of 128 tokens. For the primary CEQE results in this section we use a single layer of the contextualized representation, the second to last layer (11) of BERT. This layer was shown to be the most effective single layer on NER [6] and it was shown that later layers (before the last) were the most effective word representations for multiple language tasks [28] that use contextual embeddings as features. Initial preliminary experiments confirmed this finding.

Neural Ranking Models. For our neural models we adopt CEDR [18]. In particular, to align with the use of the contextualized models we use the BERT variant. For Robust, we use the CEDR-KNRM model trained by the authors [18]. Throughout the paper we refer to the CEDR-KNRM as CEDR. For DL we use a CEDR variant trained on a random sample of 1000 MS MARCO train queries with early stopping to terminate when there is no validation improvement for 20 iterations.

Parameter Settings. The unsupervised retrieval and feedback hyperparameters are tuned using grid search. The b and $k1$ are tuned for BM25 as well as mu for the QL model in the RM3 score. For all PRF query expansion methods we tune the number of documents ($\{5, 10, ..., 100\}$ by 5), terms ($\{10, 20, ..., 100\}$ by 10) , and interpolation coefficient ($\{0.1, 0.2, ..., 0.9\}$ by 0.05). For Robust, we use five-fold cross-validation with the splits introduced by Huston and Croft [9]. For DL the original 2019 track only used MS MARCO for training. We set hyper-parameters using five cross-validation with random splits on the topics.

5 Experimental Results

First, in Sect. 5.1 we study how to incorporate contextualized embeddings for the task of unsupervised query expansion (RQ1). Then, in Sect. 5.2 we explore the effect of CEQE variants in combination with neural ranking methods (RQ2). Finally, in Sect. 5.3 section we study how a reranked neural result can be used as a basis for further expansion and reranking (RQ3).

[1] http://www.lemurproject.org/galago.php.

5.1 Unsupervised Expansion Comparison

We first evaluate our expansion model on retrieval effectiveness in extrinsic evaluation. We study this setup because these are the most widely used algorithms for first pass retrieval. In this pass it is critical to focus on recall at a cutoff, particularly with a low cutoff due to the computational requirements of second pass reranking (e.g., the top 100 documents as in [18] and the Deep Learning Track [3]). We present the results of the methods as well as baselines for Robust04 in Table 1 and 2019 Deep Learning Track in Table 2.

Robust (Table 1). The results on Robust show that all expansion methods outperform the baseline BM25 retrieval method across all measures. The static embedding models outperform BM25, but do not perform as well as the Relevance Model (RM3). The effectiveness of the Static-Embed-PRF method that only uses terms in the PRF documents' vocabulary is more effective across all measures over the Static-Embed approach with a global vocabulary. We hypothesize that this may be due to the fact that the query results provide a topically focused vocabulary and filters out generally similar noise. RM3 significantly outperforms the Static-Embed method for MAP, but not other measures. To give an indicator of the BM25 + RM3 parameters, the average parameter settings across the folds is: 22 feedback docs, 71 expansion terms, and interpolation weight of 0.3. We observe that the contextualized expansion methods outperform the static embedding models. The results show the best method is CEQE-MaxPool. The Centroid method is slightly lower than MaxPool, and both outperform multiplicative pooling. The CEQE-MaxPool result outperforms the BM25+RM3 across all measures and in Recall@1000 is significant over both static embedding methods and BM25+RM3, which demonstrates the utility of context-dependent embeddings.

Table 1. Ranking effectiveness on the Robust collection. The superscript † and ‡ denotes statistical significance over BM25 + RM3 and Static-Embed-PRF, respectively.

Model	P@20	nDCG@20	mAP@1000	Recall@100	Recall@1000
BM25	0.3657	0.4193	0.2574	0.4165	0.6933
BM25 + RM3	0.3998	0.4517	0.3069	0.4610‡	0.7588‡
Static-Embed	0.3675	0.4285	0.2615	0.4217	0.7125
Static-Embed-PRF	0.3781	0.4400	0.2703	0.4324	0.7231
CEQE-Centroid	0.3922	0.4462	0.3019‡	0.4593‡	0.7653†‡
CEQE-MulPool	0.3847	0.4360	0.2845‡	0.4517‡	0.7435‡
CEQE-MaxPool	**0.4040**‡	**0.4587**	**0.3086**‡	**0.4651**‡	**0.7689**†‡
CEQE-MaxPool(fine-tuned)	0.3986‡	0.4528	0.3071‡	0.4647‡	0.7626‡

The last line of the table shows the result of using MaxPool with 'fine-tuned' contextual embeddings from a BERT model trained for ranking on Robust. The results show small and insignificant differences across all measures. It is almost identical to vanilla embedding effectiveness after being combined with RM3. This indicates that, when used for CEQE-based expansion, pre-trained models are comparable in effectiveness to ones fine-tuned for ranking. To our knowledge these are the best unsupervised query expansion results for Robust that do not use external collections.

Deep Learning 19 (Table 2). We report the official evaluation measures for the TREC 2019 Deep Learning Track [3] as well as Recall@1000. For NDCG@10, the baseline BM25 retrieval is more effective than all expansion methods. To give an indicator of the BM25 + RM3 parameters, the average parameter settings across the folds is: 15 feedback docs, 85 expansion terms, and interpolation weight of 0.4. Similar to Robust, we observe that a tuned RM3 outperforms the static embedding methods across all measures. CEQE-MulPool and CEQE-MaxPool also outperform the static embedding model across all measures. The best performing expansion method is CEQE-MaxPool, outperforming RM3. We note that given the small sample size (43 topics), none of the unsupervised methods show statistically significant differences between them. As shown later, that requires performing expansion on top of neural rankings.

Although our experimental setup is based on cross-fold validation (rather than tuning on MARCO), we include the reported values from the Deep Learning track overview [3] for reference. Importantly, we observe that the CEQE-MaxPool outperforms all submitted TREC systems on recall@1000 and is in the top five for recall@100. It's noteworthy that the unsupervised CEQE-MaxPool 'traditional' model is only slightly lower than the median for P@10 and NDCG@10 with runs that include many state-of-the-art neural models.

Intrinsic Evaluation. In this section we examine the effectiveness of the expansion approaches to rank positive expansion terms that improve Mean Average Precision (at 1000) when added to the query. This experiment evaluates

Table 2. Ranking effectiveness of CEQE on unsupervised baseline retrieval for Deep Learning 2019 Track for the task of full document ranking. The superscript † and ‡ denotes statistical significance over BM25 + RM3 and Static-Embed, respectively.

Model	P@10	nDCG@10	mAP@1000	Recall@100	Recall@1000
BM25	0.6535	**0.5730**	0.3513	0.4053	0.6950
BM25 + RM3	0.6256	0.5343	$0.3975^{‡}$	$0.4434^{‡}$	$0.7750^{‡}$
Static-Embed	0.6186	0.5427	0.3373	0.3973	0.7179
Static-Embed-PRF	0.5605	0.4925	0.3166	0.3715	0.6737
CEQE-Centroid	0.5580	0.5580	$0.4144^{‡}$	$0.4464^{‡}$	$0.7804^{‡}$
CEQE-MulPool	0.6442	0.5563	$0.3724^{‡}$	$0.4295^{‡}$	$0.7560^{‡}$
CEQE-MaxPool	**0.6581**	0.5614	$0.4161^{†‡}$	$0.4506^{‡}$	$0.7832^{‡}$
TREC 2019 Median	0.6597	0.5834	0.2984	0.3748	0.5484
TREC 2019 Best	0.8093	0.7260	0.4280	0.4670	0.7553

Table 3. Intrinsic ranking evaluation of expansion terms on Robust. Significance over Relevance Model is indicated by † and Static-Embed-PRF by ‡.

Model	P@10	P@20	P@100
Relevance model	0.1693‡	0.1419‡	**0.0871**‡
Static-Embed	0.1008	0.0780	0.0511
Static-Embed-PRF	0.1357	0.1083	0.0655
CEQE-MulPool	0.1349	0.1174	0.0737
CEQE-Centroid	0.1751‡	0.1481‡	0.0826‡
CEQE-MaxPool	**0.1830**†‡	**0.1500**†‡	0.0841‡

a method's ability to identify good expansion terms in isolation. The results are shown in Table 3 for the key expansion models to compare for Robust collection. Because a fixed top-k expansion terms are usually selected for expansion we evaluate the intrinsic evaluation with set-based precision numbers at common thresholds for the number of expansion terms. The results show that a well-tuned Relevance Model outperforms query expansion models based on static embeddings. In contrast, we find that our proposed contextualized embedding model, CEQE, provide improvements in early ranks for P@10 and P@20. All the CEQE models significantly improve over static embedding models across all metrics. And further, we find that CEQE-MaxPool significantly outperforms the Relevance Model expansion effectiveness for P@10 and P@20. It is insignificantly different from the Relevance Model at rank 100. This indicates that strength of CEQE is selecting a higher number of "good" terms earlier, allowing improved effectiveness with fewer expansion terms.

We explore the intrinsic results in more depth with an example for one topic on Robust in Table 4. The first column has the terms (unstemmed) with the greatest improvement for the query. The ranking of expansion terms for the Relevance Model and CEQE-MaxPool are shown for comparison. We observe that CEQE model identifies all of the terms from RM as well as three additional relevant terms. More generally, we see that the CEQE terms appear to have

Table 4. Example top expansion terms for Topic 685, [oscar winner selection]. This includes a sample of the most important intrinsic positive labels, Relevance Model terms, and CEQE Expansion terms. Terms with positive intrinsic labels are highlighted.

Positive terms:	academy, academys, nominations, nomination, critics, members, branch, ignored, true, films, film, directors, director, filmmaker
RM:	best, **film**, picture, million, **academy**, years, **award**, home, edition, **films**, man, four, 1, 5
CEQE-Maxpool:	**film**, **academy**, picture, winners, **award**, **films**, million, oscars, box, presented, **awards**, **director**, years, **nominations**

Table 5. Ranking effectiveness of neural ranking on top of query expansion methods for Robust. The superscript † and ‡ indicates significance over BM25 + CEDR and (BM25 + RM3) + CEDR with re-ranking top 1000, respectively.

Model	P@20	nDCG@20	mAP@1000	Recall@100	Recall@1000
BM25 + RM3	0.3998	0.4517	0.3069	0.4610	0.7588
BM25 + CEDR [18]	0.4713	0.5458	0.3312	0.4983	0.6933
(BM25 + RM3) + CEDR	0.4719	0.5435	0.3500†	0.5192†	0.7570†
(BM25 + CEQE-MaxPool) + CEDR	**0.4735**	**0.5462**	**0.3532**†	**0.5258**†‡	**0.7719**†‡

a stronger semantic relationship with the query terms. The RM terms appear most loosely related and have additional noise terms, including single digit numbers. This is because RM focuses on terms that co-occur across multiple PRF documents, but it does not explicitly model the relationship to the query. In contrast our proposed model explicitly focuses on the query. As a result, the CEQE model produces fewer terms that co-occur by chance.

5.2 PRF Effect on Neural Reranking

We now study how PRF methods impact the effectiveness of neural reranking models (RQ2). It is important to have effective expansion in the first pass to retrieve sufficient numbers of documents to rerank. The results of our experiments on Robust are shown in Table 5. Applying neural reranked models baselines designed for document ranking, CEDR [18], on expanded query runs results in significant gains to average precision, recall@100, and recall@1000 for both RM3 and CEQE. Replacing RM3 with CEQE for expansion results in significant improvement over Recall@100 and Recall@1000. The PRF parameters are 20 documents, 90 terms, and interpolation weight of 0.3.

5.3 Expansion After Reranking

In this section we study how a reranked neural result can be used as a basis for further expansion and reranking (RQ3). This is a critical step because there must be a sufficient number of relevant documents in the top ranks for PRF to be effective. We evaluate multi-round supervised reranking based on expansion runs for Robust in Table 6. The top of the table shows results from the leading neural ranking and PRF approaches, including Neural PRF [14], CEDR, and Birch [35]. The results in this section all perform re-ranking on 1000 results from the baseline. We experimented with reranking 100 results and found it consistently performed worse. The baseline model run is BM25+CEDR followed by RM3 expansion with CEDR reranking, which we denote as *(BM25 + CEDR) + RM3 + CEDR*. The results show it outperforms Birch in NDCG@20 and P@20, as well as its own previous result for P@20 on just BM25. Replacing RM3 with CEQE for the expansion consistently outperforms the previous best CEDR results across all measures and significantly over Recall@1000. The runs compare

Table 6. Ranking effectiveness of multi-round neural re-ranking and expansion for Robust. The superscript † and ‡ indicates significance over BM25 + CEDR and (BM25 + CEDR) + RM3 baselines, respectively.

Model	P@20	nDCG@20	mAP@1000	Recall@100	Recall@1000
Neural PRF-DRMM [14]	0.4064	0.4576	0.2904	–	–
BM25 + CEDR [18]	0.4713	0.5458	0.3312	0.4983	0.6933
Birch [35]	0.4657	0.5325	0.3697	–	–
(BM25 + CEDR) + RM3	0.4458	0.5211	0.3321	0.4881	0.7751†
(BM25 + CEDR) + RM3 + CEDR	0.4783	0.5499	0.3574†	0.5291†	0.7751†
(BM25 + CEDR) + RM3 + CEDR Interp	0.4837†	0.5565	0.3739†	0.5440†	0.7751†
(BM25 + CEDR) + CEQE-MaxPool	0.4504	0.5250	0.3366	0.4931	0.7874†‡
(BM25 + CEDR) + CEQE-MaxPool + CEDR	0.4799	0.5516	0.3601†	0.5332†	0.7874†‡
(BM25 + CEDR) + CEQE-MaxPool + CEDR Interp	**0.4904**†	**0.5621**†	**0.3773**†	**0.5486**†	**0.7874**†‡

performing RM3 and CEQE-MaxPool on the CEDR baseline (which reranks an initial BM25 first run). The second pass results are then reranked again using CEDR. The result is further improve over previous approaches. The same trend continues, with the CEQE-MaxPool outperforming the reranked RM3 run.

A common approach when applying BERT-based neural ranking is to perform learning-to-rank to combine the BERT and retrieval score. A simple proven approach is the linear interpolation of the underlying retrieval score with neural ranking model [34,35]. We apply this to the two best runs, learning the interpolation using the previously described cross-validation setup. The results demonstrate that linear interpolation with these expansion runs continues to show gains. The interpolation with CEQE-MaxPool is the best performing, and compared with the previous Birch shows over 5% relative gain P@20 and nDCG@20 as well as improving MAP. These results show that multiple rounds of expansion and reranking can continue to result in significant improvements.

6 Conclusion

We introduce a new method, CEQE, for query expansion that extends relevance feedback approaches to recent advances in contextualized language models. CEQE address fundamental challenges using context-dependent term representations for unsupervised pseudo-relevance feedback. We study its empirical effectiveness on multiple standard test collections and the results demonstrate that they are superior to previous static embedding approaches.

Acknowledgement. This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #IIS-1617408. 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. Akkalyoncu Yilmaz, Z., Yang, W., Zhang, H., Lin, J.: Cross-domain modeling of sentence-level evidence for document retrieval. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Association for Computational Linguistics, Hong Kong, China, November 2019
2. Cao, G., Nie, J.Y., Gao, J., Robertson, S.: Selecting good expansion terms for pseudo-relevance feedback. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR 2008, ACM, New York, NY, USA (2008)
3. Craswell, N., Mitra, B., Yilmaz, E., Campos, D.: Overview of the trec 2019 deep learning track. In: Proceedings of The Twenty-Eight Text REtrieval Conference, TREC 2019, Gaithersburg, Maryland, USA, November 13–15, 2019 (2019)
4. Dai, Z., Callan, J.: Deeper text understanding for IR with contextual neural language modeling. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR 2019, Association for Computing Machinery, New York, NY, USA (2019)
5. Dalton, J., Naseri, S., Dietz, L., Allan, J.: Local and global query expansion for hierarchical complex topics. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11437, pp. 290–303. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15712-8_19
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, June 2019
7. Diaz, F., Mitra, B., Craswell, N.: Query expansion with locally-trained word embeddings. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2016)
8. Gao, L., Dai, Z., Fan, Z., Callan, J.: Complementing lexical retrieval with semantic residual embedding. arXiv preprint arXiv:2004.13969 (2020)
9. Huston, S., Croft, W.B.: Parameters learned in the comparison of retrieval models using term dependencies. University of Massachusetts, Ir (2014)
10. Imani, A., Vakili, A., Montazer, A., Shakery, A.: Deep neural networks for query eexpansion using word embeddings. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11438, pp. 203–210. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15719-7_26
11. Khattab, O., Zaharia, M.: ColBERT: efficient and effective passage search via contextualized late interaction over BERT. In: Proceedings of the 43rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2020) (2020)
12. Kuzi, S., Shtok, A., Kurland, O.: Query expansion using word embeddings. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. ACM (2016)
13. Lavrenko, V., Croft, W.B.: Relevance based language models. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR 2001, ACM, New York, NY, USA (2001)
14. Li, C., et al.: NPRF: a neural pseudo relevance feedback framework for ad-hoc information retrieval. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (2018)

15. Li, C., Yates, A., MacAvaney, S., He, B., Sun, Y.: Parade: passage representation aggregation for document reranking. arXiv preprint arXiv:2008.09093 (2020)
16. Lv, Y., Zhai, C.: A comparative study of methods for estimating query language models with pseudo feedback. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management. ACM (2009)
17. MacAvaney, S., Nardini, F.M., Perego, R., Tonellotto, N., Goharian, N., Frieder, O.: Expansion via prediction of importance with contextualization. arXiv preprint arXiv:2004.14245 (2020)
18. MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: CEDR: contextualized embeddings for document ranking. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21–25 (2019)
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems (2013)
20. Naseri, S., Foley, J., Allan, J., O'Connor, B.: Exploring summary-expanded entity embeddings for entity retrieval. In: CEUR Workshop Proceedings (2018)
21. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)
22. Nogueira, R., Jiang, Z., Lin, J.: Document ranking with a pretrained sequence-to-sequence model. arXiv preprint arXiv:2003.06713 (2020)
23. Nogueira, R., Yang, W., Lin, J., Cho, K.: Document expansion by query prediction. arXiv preprint arXiv:1904.08375 (2019)
24. Padaki, R., Dai, Z., Callan, J.: Rethinking query expansion for BERT reranking. In: Jose, J.M., et al. (eds.) ECIR 2020. LNCS, vol. 12036, pp. 297–304. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45442-5_37
25. Padigela, H., Zamani, H., Croft, W.B.: Investigating the successes and failures of bert for passage re-ranking. arXiv preprint arXiv:1905.01758 (2019)
26. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014)
27. Peters, M., et al.: Deep contextualized word representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). Association for Computational Linguistics, New Orleans, Louisiana, June 2018
28. Peters, M.E., Ruder, S., Smith, N.A.: To tune or not to tune? adapting pretrained representations to diverse tasks. In: RepL4NLP@ACL (2019)
29. Qiao, Y., Xiong, C., Liu, Z., Liu, Z.: Understanding the behaviors of bert in ranking. arXiv preprint arXiv:1904.07531 (2019)
30. Rocchio, J.J.: Relevance feedback in information retrieval. In: Salton, G. (ed.) The SMART Retrieval System: Experiments in Automatic Document Processing, chap. 14, pp. 313–323. Prentice-Hall Series in Automatic Computation, Prentice-Hall, Englewood Cliffs NJ (1971)
31. Roy, D., Paul, D., Mitra, M., Garain, U.: Using word embeddings for automatic query expansion, July 2016
32. Schuster, M., Nakajima, K.: Japanese and Korean voice search. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2012)
33. Xiong, L., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: International Conference on Learning Representations (2021)

34. Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M., Lin, J.: End-to-end open-domain question answering with bertserini. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations) (2019)
35. Yilmaz, Z.A., Wang, S., Yang, W., Zhang, H., Lin, J.: Applying bert to document retrieval with birch. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations, pp. 19–24 (2019)
36. Zamani, H., Croft, W.B.: Embedding-based query language models. In: Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval. ACM (2016)
37. Zamani, H., Croft, W.B.: Relevance-based word embedding. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 505–514. ACM (2017)
38. Zhai, C., Lafferty, J.: Model-based feedback in the language modeling approach to information retrieval. In: Proceedings of the Tenth International Conference on Information and Knowledge Management. CIKM 2001, ACM, ACM, New York, NY, USA (2001). http://doi.acm.org/10.1145/502585.502654
39. Zhan, J., Mao, J., Liu, Y., Zhang, M., Ma, S.: Repbert: contextualized text embeddings for first-stage retrieval. arXiv preprint arXiv:2006.15498 (2020)
40. Zhang, H., et al.: Generic intent representation in web search. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, 21–25 July, Paris, France (2019)
41. Zheng, Z., Hui, K., He, B., Han, X., Sun, L., Yates, A.: Bert-QE: Contextualized query expansion for document re-ranking. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, pp. 4718–4728 (2020)

Pattern-Aware and Noise-Resilient Embedding Models

Mojtaba Nayyeri[1,2(✉)], Sahar Vahdati[2,3], Emanuel Sallinger[3,4],
Mirza Mohtashim Alam[1,2], Hamed Shariat Yazdi[1], and Jens Lehmann[1,5]

[1] University of Bonn, Bonn, Germany
nayyeri@cs.uni-bonn.de
[2] InfAI Lab, Dresden, Germany
{vahdati,mohtasim}@infai.org
[3] University of Oxford, Oxford, UK
{sahar.vahdati,emanuel.sallinger}@cs.ox.ac.uk
[4] TU Wien, Vienna, Austria
sallinger@dbai.tuwien.ac.at
[5] Fraunhofer IAIS, Dresden, Germany
jens.lehmann@iais.fraunhofer.de

Abstract. Knowledge Graph Embeddings (KGE) have become an important area of Information Retrieval (IR), in particular as they provide one of the state-of-the-art methods for Link Prediction. Recent work in the area of KGEs has shown the importance of relational patterns, i.e., logical formulas, to improve the learning process of KGE models significantly. In separate work, the role of noise in many knowledge discovery and IR settings has been studied, including the KGE setting. So far, very few papers have investigated the KGE setting considering both relational patterns and noise. Not considering both together can lead to problems in the performance of KGE models. We investigate the effect of noise in the presence of patterns. We show that by introducing a new loss function that is both pattern-aware and noise-resilient, significant performance issues can be solved. The proposed loss function is model-independent which could be applied in combination with different models. We provide an experimental evaluation both on synthetic and real-world cases.

Keywords: Knowledge graph · Embedding · Noise · Relational pattern

1 Introduction

Knowledge Graph Embeddings (KGE) have become an important area of information retrieval, in particular as they provide one of the state-of-the-art methods for Link Prediction. Embedding models typically receive Knowledge Graphs as a set of *correct* edges, i.e., triples in the form of (*subject*, *relation*, *object*) representing a fact such as (*Student1*, *isPhDin*, *Group1*). The role of KG embedding models is to take the symbolic representation (i.e., edges) and embed it into a

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 483–496, 2021.
https://doi.org/10.1007/978-3-030-72113-8_32

vector space. For learning such embeddings, a typical scenario is to use existing edges as positive samples. Also, negative samples are generated by applying random corruptions to positive samples. As described so far, we see the ideal scenario, yet once knowledge graph embeddings are applied in real word settings, we see two factors: the appearance of *patterns* between relations, and the existence of *noise*. Let us first consider the former. Recent work in the KGE area has shown the importance of *relational patterns*, i.e., logical formulas, to improve the learning process of KGE models significantly [9]. Relations between entities of a KG often form particular patterns which can, e.g., be represented as logical rules. Such patterns can be either stated directly based on domain knowledge or inferred statistically based on the data. For example, if we know the following statistical pattern from the data, namely that (*Student*, *isPhDin*, *Group*) and (*Group*, *ledBy*, *Leader*) in most cases implies (*Student*, *supervisedBy*, *Leader*), we may expect our KGE model to infer such a fact as well. Such relational patterns can be injected into the learning process, or be statistically inferred during the learning process of a KGE model [5,7,8,21]. The other factor when we hit the real world, apart from patterns, is the existence of *noise*. The role of noise has been studied in many knowledge-based settings [10], including the KGE setting [24]. It is very hard in reality to distinguish noise (i.e., incorrect edges) from correct edges [12,20,25]. Noise in the presence of patterns in many off-the-shelf KGE models actually propagates along relational patterns. One of the reasons for this can be that the original edges are actually not correct, but noise. Another reason is that noise may, via relational patterns, lead to the creation of edges creating further contradictions. We frequently encounter situations where we have a number of such contradictions coming from incorrect triples, i.e., noise, as well as contradictions coming from edges learned via patterns. So far, very few papers have investigated the KGE setting considering *both* relational patterns and noise, which, allows one to overcome significant problems in the performance of KGEs. We investigate the effect of noise in the presence of patterns and, specifically, show that, noise on a particular edge (triple) will, via patterns, affect the score of other edges. We introduce a new loss function UNITE that is both noise-resilient and pattern-aware, i.e., allows to provide good performance even in the presence of noise and relational patterns. This new loss function is model independent and can be employed across KGE models. We provide an experimental evaluation, both on synthetic KGs where noise is explicitly introduced based on known patterns, and a large-scale real-world evaluation, where noise is randomly introduced on mostly unknown patterns.

2 Motivating Example

To directly illustrate the destructive effect of noise in combination with relational patterns, let us consider an example. In the lower part of Fig. 1, we see a Knowledge Graph describing academic research groups (shown in yellow and marked as "group" in the diagram), students (shown in orange) and group leaders (shown in blue). We have three relations shown as directed edges, namely that a group can be led by a group leader, a student can be supervised by a group leader, and a student can work in, or be doing a PhD in a group. We here zoom-in into one portion of the graph to highlight typical relationships – other areas are shown in gray color in the background (for 22 triples in our example).

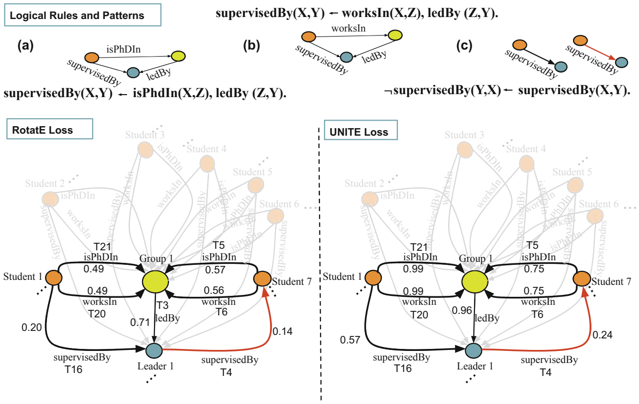
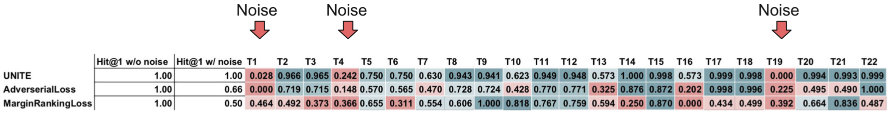


Fig. 1. Comparison of scoring triple correctness between RotatE and the UNITE – with three relational patterns in a KG in presence of noise.


Noise
Noise
Noise

	Hit@1 w/o noise	Hit@1 w/ noise	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22
UNITE	1.00	1.00	0.028	0.966	0.965	0.242	0.750	0.750	0.630	0.943	0.941	0.623	0.949	0.948	0.573	1.000	0.998	0.573	0.999	0.998	0.000	0.994	0.993	0.999
AdverserialLoss	1.00	0.66	0.000	0.719	0.715	0.148	0.570	0.565	0.470	0.728	0.724	0.428	0.770	0.771	0.325	0.876	0.872	0.202	0.998	0.996	0.225	0.495	0.490	1.000
MarginRankingLoss	1.00	0.50	0.464	0.492	0.373	0.366	0.655	0.311	0.554	0.606	1.000	0.818	0.767	0.759	0.594	0.250	0.870	0.000	0.434	0.499	0.392	0.664	0.836	0.487

Fig. 2. Loss functions and noise with repeated patterns of Fig. 1.

The key aspects of the problem are relational patterns on the one hand, and noise on the other hand. One simple example of noise is shown through the red edge in Fig. 1, which does not hold in the model world, but is present in our data set. On the upper side of Fig. 1, we see the relational patterns of our model world where (a) and (b) are composition, and (c) antisymmetry. Figure 2 shows ranks (Hit@1) of the triples from TransE model for this scenario with three loss functions including ours. The second and third columns are the setting without and with noise, respectively. In Fig. 2, we see the three edges marked as noise, i.e., incorrect edges. We record the resulting classification as correct (blue) and incorrect (red). While there are differences between the scores for our three noise tuples with the three loss functions, one could still consider them adequate, as the classification is in all cases negative. That is, for edges that represent noise, the scores are not truly problematic. The reason for the lower evaluation metrics becomes apparent on those edges shown in Fig. 2 such as T3 or T16 which do not represent noise, but are related to noise via relational patterns. We see in these cases that the scores under margin ranking loss are in most cases the worst, followed by adversarial loss, while UNITE loss is clearly the least affected. Intuitively, UNITE dampens the effect of noise propagating along relational patterns. This is in particular highlighted when looking at the lower left part of Fig. 1 (where scores under the adversarial loss are annotated above edges) comparing it to the lower right part of Fig. 1 (where scores under the UNITE loss are annotated above edges).

3 Related Work

Here, we review highlights of related contributions considering both the score and loss functions of models which fall in the same category as ours, namely distance models, i.e., translation-based, or rotation-based models.

Score Functions. The score function of a KGE model ($f_r(s, o)$) takes the embeddings of a triple, i.e., $(\mathbf{s}, \mathbf{r}, \mathbf{o})$ and returns a value – often denoted as $f^r_{s,o}$ – indicating the extent to which a triple is plausible in the embedding space. We consider one baseline (TrasnE) from translation-based embedding model and one state-of-the-art (RotatE) rotation-based KGE model.

TransE [1] model takes a vector representation of a triple $(\mathbf{s}, \mathbf{r}, \mathbf{o})$ and models its relation $(\mathbf{r})$ as a translation from subject $(\mathbf{s})$ to object $(\mathbf{o})$, i.e., $\mathbf{s} + \mathbf{r} \approx \mathbf{o}$. The score function of TransE is formulated as $\|\mathbf{s} + \mathbf{r} - \mathbf{o}\|$.

RotatE [21] forces $\mathbf{s}_j \mathbf{r}_j \approx \mathbf{o}_j$ to hold for all $j \in \{0, \dots, d\}$ per each given triple $(\mathbf{s}, \mathbf{r}, \mathbf{o})$. The model performs a rotation of the j-th element s_j of the subject vector $\mathbf{s}$ by the j-th element $r_j = e^{i\theta_{r_j}}$ of a relation vector $\mathbf{r}$ to get the j-th element o_j of the object vector $\mathbf{t}$, where θ_{r_j} is the phase of the relation r.

Loss Functions. The loss function of a KGE ($\mathcal{L}$) is an optimization to adjust the embedding vectors. The losses of TransE and RotatE models have been reported strongly outperforming the other possible KGEs [19,21].

Margin Ranking Loss (MRL) has been primarily designed for training TransE and its variants. The loss optimization aims to put a margin (inspired by SVM [2]) between each positive sample (s, r, o) and its corresponding negative sample (s', r, o'), obtained by corruption in s, or o [1]. Let γ is a margin and $[x]_+ = max(0, x)$. MRL is defined as $\mathcal{L} = \sum_{(s,r,o)\in S^+} \sum_{(s',r,o')\in S^-_{(s,r,o)}} [\gamma + f_r(s, o) - f_r(s', o')]_+$. Despite the major use of MRL, it suffers from the margin sliding problem [15]. Therefore, there are many solutions which do not fulfil the translation, i.e., $\mathbf{s} + \mathbf{r} \neq \mathbf{o}$ [26]. The loss of the RotatE model is called *Adversarial Loss* which is defined as $\mathcal{L} = -\sum_{(s,r,o)\in S^+} \Big(\log \sigma(\gamma - f_r(s, o)) + \sum_{(s',r,o')\in S^-} p(s', r, o') \log \sigma(f_r(s', o') - \gamma) \Big)$, where $\sigma(.)$ is the Sigmoid function, $p(s', r, o') = \frac{exp(\alpha f_r(s',o'))}{\sum exp(\alpha f_r(s',o'))}$ is the probability of the triple (s', r, o') to be true negative, and α is the temperature of sampling. Other loss functions are also designed for particular usages [14,15].

Noise and KGE Models. The sensitivity of KG embeddings to sparse and unreliable data is discussed in [16] without considering noise and relational patterns. GTransE [11] deals with uncertainty (meaning triple incorrectness) on KGs using dynamic and static weighting. In a different work, puTransE [22] proposes an approach to make KGEs semantically and structurally aware of noise, but it did not consider loss functions in particular. Node similarity Preserving (NSP) [17] proposes a loss function without considering multi-relational KGs. Graph Denoising Policy Network (GDPNet) [23] focuses on an inductive approach from reinforcement learning on noise in scholarly KGs. Most of the other works have been focused on negative sampling, or feature selection for noise detection [12,20,25]. We focus on the role of loss functions in the existence of incorrect triples as noisy data and considers relational patterns.

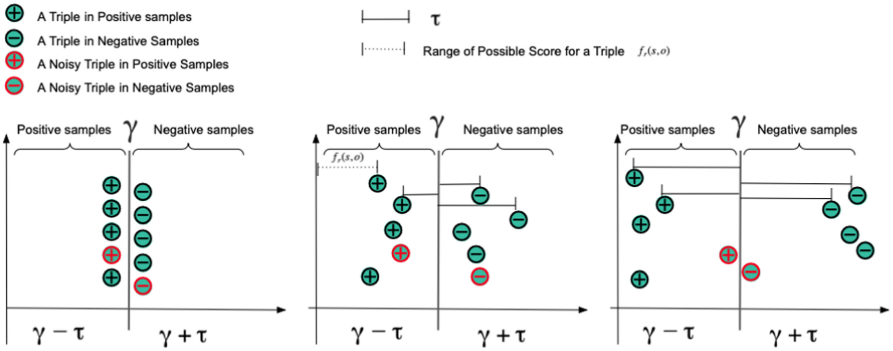

A Triple in Positive samples
A Triple in Negative Samples
A Noisy Triple in Positive Samples
A Noisy Triple in Negative Samples
τ
Range of Possible Score for a Triple $f_r(s,o)$
Positive samples
γ
Negative samples
$f_r(s,o)$
γ − τ
γ + τ

Fig. 3. Learning steps of UNITE. Initial state (left), intermediate state (middle) and final state (right).

Relational Patterns and KGE Models. Early literature conjectured KGE models are evaluated in rules encoding [3,4,9,18]. Only some rule injection frameworks such as Ruge [8], KALE [7] and few embedding models such as RotatE [21] and its special case TorusE [5] have considered the issue of relational patterns. Specially, a recent work has theoretically and empirically proven that the score function of the RotatE model is capable of inferring various relational patterns including symmetric/antisymmetric, inverse and composition patterns. This inference ability is when for any pattern in the form of *premise* → *conclusion*, the model approves correctness of *conclusion* when the correctness of *premise* is confirmed. Overall, the existence of noise in KGs in the context of relational patterns has not been addressed.

4 Method

Given a KGE model, a typical optimization framework for link prediction consists of the following steps: (1) initialization of embedding vectors, (2) setting criteria (e.g., margin, square error), and (3) optimizing a loss function in an iterative way to enforce that embedding vectors satisfy the criteria. In order to unify the power of implicit patterns shaped inside the underlying KG and mitigate the negative effect of noise, we adapt these steps in a proposed optimization framework named UNITE model.

In step (2), i.e., setting criteria, we consider a point $\gamma \in [0, 1]$ as discriminator for separation of positive and negative samples. Relative to γ, the correctness/incorrectness of a triple (s, r, o) is measured by its distance $(\tau^r_{s,o})$ to the discriminator. Note that $\tau^r_{s,o}$ is initialized to zero originally, and during the learning process will, in general, increase. UNITE aims at adjusting the distance as well as embedding vectors in an iterative process to reduce the degree of correctness of implicit noisy triples while the model learns from patterns that the triple is wrongly labeled as positive or negative.

To illustrate the process of step (3), i.e., optimization, we introduce Fig. 3 which guides us through this method section, and the components of which we will introduce step by step in this section. Overall, we see three states of the learning process: On the left side of Fig. 3 the initial state, on the right side the final state and an intermediate state in the middle. The boundary γ, illustrated in each state in the figure, separates positive and negative samples. The distance to γ is given by $\tau^r_{s,o}$, and illustrated in Fig. 3. In the rest of this section, We first design a loss function and explain the process to perform optimization for positive samples. This will allow us to understand the areas to the left of γ in Fig. 3. The same procedure is explained for negative samples, providing an intuition for the areas to the right of γ. Finally, the two designed parts are united by proposing the optimization framework UNITE, which allows us to understand Fig. 3.

4.1 UNITE Loss for Positive Samples

As briefly introduced in the previous section, apart from the embedding itself, the primary values to be optimized during the optimization phase is the distance $\tau^r_{s,o}$. We will now describe, bottom-up, how the loss function and the optimization problem is defined for positive samples. Let us first define domain and range of $\tau^r_{s,o} : S^+ \to [0, \infty]$. More specifically, by constraints of the optimization problem introduced later, the effective range of $\tau^r_{s,o}$ will actually be constrained to be $[0, \gamma]$. Within our framework, we will apply a probability function P to $\tau^r_{s,o}$, and use the notation $P^r_{s,o} = P(\tau^r_{s,o})$. The specific choice of such a probability function is up to the user of the framework. In our evaluation, we use a Gaussian function with the variance optimized as a hyper-parameter. Precise definitions will be given in the evaluation section, where experiments for particular configurations of the framework are performed. Intuitively, values $P^r_{s,o}$ for positive samples have the following meaning: 1) a triple with $P^r_{s,o} = 0$, has the highest probability of being correct ("positive"); 2) a triple with $P^r_{s,o} = 1$, has the lowest probability of being correct ("unknown"); 3) a triple with $P^r_{s,o}$ between 0 and 1 describes the extent to which a triple is considered as positive or unknown. In the beginning of the learning process, $\tau^r_{s,o}s$ are randomly assigned to a very small value ($\tau^r_{s,o}s \approx 0$).

Therefore, $P^r_{s,o}s$ are very high in the beginning of the learning process ($P^r_{s,o} \approx 1$). The ultimate objective is to minimize the loss function. The embedding vectors and $\tau^r_{s,o}s$ are optimized in an iterative process. We define the *loss function* as:

$$\mathcal{L}^+ = \prod_{(s,r,o)\in S^+} P^r_{s,o} \tag{1}$$

where S^+ is the set of all triples in the KG. We define the *objective function of the optimization problem* as follows:

$$\min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \mathcal{L}^+ \tag{2}$$

where $\mathbf{S}^+$ is the embedding of all entities and relations. We apply a second probability function $\mathcal{Q}$ for the purpose of defining constraints for our optimization problem. The specific choice of such a probability function is again up to the user of the framework. In our evaluation, we use a Sigmoid function. As before, precise definitions will be given in the evaluation section. We now give the *constraint* of our optimization problem

$$\mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}). \tag{3}$$

Observe that this constraint effectively limits $\tau^r_{s,o}$ to be no larger than γ, and it forces the score $f^r_{s,o}$ to be in the range $[0, \gamma]$ as well. This yields the following optimization problem using the objective function from Eq. 2 and the constraint from Eq. 3:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \prod_{(s,r,o)\in S^+} P^r_{s,o}\,, \\ \text{s.t. } \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}). \end{cases} \tag{4}$$

Considering the fact that $\min \mathcal{L}$ is equivalent to $\min \log(\mathcal{L})$, the following optimization problem is solved instead of Eq. 4:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \sum_{(s,r,o)\in S^+} \log P^r_{s,o}\,, \\ \text{s.t. } \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}). \end{cases} \tag{5}$$

This essentially makes the mathematical operations applied simpler, while still solving the same optimization problem.

4.2 UNITE Loss for Negative Samples

Existing KGE models mostly generate negative samples by corruption of positive samples. In this paper, we consider one of the simplest corruption techniques namely *uniform negative sampling* used in [1]. To this end, either subject (s) or object (o) of a given positive triple (s, r, o) is replaced by an entity (s' or o'). A candidate entity (s' or o') is selected randomly using uniform distribution. Let the set $S^-_{(s,r,o)}$ denotes all such corruptions of the triple (s, r, o), and let S^- denote the overall set of all the negative samples. Due to randomness of negative sample generation, there is always uncertainty in the negative samples. The definition of the optimization problem for negative samples follows similar principles as the one described before for positive samples: 1) a triple with $P^r_{s',o'} = 0$, has the highest probability of being incorrect ("negative"); 2) a triple with $P^r_{s',o'} = 1$, has the lowest probability of being incorrect ("unknown"); 3) a triple with $P^r_{s',o'}$ between 0 and 1 describes the extent to which a triple is considered as negative or unknown. The constraint for negative samples is $\mathcal{Q}(f^r_{s',o'} - \gamma) \geq \mathcal{Q}(\tau^r_{s',o'})$ where S^- is the set of all negative samples and $\mathbf{S}^-$ is the set of all embeddings of entities and relations participating in the negative sample set S^-.

4.3 United Optimization

We now merge the two optimization problems previously formulated for positive and negative samples, completing the overall situation illustrated in Fig. 3. There are two possible assumptions for uniting the formulations of positive and negative samples, namely: independent uncertainty and dependent uncertainty.

Independent Uncertainty: this assumption indicates that although a negative sample (s', r, o') is generated by corruption of either subject or object of the positive sample (s, r, o), the degrees of uncertainty for positive and negative samples are independent (UNITE-I). Therefore, we set two different parameters for the positive and its negative sample i.e., $\tau^r_{s,o}$ for positive and $\tau^r_{s',o'}$ for negative samples. This can be achieved by simply combining the two optimization problems we introduced so far without any further modification, yielding the following optimization problems:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\{(\mathbf{s}',\mathbf{r},\mathbf{o}')\}\in\mathbf{S}^-,\tau^r_{s,o},\tau^r_{s',o'}} \\ \qquad \sum_{(s',r,o')\in S^-} \log P^r_{s',o'} + \sum_{(s,r,o)\in S^+} \log P^r_{s,o} \\ \text{s.t. } \mathcal{Q}(f^r_{s',o'} - \gamma) \geq \mathcal{Q}(\tau^r_{s',o'}) \ , \{(s',r,o')\} \in S^-, \\ \qquad \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}), \ \{(s,r,o)\} \in S^+. \end{cases} \tag{6}$$

Dependent uncertainty: we set the same parameters for the positive and its negative sample to measure the degree of uncertainty dependently. Therefore, $\tau^r_{s,o}$ is used for both of the positive and its corresponding negative samples (UNITE-D). The formulation of this optimization is:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \sum_{(s,r,o)\in S^+} \log P^r_{s,o} \, , \\ \text{s.t. } \mathcal{Q}(f^r_{s',o'} - \gamma) \geq \mathcal{Q}(\tau^r_{s,o}) \, , \\ \{(s',r,o')\} \in S^-_{(s,r,o)}, \ \{(s,r,o)\} \in S^+, \\ \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}), \ \{(s,r,o)\} \in S^+ \end{cases} \tag{7}$$

For both positive and negative samples, $\tau^r_{s,o}$ of the positive samples is used. Finally, in order to solve the optimization problems given by Eqs. 6 and 7, we bring the constraints to the objectives, as is usually done, and solve the unconstrained optimization problems using stochastic gradient descent.

The Role of $\boldsymbol{\tau}$. Let us focus on the negative samples in Fig. 3 which are distributed in the right side of γ. This is effectively enforced by the constraint introduced in negative constraint. which consequently enforce the eventual scores of negative samples to be bigger than γ. To understand the role of τ in determining the plausibility of triples in presence of noise, for example let r be a symmetric rule in the form of *premise* $\Longleftrightarrow$ *conclusion*, where *premise*, (a triple) and *conclusion* (a triple) have a common relation with different entities in head and tail positions i.e. $(s, r, o) \Longleftrightarrow (o, r, s)$. If we exemplify this on *colleagueOf* relation, then $(S, colleagueOf, O) \Longleftrightarrow (O, colleagueOf, S)$ In case $(S, colleagueOf, O)$ is a correct triple in a KG, then the plausibility of $(O, colleagueOf, S)$ can be defined with different conditions: (a) it is a correct triple in positive samples, or (b) it is not in positive samples nor in negative samples, or (c) it is in negative samples (false negative) which creates a conflict with what the model is enforced to learn. Here we focus on case (c) when $(O, colleagueOf, S)$ is a false negative sample (noise) in the training set. If we use RotatE for this case (as one of the best reported distance-based model) with the score function $f^r_{s,o} = \|\mathbf{s} \circ e^{\theta_\mathbf{r}} - \mathbf{o}\|$ which is proven to be capable of inferring symmetric relations when $\theta_r = 0$ or π [21], therefore the triple $(O, colleagueOf, S)$ will be learned (inferred) as a positive triple. This poses a conflict between what the model infers about the plausibility of this triple from the patterns (positive), and what the model sees about the plausibility of this triple in the training set (negative). Here τ comes into play with an important role to resolve this conflict by giving the high uncertainty value to the false negative sample $(O, colleagueOf, S)$. For simplicity of explanation, let $\mathcal{Q}$ be a linear function (in negative constraint) and $\gamma = 0$, so we have $f^{colleagueOf}_{(O,S)} \geq \tau^{colleagueOf}_{(O,S)}$. Since $\theta_r = 0$ or π and $(S, colleagueOf, O)$ is positive, $f^{colleagueOf}_{(O,S)} \approx 0$ (is positive) , therefore, $\tau^{colleagueOf}_{(O,S)}$ is constrained to be close to zero (is positive). Therefore, the main optimization problem of Eq. 6 considers $\tau \approx 0$ as an optimal solution. Therefore, the triple gets high uncertainty value based on τ i.e. $P^{colleagueOf}_{O,S} \approx 1$. Giving such high uncertainty to $(O, colleagueOf, S)$ enables the model to keep θ_r still close to 0 or π which preserves the ability of inferring a symmetric pattern. The opposite of this scenario is the case where $(S, colleagueOf, O)$ gets the score to be negative by the model because $(O, colleagueOf, S)$ is in the negative set and r is a symmetric relation ($\theta_r = 0, \pi$). However, this scenario is less likely to happen because there are other patterns and triples that by using them the model recognizes $(S, colleagueOf, O)$ as positive during learning process.

5 Evaluation

In this section, we evaluate the UNITE framework in the context of two other loss functions: margin ranking loss (baseline) and adversarial loss (state-of-the-art).

Evaluation Metrics. Three statistic measurement metrics are considered: Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits@K. In order to calculate MR, a corrupted version of the test set is created: (1) replacing the subject of triples by possible other entities, and (2) replacing the object of triples by all possible entities. For each triple (s, r, o) in the test set S, a sample set S_i for the i^{th} test triple is generated such that $S_i^s = (?, r, o), ? \in KG$. The same holds for the second round. Scores of all the generated triples (including the test triple (s, r, o)) are computed and sorted in S_i^s and S_i^o. Let $Rank_i = (Rank_i^s + Rank_i^o)/2$ denote the general rank of the i-th triples, the Mean Rank (MR) is computed as $MR = \frac{\sum Rank_i}{n_t}$ where n_t is the number of test triples. Hits@K is the rate of correct triples appearing in top K position. The filtered setting [1] is used for evaluation of our model.

Datasets. We used four standard benchmarks (statistics in Table 1) with the assumption that they contain implicit noise. AMIE [6] was used for rule mining. We contaminate FB15K by more than 100,000 random corrupted triples (about 20% noise) to evaluate the ability of UNITE to recognize and be resilient to noise. We keep the ratio relatively high (20%) however for other datasets, we stayed with lower ratio because it was compatible with the statistics of patterns in the whole dataset. The same ratio of noise is considered for WN18. For FB15K-237, we generate 5% random noise.

Table 1. Dataset statistics. Number of triples and patterns.

Dataset	Inv.	Imp.	Eql.	Sym.	#train	#valid.	#test
FB15K	67,757	3,259	8,771	7,740	483,142	50,000	59,071
FB15K-237	4,645	578	861	–	272,115	17,535	20,466
WN18	116,464	–	–	–	141,442	5,000	5,000
WN18RR	–	–	–	–	6,084	3,034	3,134

Training Setting. We train TransE and RotatE using optimization framework Eq. 6 (UNITE-I) and Eq. 7 (UNITE-D). In our experiments, we use $P_{s,o}^r = \exp -\sigma {\tau_{s,o}^r}^2$ (Gaussian) and $\mathcal{Q}(x) = \frac{1}{1+\exp -x}$ (Sigmoid). The Sigmoid function is strictly monotone, therefore we use a linear function ($\mathcal{Q}$) instead to enforce the constraints. The embedding dimension d is set to be 200 and 10 negative sample are generated ($n = 10$). The hyper-parameter of the Gaussian function (σ) is fixed to 1000. The batch size is set to 1024 for training on FB15k and FB15k-237, and 512 for WN18 and WN18RR. The corresponding values for γ is from the set $\{0, 5, 10, 15, 20, 25, 30\}$.

5.1 Experimental Results

The results of our experiments have been divided into multiple parts. First, we evaluate UNITE with and without artificially generated noise. We additionally report the results of different models/losses with/without pattern injection. We also inject patterns (only for inverse) by adding a regularization term to

Table 2. Evaluation Results. Comparisons of results for Adversarial Loss, Margin Ranking Loss and UNITE are depicted for FB15k and FB15k-237, and WN18.

Dataset				FB15K		FB15K-237		WN18	
				MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
Adv. Loss	w/o noise	w/ inject	RotatE	73.3	87.9	32.1	50.8	-	-
			TransE	66.7	86.3	30.8	50.3	-	-
		w/o inject	RotatE	72.3	87.8	32.3	51.2	94.9	96.3
			TransE	68	86	31.3	51.3	70.7	95.2
	w/- noise	w/ inject	RotatE	63.3	82.8	31.2	51.1	-	-
			TransE	32.3	75.7	30.7	50.2	-	-
		w/o inject	RotatE	61.2	81.9	32.3	51.2	94.8	96.2
			TransE	32.2	74.8	31.3	49.8	69.8	94.7
MR Loss	w/o noise	w inject	RotatE	62.3	81.1	-	-	-	-
			TransE	45.8	74.3	-	-	-	-
		w/o inject	RotatE	60.8	80.7	27.9	47.3	94.2	94.3
			TransE	46.9	74.3	27.7	47.3	50.1	94.8
	w/- noise	w/ inject	RotatE	52.3	72.8	-	-	91.9	93.8
			TransE	33.3	61.2	-	-	50.4	94.8
		w/o inject	RotatE	50.3	72.7	27.9	46.8	92.3	94.2
			TransE	32.2	60.1	26.7	47.9	45.2	95.2
UNITE Loss	w/o noise	w/ inject	RotatE	74.3	88.8	33.3	51.9	-	-
			TransE	67.9	86.8	31.1	51.2	-	-
		w/o inject	RotatE	73.3	88.8	21.2	50.8	95.3	96.2
			TransE	69.3	86.3	31.3	51.1	75.9	95.8
	w/- noise	w/ inject	RotatE	64.2	83.8	33.3	51.8	-	-
			TransE	33.3	76.2	30.9	50.8	-	-
		w/o inject	RotatE	61.1	83.3	32.2	51.8	94.9	95.8
			TransE	32.2	76.2	30.9	51.2	74.8	95.8

the objective as in [13]. As shown in Table 2, UNITE achieves improvements in FB15K and FB15K-237 in terms of MRR and Hits@10 with and without noise. UNITE achieves an MRR of 74 for FB15K whereas MRL and adversarial loss achieve 62 and 73, respectively. The relatively good results of adversarial loss is due to its noise resiliency ability, however our model is particularly designed for being pattern-aware in presence of noise. In our evaluation setup with randomly generated noise, UNITE significantly outperforms margin ranking loss by more than 10%. As visible, the results of the RotatE model when trained with adversarial loss stay very close to UNITE. Our assessment for these results can be summarized as three points: 1) **lack of diversity in types of rules**: despite the existence of patterns in FB15K, it lacks diversity of patterns; 2) **relatively small ratio of other pattern types to inverse**: the majority of the patterns are inverse relations and the rest belong to symmetric, implication and equivalence patterns Table 1; 3) **test set leakage**: not only the diversity but also the ratio of overall grounding of patterns has dropped dramatically in the case of FB15K-237. Considering the above points, of particular interest for future work will be in-depth studies on the existence of complex patterns, and advances in the area of pattern extraction, especially focusing on complex patterns. Both of the optimizations, UNITE-I and UNITE-D, perform closely – in all the previous evaluations, we only reported UNITE-I. UNITE-D gets a lower MR in both datasets of WN18 and FB15k with same result in Hits@1 on WN18. In FB15k for Hits@1, UNITE-I gets 82.2% while UNITE-D is performing with 81.7%. In Table 2 we reported the relevant results for WN18. TransE trained by UNITE obtains 76% on MRR, which significantly outperforms TransE with adversarial loss with MRR of 71%. In the case of WN18RR, as discussed before, the rule mining system that we used, AMIE, did not extract any patterns. Despite the pattern-free characteristic of WN18RR, RotatE trained with UNITE was able to achieve 57% in hits@10 whereas MRL could only reach 37%.

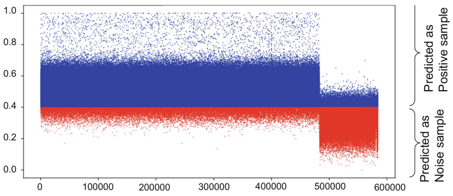


Fig. 4. Distribution of scores in FB15K.

Correctness Prediction. In Fig. 4, we illustrate the distribution of scores predicted by UNITE for FB15K. The X axis shows the index of triples and the Y axis presents normalized scores. Predicted correct triples (in blue) by UNITE are separated on 0.4 from predicted noisy triples (in red). Out of 483,142 triples in the train set of FB15k, 101,123 triples have been made noisy through a random procedure (21%). As shown, our model is able to assign a low score to most of the 21% generated noise triples. The triples in the upper right (in blue) are assumed to fall in this category. The lower left part of the plot shows the existence of noise (in red) in the positive samples. We manually validated several triples in Table 3.

Table 3. Validation test shows correct (C) and noise (N) triples identified by UNITE.

Triple/Wikipedia	Prediction
Harvey_Weinstein($/m/05hj_k$) isSiblingOf Bob_Weinstein($/m/06q8hf$)	1.0 (Originally Positive - Identified Positive
- ($/m/07s9rl0$) hasSameGenre ($/m/06fvc$)	0.169 (Originally Positive - Identified Noise)

6 Conclusion

In this paper, we investigated KGE models in the presence of both relational patterns and noise. We introduced the new loss function UNITE and its variations UNITE-I and UNITE-D. We evaluated UNITE both within translation-based models (TransE) and rotation-based models (RotatE), and in synthetic and real-world scenarios. As future work, we plan to further investigate the effect of non-uniformly distributed random noise. This will need some advances in the area of detecting and extracting more complex relational patterns than current methods can do, but would be able to shed more light on the situation in which noise is specifically affecting patterns in large real-world scenarios.

Acknowledgements. We acknowledge the support of the EU projects TAILOR (GA 952215), Cleopatra (GA 812997), the BmBF project MLwin, ScaDS.AI (01/S18026A-F), WWTF (Vienna Science and Technology Fund) grant VRG18-013, the EPSRC grant EP/M025268/1, and the EU Horizon 2020 grant 809965.

References

1. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, pp. 2787–2795 (2013)
2. Cortes, C., Vapnik, V.: Support vector machine. Mach. Learn. **20**(3), 273–297 (1995)
3. Du, J., Qi, K., Shen, Y.: Knowledge graph embedding with logical consistency. In: Sun, M., Liu, T., Wang, X., Liu, Z., Liu, Y. (eds.) CCL/NLP-NABD -2018. LNCS (LNAI), vol. 11221, pp. 123–135. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01716-3_11
4. Du, J., Qi, K., Wan, H., Peng, B., Lu, S., Shen, Y.: Enhancing knowledge graph embedding from a logical perspective. In: Wang, Z., Turhan, A.-Y., Wang, K., Zhang, X. (eds.) JIST 2017. LNCS, vol. 10675, pp. 232–247. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70682-5_15
5. Ebisu, T., Ichise, R.: Toruse: knowledge graph embedding on a lie group. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
6. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: Amie: association rule mining under incomplete evidence in ontological knowledge bases. In: Proceedings of the 22nd international conference on World Wide Web, pp. 413–422 (2013)
7. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 192–202 (2016)
8. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Knowledge graph embedding with iterative guidance from soft rules. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
9. Gutiérrez-Basulto, V., Schockaert, S.: From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. In: Sixteenth International Conference on Principles of Knowledge Representation and Reasoning (2018)
10. Heindorf, S., Potthast, M., Stein, B., Engels, G.: Vandalism detection in wikidata. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 327–336 (2016)
11. Kertkeidkachorn, N., Liu, X., Ichise, R.: GTransE: generalizing translation-based model on uncertain knowledge graph embedding. In: Ohsawa, Y., Yada, K., Ito, T., Takama, Y., Sato-Shimokawara, E., Abe, A., Mori, J., Matsumura, N. (eds.) JSAI 2019. AISC, vol. 1128, pp. 170–178. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-39878-1_16
12. Luo, S., Fang, W.: Potential probability of negative triples in knowledge graph embedding. In: Cheng, L., Leung, A.C.S., Ozawa, S. (eds.) ICONIP 2018. LNCS, vol. 11303, pp. 48–58. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04182-3_5

13. Minervini, P., Costabello, L., Muñoz, E., Nováček, V., Vandenbussche, P.-Y.: Regularizing knowledge graph embeddings via equivalence and inversion axioms. In: Ceci, M., Hollmén, J., Todorovski, L., Vens, C., Džeroski, S. (eds.) ECML PKDD 2017. LNCS (LNAI), vol. 10534, pp. 668–683. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71249-9_40
14. Nayyeri, M., Vahdati, S., Zhou, X., Shariat Yazdi, H., Lehmann, J.: Embedding-based recommendations on scholarly knowledge graphs. In: Harth, A., Kirrane, S., Ngonga Ngomo, A.-C., Paulheim, H., Rula, A., Gentile, A.L., Haase, P., Cochez, M. (eds.) ESWC 2020. LNCS, vol. 12123, pp. 255–270. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49461-2_15
15. Nayyeri, M., Zhou, X., Vahdati, S., Izanloo, R., Yazdi, H.S., Lehmann, J.: Let the margin slide±for knowledge graph embeddings via a correntropy objective function. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–9. IEEE (2020)
16. Pujara, J., Augustine, E., Getoor, L.: Sparsity and noise: where knowledge graph embeddings fall short. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1751–1756 (2017)
17. Qiu, Z., Hu, W., Wu, J., Tang, Z., Jia, X.: Noise-resilient similarity preserving network embedding for social networks. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 3282–3288. AAAI Press (2019)
18. Du, J.: Ranking diagnoses for inconsistent knowledge graphs by representation learning. In: Ichise, R., Lecue, F., Kawamura, T., Zhao, D., Muggleton, S., Kozaki, K. (eds.) JIST 2018. LNCS, vol. 11341, pp. 52–67. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04284-4_4
19. Ruffinelli, D., Broscheit, S., Gemulla, R.: You {can} teach an old dog new tricks! on training knowledge graph embeddings. In: International Conference on Learning Representations (2020)
20. Shan, Y., Bu, C., Liu, X., Ji, S., Li, L.: Confidence-aware negative sampling method for noisy knowledge graph embedding. In: 2018 IEEE International Conference on Big Knowledge (ICBK), pp. 33–40. IEEE (2018)
21. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)
22. Tay, Y., Luu, A.T., Hui, S.C.: Non-parametric estimation of multiple embeddings for link prediction on dynamic knowledge graphs. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
23. Wang, L., et al.: Learning robust representations with graph denoising policy network. arXiv preprint arXiv:1910.01784 (2019)
24. Xie, R., Liu, Z., Lin, F., Lin, L.: Does william shakespeare really write hamlet? knowledge representation learning with confidence. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
25. Zhao, Y., Liu, J.: Scef: a support-confidence-aware embedding framework for knowledge graph refinement. arXiv preprint arXiv:1902.06377 (2019)
26. Zhou, X., Zhu, Q., Liu, P., Guo, L.: Learning knowledge embeddings by combining limit-based scoring loss. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1009–1018. ACM (2017)

TLS-Covid19: A New Annotated Corpus for Timeline Summarization

Arian Pasquali[1](✉), Ricardo Campos[1,2], Alexandre Ribeiro[1], Brenda Santana[1], Alípio Jorge[1,3], and Adam Jatowt[4]

[1] LIAAD – INESCTEC, Porto, Portugal
{arian.r.pasquali,ricardo.campos,alexandre.m.ribeiro, brenda.s.santana}@inesctec.pt

[2] Polytechnic Institute of Tomar, Ci2 - Smart Cities Research Center, Tomar, Portugal
ricardo.campos@ipt.pt

[3] FCUP, University of Porto, Porto, Portugal
amjorge@fc.up.pt

[4] University of Innsbruck, Innsbruck, Austria
adam.jatowt@uibk.ac.at

Abstract. The rise of social media and the explosion of digital news in the web sphere have created new challenges to extract knowledge and make sense of published information. Automated timeline generation appears in this context as a promising answer to help users dealing with this information overload problem. Formally, Timeline Summarization (TLS) can be defined as a subtask of Multi-Document Summarization (MDS) conceived to highlight the most important information during the development of a story over time by summarizing long-lasting events in a timely ordered fashion. As opposed to traditional MDS, TLS has a limited number of publicly available datasets. In this paper, we propose TLS-Covid19 dataset, a novel corpus for the Portuguese and English languages. Our aim is to provide a new, larger and multi-lingual TLS annotated dataset that could foster timeline summarization evaluation research and, at the same time, enable the study of news coverage about the COVID-19 pandemic. TLS-Covid19 consists of 178 curated topics related to the COVID-19 outbreak, with associated news articles covering almost the entire year of 2020 and their respective reference timelines as gold-standard. As a final outcome, we conduct an experimental study on the proposed dataset over two extreme baseline methods. All the resources are publicly available at https://github.com/LIAAD/tls-covid19.

Keywords: Timeline summarization · Datasets · Evaluation

1 Introduction

Following media coverage of long-lasting events like wars, epidemics or economic crises is demanding for readers, journalists, specialists and scholars. How did the S.A.R.S. epidemic crisis evolve in the early 2000s? What are the similarities with modern events? One common solution to this problem that can offer answers to the above-mentioned

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 497–512, 2021.
https://doi.org/10.1007/978-3-030-72113-8_33

example questions is the adoption of timelines to support storytelling as a method to organize the different phases of complex events. For instance, media outlets frequently use timelines to illustrate stories. However, manually building such timelines can be very laborious and time-consuming even with the support of modern search engines. Understanding the evolution and implications of these events often requires a combination of tools and search queries. Timeline summarization systems (TLS) emerge in this context as an alternative to manually digesting huge volumes of data in a short period of time by offering the possibility of creating summaries of multiple documents over time.

The recent surge of the COVID-19 outbreak is a very up-to-date example of this information overload problem exerting tremendous effort and pressure on users who want to keep up with the news. By January 20th 2021, the novel COVID-19 has been reported in 219 countries; resulting in approximately 100M confirmed cases and more than 2M deaths[1] Fighting this pandemic situation requires isolation, social distance measures, research in health and medicine care, but also contributions from the research community. The Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) was one of the firsts to make available a data repository[2] and a visual dashboard that gathers information from multiple sources. Multiple other similar initiatives have also been established worldwide. The Coronavirus Corpus[3], first released in May 2020 and currently 814M of words has also been created to shed light on what people are saying in online newspapers and magazines. Perhaps, the most widely known initiative to date was the release of the COVID-19 Open Research Dataset (CORD-19)[4]. Created by the Allen Institute for AI in partnership with five other institutes, CORD-19 [32] consists of over 158,000 scholarly articles, including over 75,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses and has fostered the emergence of multiple solutions. This is the case of the TREC-COVID challenge [30], which uses the CORD-19 dataset to build a set of Information Retrieval (IR) test collections. Aiming to support the fight against this pandemic Alam et al. [1] has also manually annotated a dataset of COVID-19 related tweets to tackle the problem of disinformation. These datasets were already applied to a variety of NLP tasks such as question answering and abstractive summarization [15]. Similarly, Yang et al. [34] developed a dialog dataset containing conversations between patients and doctors about COVID-19 to support chatbots research. Timelines can also be understood in this context as an essential resource for readers of major news outlets to quickly have access to a concise view of a given topic over time. A good temporal summary of the "*World Health Organization*" topic over the recent months should refer, for instance, to the chronological evolution of the COVID-19 outbreak, possible vaccine solutions, or the Donald Trump's ultimatum to WHO on May 2020, among many other summaries.

While several methods have been proposed to generate condensed news timelines, the problem of timeline generation is yet to be solved. One of the reasons for this is that traditional TLS datasets are restricted to just a limited number of topics [29]. However, deeply understanding long-lasting events, as is the case of COVID-19, requires a

[1] https://www.worldometers.info/coronavirus/countries-where-coronavirus-has-spread/.

[2] https://github.com/CSSEGISandData/COVID-19.

[3] https://www.english-corpora.org/corona/.

[4] https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge.

significantly larger number of topics, news articles, different sources, annotated timelines, and longer time spans. Previous work on TLS also does not make it clear on how proposed methods behave across different languages. This makes it hard to assess how the methods behave under different scenarios, since almost all the datasets, with few exceptions, are for the English language, and none is multi-lingual. Sorting out these questions is crucial for researchers who lack diversified datasets to evaluate their proposed algorithms. Addressing the issues mentioned above, requires significant efforts in terms of: (1) collecting manually edited timelines from credible news sources; (2) collecting timelines and news articles, relevant to Covid-19 both in temporal and textual dimensions; and (3) selecting a representative and diversified number of topics.

TLS-Covid19 corpus emerges in this context to promote the development and the evaluation of new algorithms and applications in the context of the timeline summarization task, and at the same time, to enable the study of news coverage about the COVID-19 pandemic, from the evolution of a topic over time, to the comparison of what is being said about a certain topic by different news outlets. One can also look at keywords, part-of-speech tags, entities or events to see how things have changed over time. It also opens room to look at collocates. A few examples might be: keywords that were common in the same time-period, words that appear near covid-19 in different time-periods, entities, events, nouns or verbs that were more common at the beginning of the pandemics but no longer on December 2020. Finally, as it is common in most of the datasets of this kind, researchers are also offered the chance to create a sub-set of the dataset based on the publication date, the source, the country, etc., and to apply it for different purposes than the one it was initially designed for. Our corpus consists of 178 topics (35 in English and 143 in Portuguese), their associated 100,399 news articles (32,210 in English and 68,508 in Portuguese), and 178 timelines (one for each of the 178 topics). Note, however, that we have considered two news sources per language, each with its timeline, which accounts for 356 timelines. This opens room for researchers two evaluate their systems under two different scenarios. One that considers an evaluation over the news sources, based on the fact that each one has its ground-truth timeline. The other one which considers an evaluation solely over the languages, which could be made possible by a slight modification that involves merging, for each topic, the timelines of the two different news outlets. Our main contributions are as follows:

1. We develop a new TLS corpus - TLS-Covid19 - covering two languages (English and Portuguese) from two different trustworthy news sources per language (CNN and The Guardian for English, and Público and Observador for Portuguese).
2. We open room for researchers to explore language-independent summarization methods as 30 English topics (out of 35) can also be found as topics in the Portuguese variant;
3. TLS-Covid19 is made available to the research community through a Python script that enables to reconstruct the dataset and to keep collecting further news articles and ground-truth timelines;
4. Based on this dataset, we conduct an evaluation process and present experimental results by comparing two different baselines (random; oracle upper bounds) to understand the effectiveness of TLS methods under the proposed dataset.

The remainder of this paper is organized as follows. Section 2 offers an overview of the related work in timeline summarization. Section 3 presents the current available TLS datasets. Section 4 describes the construction of the TLS-Covid19 corpus. Section 5 introduces the experimental setup. Section 6 discusses the results obtained from our comparative experiments. Finally, Sect. 7 concludes this paper by summing up the most important contributions of our research and by pointing out possible future research directions.

2 Related Work on Timeline Summarization Systems (TLS)

Summarization is an active topic that has been discussed since the'50s [21]. According to McCreadie et al. [25] it can be framed within four categories: (1) Multi-Document Summarization; (2) Timeline Generation (aka timeline summarization); (3) Update Summarization; and (4) Temporal Summarization. Most researchers [9, 10, 17] focused on single and multi-document summarization (MDS) where extractive methodologies are usually employed by selecting the most relevant sentences to produce a new single document. More recently, timeline summarization (TLS) appears as a particular case of MDS aiming to summarize events across time and to put them in an automatically generated timeline. The general idea is to extract textual units from related batch documents over time through a retrospective perspective [2–5, 14, 23, 27–29]. In this case, the temporal dimension plays an important role, and documents are assumed to be time-tagged or to have at least some inherent (possibly ambiguous) temporal information in a way that texts can be anchored in a timeline. While automatically generated summaries have proved to be a valuable instrument to digest large volumes of textual data, they are hard to evaluate. The most popular, among the available evaluation methods, focus on comparative textual evaluation, where a summary produced by an automatic system is compared against one or more gold-standard summaries manually constructed by humans. Unlike MDS, which only needs to consider the compression rate between the input documents and the reference summaries, in TLS, one is required to find not only relevant information but also relevant dates to be placed in a timeline. Catizone et al. [12] formalizes this process as follows: 1) relevant documents should be included in the appropriate timeframe; 2) each timeline unit should contain accurate text labels, and 3) the timeline should include the most significant events of the document collection. Manually generating annotated summaries, however, is a laborious and time-consuming task. In the following section, we provide a discussion about the currently available datasets. Despite a few releases over the last few years, none, to the best of our knowledge, has considered making available a multi-lingual dataset across a number of topics, likely slowing down the emergence of novel methods in the context of timeline summarization. TLS-Covid19 dataset allows to fill this gap. Its description will be given in Sect. 4.

3 Shared Tasks and TLS Datasets

With the growing maturity and understanding of TLS task, the attention of researchers has progressively shifted to include formal and standard ways of evaluating their algorithms. In this section, we begin by describing two related shared tasks, before presenting five state-of-the-art TLS datasets.

3.1 Shared Tasks

The problem of evaluating timeline summarization systems is long-standing. Within this research area, there are two shared tasks, *TREC-TS* and *SemEval 2015 Task 4*, which are worth mentioning as an alternative to datasets dedicated to TLS.

TREC-TS: From 2013 to 2015 the Text Retrieval Conference (TREC) promoted the Temporal Summarization track (TREC-TS) to formalize the process of real-time temporal summarization [6–8]. This task is similar to update summarization, where a stream of documents is processed, and each sentence is evaluated in terms of its novelty and information gain. Relevant sentences are then selected to illustrate the event in summary. Although relevant, the task definition and assumptions at TREC-TS are not explicitly designed for TLS due to its streaming nature. The robustness of these datasets has also been discussed by McCreadie et al. [24].

SemEval 2015 Task 4: Another example of a related shared task is the SemEval 2015 Task 4 [26] which focusses on cross-document event coreference resolution and cross-document temporal relation extraction to identify temporal expressions. The challenge is to use a set of full-text documents as input to extract temporal relations related to a given target entity and to present a timeline with ordered events. Although related, this shared task differs from the usual timeline summarization as its purpose is to order events instead of sentences.

3.2 TLS Tasks

While several approaches have been proposed over the years, including the above-cited shared tasks, the lack of specifically annotated corpora has limited the evaluation of the initial attempts, thus demanding researchers to create their own evaluation datasets. In this section, we describe five state-of-the-art datasets (the *Timeline17*, the *crisis dataset*, the *social timeline*, the *Chen2019* dataset, and the *entities* dataset) which have been used in the process of evaluating TLS algorithms.

Timeline17: Tran et al. [28] proposed a method that links news articles with already existing timelines edited by journalists as reference summaries. The authors selected 17 of such timelines from 9 different topics published by six different news agencies, including CNN and BBC. Considering these topics as queries, they used Google search engine to retrieve the top 400 articles published in the same timespan as the original timeline. Their final dataset consists of 4,650 articles and was made publicly available[5] to the community.

Crisis Dataset: Tran et al. [29] follows Timeline17 with a similar methodology. Authors built a new and larger dataset focused on long-timespan stories on armed conflicts, such as the Egypt Revolution, Syria War, Yemen Crisis, and Libya War. The dataset comprises 15,534 news articles and 25 manually constructed timelines extracted from 24 news agencies, obtained from January 2011 to July 2013.

[5] https://l3s.de/~gtran/timeline/.

Social Timeline: Wang et al. [31] proposed the TIMELINE2014[6], which includes news articles and their respective user comments. Similar to other works, the authors crawled articles from news providers. The timeline dataset comprises 5,788 articles and 1,436,332 comments collected from the CNN, BBC, and the NYTimes on four topics, the missing Malaysia Airlines Flight MH370, the political crisis in Ukraine, the Israel-Gaza conflict and the NSA surveillance leaks. Authors provide six timelines as ground-truth based on respective Wikipedia entries for each topic.

Chen2019: Chen et al. [13] built a Chinese language dataset based on a Chinese encyclopedia[7] specially designed for abstractive timeline summarization. The dataset consists of timelines about celebrities from different countries. Each celebrity's entry in the encyclopedia contains a biographical timeline summary and a larger section detailing their experiences. In the experiences section, each event is a paragraph with an explanation and details, which is selected as an input article.

Entities: More recently, Ghalandari and Ifrim [16] have developed a dataset with 47 timelines extracted from CNN Fast[8], a CNN directory containing a large list of curated timeline articles. Authors selected mainly timeline articles about personalities as ground-truth. For each timeline, the authors defined a set of keyphrases as queries. They collected the input articles using The Guardian's API.

A summary of the datasets' statistics (including the proposed TLS-Covid19) is given in Table 1. Next, we describe the construction of our dataset.

Table 1. Available datasets for TLS.

Dataset	Language	Domain	Timespan	#Topics	#Docs	#Timelines
Timeline17	English	News	3 years	9	4, 650	17
Crisis	English	News	4 years	4	15,534	25
Social Timeline	English	News, Comments	1 year	4	5,788	6
Chen2019	Chinese	Biographies	Decades	NA	179,423	NA
Entities	English	News	Decades	47	~ = 45,075	47
TLS-Covid19	English, Portuguese	News	11 months	178	100,399	356

4 TLS-Covid19 Dataset

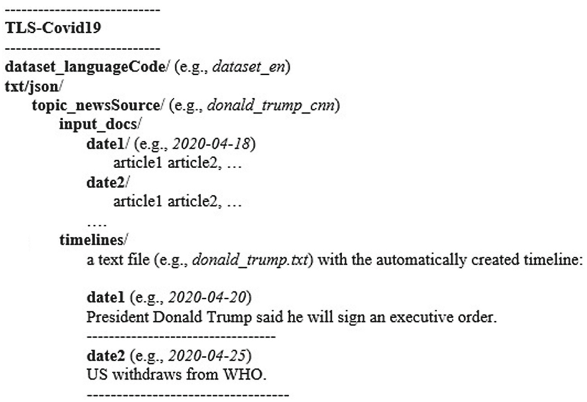
While several COVID-19 related datasets have been made available over the last few months [1], none to the best of our knowledge, is related to the timeline summarization

[6] https://web.eecs.umich.edu/~wangluxy/data.html.

[7] https://baike.baidu.com/.

[8] https://edition.cnn.com/specials/world/fast-facts.

task. In addition to this, existing datasets, as shown in the previous section, are mostly limited to a single language, thus hampering the evaluation of the proposed solutions across different scenarios. In this paper, we propose a dataset on a timely subject and relevant task that does not only address English, but also low resource languages such as Portuguese. Our future plans involve keeping collecting news articles and possibly expanding it for other languages as a means to improve its multi-lingual aspects. We invite the interested researchers on this task to join us in this effort. The current version of TLS-Covid19, consists of 178 topics (35 in English and 143 in Portuguese), their associated 100,399 news articles (31,891 in English and 68,508 in Portuguese) and timelines corresponding to the topics that cover the time period of January 2020 until December 2020. For each topic there is a number of related news articles and the corresponding ground-truth timeline. Both the news articles, as well as the timelines, are provided in two different formats (json and txt) and structured to be easily read by the tilse[9] timeline evaluation framework proposed by Martschat and Markert [23]. Figure 1 shows the format, the structure and the organization of the dataset. Details about its construction and corresponding statistics will be given in the next sections.



```
----------------------------
TLS-Covid19
----------------------------
dataset_languageCode/ (e.g., dataset_en)
txt/json/
    topic_newsSource/ (e.g., donald_trump_cnn)
        input_docs/
            date1/ (e.g., 2020-04-18)
                article1 article2, …
            date2/
                article1 article2, …
            ….
        timelines/
            a text file (e.g., donald_trump.txt) with the automatically created timeline:

            date1 (e.g., 2020-04-20)
            President Donald Trump said he will sign an executive order.
            ---------------------------------
            date2 (e.g., 2020-04-25)
            US withdraws from WHO.
            ----------------------------------
```

Fig. 1. Organization and structure of the dataset.

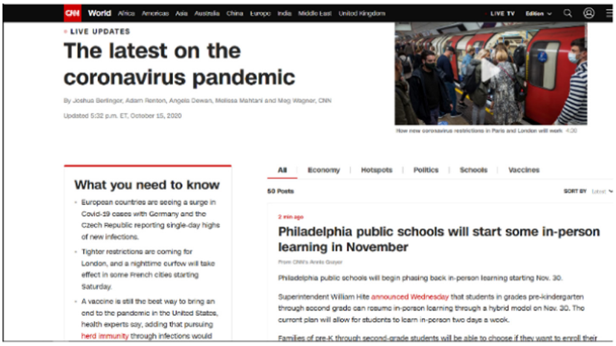
4.1 Data Collection (Input Documents and Ground-Truth)

To build this dataset, we considered two credible news sources for each language, CNN and The Guardian as the English news sources, Público and Observador as the Portuguese ones. All of them provide an everyday live coverage of the COVID-19 outbreak. The referred live coverage is provided by what is commonly known as liveblogs (CNN[10],

[9] https://github.com/smartschat/tilse.

[10] https://edition.cnn.com/world/live-news/coronavirus-pandemic-vaccine-updates-12-31-20/index.html.

The Guardian[11], Público[12] and Observador[13]), a webpage (which usually has a different URL everyday) where media outlets provide news about an ongoing event, typically in the form of frequent short updates and links to news articles. In addition to the published news articles, liveblogs contain a section that highlights in a sentence-based manner the most important events during the day. These highlights are defined by journalists, thus guaranteeing their quality and credibility, and form our ground-truth timeline for that particular date. Figure 2 depicts an example of the CNN liveblog. In the figure one can observe the highlights in the left box named "What we need to know". Articles are shown on the right-hand side.



LIVE UPDATES

The latest on the coronavirus pandemic

By Joshua Berlinger, Adam Renton, Angela Dewan, Melissa Mahtani and Meg Wagner, CNN

Updated 5:32 p.m. ET, October 15, 2020

What you need to know

- European countries are seeing a surge in Covid-19 cases with Germany and the Czech Republic reporting single-day highs of new infections.
- Tighter restrictions are coming for London, and a nighttime curfew will take effect in some French cities starting Saturday.
- A vaccine is still the best way to bring an end to the pandemic in the United States, health experts say, adding that pursuing herd immunity through infections would

2 min ago

Philadelphia public schools will start some in-person learning in November

From CNN's Annis Guyer

Philadelphia public schools will begin phasing back in-person learning starting Nov. 30.

Superintendent William Hite announced Wednesday that students in grades pre-kindergarten through second grade can resume in-person learning through a hybrid model on Nov. 30. The current plan will allow for students to learn in-person two days a week.

Families of pre-K through second-grade students will be able to choose if they want to enroll their

Fig. 2. Liveblog of CNN (snapshot taken at 15/10/2020).

As a rule-of-thumb, we consider the beginning of the liveblog coverage as the start time period of collecting the articles, and December 31st, 2020 as the end time period. For instance, CNN is tracked since January 22nd, 2020; The Guardian since January 24th, 2020; Público since March 16th, 2020; and Observador since January 30th, 2020. The acquisition of the data is entirely automatic. Instructions on how to collect this data are available on a public repository[14] under which a Python script that enables the reconstruction of the dataset is provided along with all the statistics and documentation about the dataset. Our aim is to continue expanding the dataset with further articles and possibly new topics until the end of the outbreak and/or the end of the liveblogs' coverage. We anticipate that as the pandemic evolves, the amount of data collected will grow significantly.

4.2 Selecting Candidate Topics

Next step in this process is to select a list of relevant topics. Instead of conducting a topic analysis which does not fit the purposes of our study, we consider selecting topics as

[11] https://www.theguardian.com/world/live/2020/dec/30/coronavirus-live-news-uk-approves-oxford-astrazeneca-vaccine-updates.

[12] https://www.publico.pt/2020/12/31/sociedade/noticia/covid19-portugal-1944703.

[13] https://observador.pt/liveblogs/passagem-de-ano-com-restricoes-arranca-com-proibicao-de-circulacao-entre-concelhos.

[14] https://github.com/LIAAD/tls-covid19.

named entities (persons, organizations and locations), as broad concepts tend to be often used by ordinary users accessing timeline summarization systems [27]. To accomplish this objective, we apply the well-known spaCy's NLP framework [19]. Further to this, we consider selecting relevant keyphrases from our collection of highlighted texts and news articles as popular keywords are often issued by users when interacting with search engines. To this regard, we apply YAKE! [11] keyphrase extraction tool which has shown to be effective in capturing relevant keywords (e.g., "vaccine", "easter", "coronavirus", etc.).

As the first preliminary step, we begin by selecting candidate topics within the highlighted data. Our assumption is that topics appearing within text editorially defined by journalists as daily representative are likely to be relevant topics. Next, we conduct a search and match process to find the occurrences of each candidate topic in the news articles, thus collecting the corresponding input documents. Afterwards, we remove all topics from the dataset that have low temporal coverage or that appear too often. To this regard, we set the following criteria:

1. To remove candidate topics with low temporal coverage, a candidate topic must be present, similarly to Ghalandari and Ifrim [16], in at least 5 highlighted events, in both news sources;
2. To ignore candidates that appear too often (thus moving away from the summarization task), the number of occurrences for a candidate topic in the highlights should not exceed 50% of its number of occurrences in the news articles, in both news sources.

Finally, we manually curated the list of topics to consider, merging overlapping topics (e.g., "donald trump" with "trump"), and removing noise data and typos. Figure 3 shows the word cloud of the topics for both languages. The larger the font size of the text, the higher the topic frequency. As can be observed, most of the topics, regardless the language, are related to the pandemic situation in countries/locations ("*France*", "*China*", "*Italy*"), but other entities such as persons ("*Boris Johnson*") and organizations ("*Johns Hopkins University*") can also be found. Overall, we have 143 PT topics (*PER*:

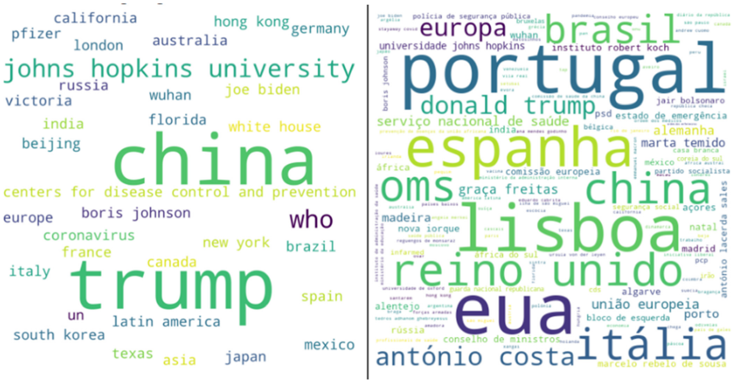


Fig. 3. English liveblog topics (left-hand side) and Portuguese liveblog topics (right-hand side).

Table 2. Overall statistics of the corpus with averages by language.

		Input Docs				Ground-Truth			Compression	
Lang	#Topics	#docs	Avg #sents	Avg #dates	Avg sents/dates	Avg #sents	Avg #dates	Avg sents/dates	Sent	Date
EN	35	31,891	3648.70	135.20	26.99	27.69	21.47	1.29	0.76	15.89
PT	143	68,508	1372.69	110.23	12.45	88.86	48.91	1.82	6.47	44.37

17; *ORG*: 33; *LOC*: 82; *Keyphrases*: 11) and 35 EN topics (*PER*: 3; *ORG*: 6; *LOC*: 25; *Keyphrases*: 1) thus representing a number of diverse topics related to the COVID-19 situation. It is also important to note that, the majority of the topics (30 out of 35) in the English dataset are represented in the Portuguese one too, thus opening room for multi-language timeline summarization research.

4.3 Dataset Statistics

Table 2 displays the main statistics of the corpus. In the table, we can observe information related to the input documents (collected news articles), ground-truth (timelines) and the compression rate, that is, the ratio between the number of sentences (or dates) in the input documents and the sentences (or dates) in the ground-truth. One can also observe that the compression rate for sentences in the English dataset is just 0.76%. Such compression rate indicates how difficult it may be to achieve high effectiveness. The lower the value, the higher the difficulty (Table. 3).

Table 3. Overall statistics by news source.

		Input Docs				Ground-Truth		
Source	#Topics	#docs	Avg #sents	Avg #dates	Avg sents/dates	Avg #sents	Avg #dates	Avg sents/dates
CNN	35	26,043	6178.54	189.71	32.57	30.11	20.97	1.44
The Guardian	35	5,848	1118.86	80.69	13.87	25.26	21.97	1.15
Público	143	28,327	1092.15	99.93	10.93	62.82	40.05	1.57
Observador	143	40,181	1653.22	120.52	13.72	114.90	57.77	1.99

5 Experimental Setup

To provide a demonstration of the validity of the proposed dataset, we conduct a set of experiments on available methods for the TLS task. The experiments conducted here serve as a guiding example. It is out of the scope of this work to make comparative experiments on top of different datasets. Although the immediate use of the dataset is tuned for unsupervised approaches, its future use is not limited to this particular setting as researchers may easily adapt it to their own needs.

5.1 Evaluation Metrics

To conduct the evaluation, we apply the tilse framework [23], a reference evaluation framework specifically designed to evaluate timeline summarization methods. In this research, we make use of the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) extension metric provided by Martschat and Markert [22] to evaluate the effectiveness of the different state-of-the-art methods. Rouge extension is particularly suited to evaluate n-grams overlaps by also taking into account the temporal information embedded in the timelines. In this work, we report the F1 scores of ROUGE-1 and ROUGE2 for the concatenation, agreement, date alignment, and date selection metrics that can be found in the tilse evaluation framework. ROUGE-1 stands for the overlap of unigrams between the automatically generated timeline and the ground-truth reference timeline, and ROUGE-2 refers to the overlap of bi-grams between the generated timeline and the ground-truth timeline summary. Naturally, dates in both the generated timeline and the ground-truth timeline may consist of one or more sentences depending solely on the number of topic references found throughout the day. Overlaps of n-grams are naturally measured within the available summary, be it a single sentence or multiple sentences. In the following, we briefly introduce each of the evaluation metrics considered in our experiments.

Concatenation: In this metric, temporal information is not considered, that is, we only look at the overlap (unigram or bigram) between the generated timeline textual summary and the corresponding ground-truth.

Agreement: In this metric, both textual, as well as temporal overlap, are taken into account. This means that, while the textual overlap between the generated timeline and the ground-truth is important, it only matters if their dates match. Otherwise, a score of 0 is assigned.

Date Selection: Finally, we consider date selection to assess how well the model behaves in exactly selecting the same dates (regardless of the textual content) between the generated timeline and the reference timelines.

5.2 Methods

In this section, we present the experimental results for the baselines *random* and *Oracle Upper Bounds*. All baselines are available in the evaluation framework tilse framework [23]. A succinct description of each one of them is presented below.

Random: is a naive baseline model that selects sentences randomly. Its results represent the worst-case scenario for a TLS constraints model.

Oracle Upper Bound (TLS Oracle): aims to calculate the best possible ROUGE scores under the input documents and the available ground-truth [18]. Such a baseline aims to estimate the best-case scenario and the level which extractive summarization algorithms can reach.

6 Results and Discussion

The results obtained from our comparative experiments are displayed in Tables 4 and 5 averaged over all generated timelines for all topics from each language in the corpus. Table 4 begins by showcasing the scores for date selection. The random baseline shows the lower bound scores that are acceptable for this task while the TLS Oracle shows the best possible results considering an extractive summarization approach. One can observe that selecting dates that match exactly with the ground-truth is easier for the Portuguese dataset because it contains a higher date coverage in the ground-truth. It is also visible how difficult it is to select the right content for the right date once we compare the ROUGE scores for the simple concatenation metric against the ROUGE in the date agreement. The difference between these two baselines represents the room for improvement that researchers can focus on. The reported results also show that scores decrease to a great extent when applying Rouge-2, thus indicating the difficulty of this task. One can conclude that, regardless of the case, there is still a long way to reach the upper bounds established by the Oracle baseline, thus opening room for further improvements within the research community. More extensive results with additional baselines are available at https://github.com/LIAAD/tls-covid19.

Table 4. Date selection scores.

	English dataset			Portuguese dataset		
Methods	Precision	Recall	F1	Precision	Recall	F1
Random	0.252	0.252	0.252	0.484	0.484	0.484
TLS Oracle	0.968	0.968	0.968	0.999	0.999	0.999

Table 5. Content selection scores using ROUGE.

			Rouge 1			Rouge 2		
Lang	Method	Metric	Prec	Recall	F1	Prec	Recall	F1
English	Random	Concat	0.183	0.190	0.187	0.022	0.023	0.023
		Agreement	0.018	0.020	0.019	0.003	0.004	0.004
	TLS Oracle	Concat	0.423	0.531	0.471	0.185	0.216	0.199
		Agreement	0.347	0.438	0.388	0.177	0.211	0.192
Portuguese	Random	Concat	0.281	0.466	0.351	0.065	0.106	0.080
		Agreement	0.059	0.097	0.073	0.013	0.023	0.017
	TLS Oracle	Concat	0.373	0.675	0.480	0.168	0.304	0.216
		Agreement	0.280	0.517	0.363	0.139	0.265	0.183

7 Conclusions

In this paper, we present the TLS-Covid19 dataset, an important resource for the TLS task. Compared to existing datasets, we provide a larger number of topics and multilingual resources on a timely subject. TLS-Covid19 consists of 178 COVID-19 related topics, 100,399 news articles and 356 reference timelines extracted from 4 news sources.

Our plan is to keep expanding this dataset until COVID-19 pandemics is over. To foster reproducibility, we provide scripts for that. To test the validity of our dataset, we performed baseline evaluations using tilse framework, a specially designed framework for TLS evaluation. The experimental results show that there is still room for improvements in this area. We believe that by providing a new dataset in this domain, we will contribute to promote the "development" of new algorithms.

Acknowledgements. The first five authors of this paper were financed by the ERDF – European Regional Development Fund through the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project PTDC/CCI-COM/31857/2017 (NORTE-01–0145-FEDER-03185). This funding fits under the research line of the Text2Story project. The first author of this paper was employed by Signal Media Ltda. When part of this work was developed. The last author was employed by Kyoto University when the first version of this paper was completed.

References

1. Alam, F., et al.: Fighting the COVID-19 infodemic: modeling the perspective of journalists, fact-checkers, social media platforms, policy makers, and the society. arXiv preprint arXiv:2005.00033 (2020)
2. Allan, J., Gupta, R., Khandelwal, V.: Temporal Summaries of New topics. SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New Orleans, Louisiana, USA. September 9 – 13, pp. 1018. ACM (2001)
3. Alonso, O., Baeza-Yates, R., Gertz, M.: Exploratory search using timelines. In: ESCHI 2007: Proceedings of the Workshop on Exploratory Search and Computer Human Interaction associated to CHI2007: SIGCHI Conference on Human Factors in Computing Systems. San Jose, CA, USA. April 29, pp. 2326. ACM (2007)
4. Alonso, O., Berberich, K., Bedathur, S., Weikum, G.: Time-based exploration of News archives. In: Proceedings of the fourth Workshop on Human-Computer Interaction and Information Retrieval (HCIR), New Brunswick, USA, pp. 12–15 (2010)
5. Ansah, J., Liu, L., Kang, W., Kwashie, S., Li, J., Li, J.: A Graph is worth a thousand words: telling event stories using timeline summarization graphs. In: Proceedings of the World Wide Web Conference (WWW 2019). San Francisco, USA. May 13 – 17, pp. 25652571. ACM (2019)
6. Aslam, J., Diaz, F., Ekstrand-Abueg, M., McCreadie, R., Pavlu, V., Sakai, T.: TREC 2014 Temporal Summarization Track Overview. In: Proceedings of the Twenty-Third Text Retrieval Conference (TREC 2014). Gaithersburg, USA, MIT Press (2015)
7. Aslam, J., Diaz, F., Ekstrand-Abueg, M., McCreadie, R., Pavlu, V., Sakai, T.: TREC 2015 Temporal Summarization TrackOverview. In: Proceedings of the Twenty-fourth Text REtrieval Conference (TREC 2014). Gaithersburg, USA. November 17 - 20: MIT Press (2016)
8. Aslam, J., Diaz, F., Ekstrand-Abueg, M., Pavlu, V., Sakai, T.: TREC 2013 Temporal Summarization. In: Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013). Gaithersburg, USA. November 19 - 22: MIT Press (2014)
9. Barzilay, R., Elhadad, N., McKeown, K.R.: Inferring strategies for sentence ordering in multidocument News summarization. J. Artif. Intell. Res. **17**(1), 35–55 (2002)

10. Berger, A., Mittal, V.O.: Query-relevant Summarization using FAQs. In: Proceedings of the 38th annual meeting on association for computational linguistics (ACL 2000), Hong Kong, China. October 03 – 06, pp. 294–301 (2000)
11. Campos, R., Mangaravite, V., Pasquali, A., Jatowt, A., Jorge, A., Nunes, C.: YAKE! keyword extraction from single documents using multiple local features. Inf. Sci. J. **509**, 257–289 (2020)
12. Catizone, R., Dalli, A., Wilks, Y.: Evaluating automatically generated timelines from the web. In: LREC 2006: Proceedings of the 5th International Conference on Language Resources and Evaluation. Genoa, Italy. May 24 - 26: ELDA, pp. 885888 (2006)
13. Chen, X., Chan, Z., Gao, S., Yu, M.-H., Zhao, D., Yan, R.: Learning towards Abstractive Timeline Summarization. Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19), pp. 4939–4945 (2019)
14. Chieu, H.L., Lee, Y.K.: Query based event extraction along a timeline. In: Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval (SIGIR2004), Sheffield, UK. July 25–29, pp. 425–432. ACM (2004)
15. Esteva, A., et al.: Co-search: Covid-19 information retrieval with semantic search, question answering, and abstractive summarization. arXiv preprint arXiv:2006.09595 (2020)
16. Ghalandari, D.G., Ifrim, G.: Examining the state-of-the-art in News timeline summarization. arXiv preprint arXiv:2005.10107 (2020)
17. Goldstein, J., Mittal, V., Carbonell, J., Kantrowitz, M.: Multi-document Summarization by Sentence Extraction. In: Proceedings of the Workshop on Automatic summarization (ANLP@NAACL2000), Seattle, Washington. April 30, pp. 40–48 (2000)
18. Hirao, T., Nishino, M., Suziki, J., Nagata, M.: Enumeration of extractive oracle summaries. arXiv preprint arXiv:1701.01614 (2017)
19. Honnibal, M., Montani, I.: spaCy 2: natural language understanding with bloom embeddings. Convolutional Neural Netw. Incremental Parsing **7**(1) (2017)
20. Lin, H., Bilmes, J.: Multi-document summarization via budget maximization of submodular functions. In: Proceedings of Human Language Technologies 2010: The Conference of the North American Chapter of the Association for Computational Linguistc, Los Angeles, pp. 912–920 (2010)
21. Luhn, H.P.: The automatic creation of literature abstracts. IBM J. Res. Dev. **2**(2), 159–165 (1958)
22. Martschat, S., Markert, K.: Improving {ROUGE} for timeline summarization. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain. April 3–7, pp. 285–290 (2017)
23. Martschat, S., Markert, K.: A temporally sensitive submodularity framework for timeline summarization. In: Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018). Brussels, Belgium. October 31 - November 1: Association for Computational Linguistic, p. 230 (2018)
24. McCreadie, R., Rajput, S., Soboroff, I., Macdonald, C., Ounis, I.: On enhancing the robustness of time-line summarization test collections. Inf. Process. Manage. **56**(5), 18151836 (2019)
25. McCreadie, R., Santos, R.L.T., Macdonald, C., Ounis, I.: Explicit diversification of event aspects for temporal summarization. ACM Trans. Inf. Syst. **36**(3), 1–31 (2018). https://doi.org/10.1145/3158671
26. Minard, A.-L., et al.: SemEval-2015 Task 4: Timeline: cross-document event ordering. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval2015). Denver, USA, June 4–5: Association for Computational Linguistic, pp. 778–786 (2015)
27. Pasquali, A., Mangaravite, V., Campos, R., Jorge, A.M., Jatowt, A.: Interactive system for automatically generating temporal narratives. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11438, pp. 251–255. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15719-7_34

28. Tran, G.B., Alrifai, M., Nguyen, D.Q.: Predicting relevant news events for timeline summaries. In: WWW2013 Proceedings of the Companion Publication of the 22nd International Conference on World Wide Web Companion, Rio de Janeiro, Brazil. May 13 – 17, pp. 91–92 (2013)
29. Tran, G., Alrifai, M., Herder, E.: Timeline summarization from relevant headlines. In: Hanbury, A., Kazai, G., Rauber, A., Fuhr, N. (eds.) ECIR 2015. LNCS, vol. 9022, pp. 245–256. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16354-3_26
30. Voorhees, E., et al.: TREC-COVID: constructing a pandemic information retrieval test collection. ArXiv abs/2005.04474 (2020)
31. Wang, L., Cardie, C., Marchetti, G.: Socially-informed timeline generation for complex events. In: Proceedings of the Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL. Denver, Colorado. May 31-June 5: Association for Computational Linguistic, p. 1055 (2015)
32. Wang, L., et al.: CORD-19: The Covid-19 open research dataset. arXiv:2004.10706v4 (2020)
33. Yan, R., Wan, X., Otterbacher, J., Kong, L., Li, X., Zhang, Y.: Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In: Proceedings of the 34th International Conference on Research and Development in Information Retrieval (SIGIR 2011). Beijing, China. July 24–28, pp. 745–754. ACM (2011)
34. Yang, W., et al.: On the generation of medical dialogues for COVID19. arXiv:2005.05442v2 (2020)

A Multi-task Approach to Neural Multi-label Hierarchical Patent Classification Using Transformers

Subhash Chandra Pujari[1,2](✉), Annemarie Friedrich[1], and Jannik Strötgen[1]

[1] Bosch Center for Artificial Intelligence, Renningen, Germany
{subhashchandra.pujari,annemarie.friedrich,jannik.stroetgen}@de.bosch.com
[2] Institute of Computer Science, Heidelberg University, Heidelberg, Germany

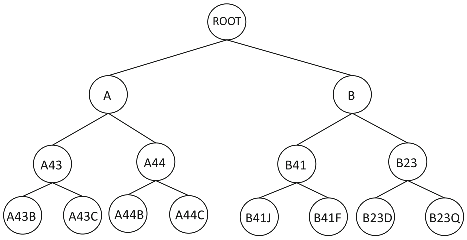
Abstract. With the aim of facilitating internal processes as well as search applications, patent offices categorize documents into taxonomies such as the Cooperative Patent Categorization. This task corresponds to a multi-label hierarchical text classification problem. Recent approaches based on pre-trained neural language models have shown promising performance by focusing on leaf-level label prediction. Prior works using intrinsically hierarchical algorithms, which learn a separate classifier for each node in the hierarchy, have also demonstrated their effectiveness despite being based on symbolic feature inventories. However, training one transformer-based classifier per node is computationally infeasible due to memory constraints. In this work, we propose a *Transformer-based Multi-task Model* (TMM) overcoming this limitation. Using a multi-task setup and sharing a single underlying language model, we train one classifier per node. To the best of our knowledge, our work constitutes the first approach to patent classification combining transformers and hierarchical algorithms. We outperform several non-neural and neural baselines on the WIPO-alpha dataset as well as on a new dataset of 70k patents, which we publish along with this work. Our analysis reveals that our approach achieves much higher recall while keeping precision high. Strong increases on macro-average scores demonstrate that our model also performs much better for infrequent labels. An extended version of the model with additional connections reflecting the label taxonomy results in a further increase of recall especially at the lower levels of the hierarchy.

Keywords: Patent classification · Hierarchical classification · Multi-label classification · Neural modeling · Multi-task learning

1 Introduction

A patent is a legal text document describing an invention and granting its owner exclusive rights for monetary exploitation thereof. Upon submission of a patent application, patent offices assign one or several labels categorizing the described

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 513–528, 2021.
https://doi.org/10.1007/978-3-030-72113-8_34



Level	ID	Description
Section	A	Human Necessities
Class	A43	Footwear
Subclass	A43B	Characteristic features of footwear; parts of footwear

Fig. 1. Excerpt of the hierarchical **Cooperative Patent Classification** (CPC) scheme.

invention according to a taxonomy such as the Cooperative Patent Classification (CPC). This scheme, developed jointly by the US Patent and Trademark Office (USPTO) and the European Patent Office, organizes types of inventions in a hierarchical tree structure as illustrated in Fig. 1. CPC information is used internally by the patent offices, e.g., for routing the application to the respective experts. It is also released publicly with each patent in order to facilitate search-related tasks including the retrieval and filtering of patents.

From a machine learning (ML) point of view, assigning CPC codes to patents constitutes a hierarchical multi-label text classification problem and has high relevance to a variety of information-retrieval (IR) related real-life tasks. First, with currently almost 2,000 patent applications being submitted per day to USPTO alone, the automatic prediction of CPC codes helps to speed up manual work considerably. Second, patent language often intentionally conceals the type of invention by avoiding terminology commonly used in technical reports [28]. Detecting the underlying CPC codes present in patents, scientific reports or other types of text-based queries will lead to more meaningful rankings of patents during retrieval. Finally, the CPC taxonomy itself is under constant development with new categories being added or parts being restructured. Accurate automatic classification methods will help to keep patent databases up-to-date with the taxonomy, a prerequisite for the above mentioned search applications.

At its top level, the CPC scheme has nine *sections*. *Subclasses* are further divided into *main groups* and *subgroups*, amounting to a total of 250,000 categories. Patent classification has been addressed by the IR and ML communities in the context of several shared tasks organized by ALTA and CLEF-IP [27,30], operating at various granularities of the taxonomy. In this paper, following previous work [21,22], we address the first three levels of the taxonomy, resulting in hierarchical multi-label classification tasks with huge label inventories of around 600 classes in our datasets (Sect. 4).

We address this large-scale classification task using a novel combination of a pre-trained language model [3,8] and a *local* hierarchical learning algorithm. Such algorithms train one "local" classifier per node of the taxonomy predicting whether an instance belongs to the respective category or not, and have been shown to be highly effective for hierarchical patent classification in previous work using symbolic features such as n-grams and part-of-speech tags [4,5]. Similarly, approaches based on contextual word embeddings and transformers have shown promising performance [12,21,22]. They apply a *flat* strategy, i.e., they train a single classifier that simply predicts leaf-level classes and infer ancestor classes from them. In this paper, we combine the advantages of using powerful document embeddings generated by a pre-trained language model with the gains that can be achieved by localizing decisions. It is arguably computationally infeasible in most infrastructures to instantiate hundreds of transformer-based language models in parallel. Therefore, we propose a new multi-task based neural architecture for hierarchical multi-label classification in which the individual classifiers corresponding to the nodes of the taxonomy constitute the classification heads in a neural network, sharing the same underlying transformer-based language model. In addition, we create a variant adding connections between the classification heads that are related in the label taxonomy.

We benchmark our approach with a variety of non-neural and neural hierarchical text classification algorithms using the WIPO-alpha dataset and a new patent dataset spanning the years 2006–2019. We publish the latter along with our paper. On both datasets, our models strongly outperform prior work both in terms of macro- and micro-averages. Our detailed analysis of performance at the different levels of the taxonomy reveals that our models are much better (a) at predicting less frequent categories and (b) at predicting finer-grained labels. Adding taxonomy-based connections to our model results in further increases in recall especially for leaf-level labels.

Our contributions are as follows. (i) We propose a novel *Transformer-based Multi-task Model* neural-network architecture and a variant adding hierarchical connections (Sect. 3). We open-source our implementation in order to foster future research. (ii) We sample a new dataset of 70k recent USPTO patents that we make publicly available for benchmarking (Sect. 4).[1] (iii) We perform an in-depth analysis demonstrating that our models strongly outperform prior work, achieving much better accuracy on the lower levels of the hierarchy as well as for less frequent CPC classes (Sect. 5).

2 Related Work

Despite having been studied in the data mining, ML, and IR communities for many years [2], text classification remains a very active research field addressing a variety of domains [15,23,37]. Since the seminal works using Convolutional Neural Networks (CNNs) for sentence classification [16,18], neural modeling has become the predominant approach. In this work, we focus on **hierarchical text classification** [34], in which the label set constitutes a hierarchy. While some architectures or algorithms directly reflect these taxonomies [4,5], others apply *flat* or *global* approaches either predicting only leaf-level labels or simply treating all labels independently [14,21,22].

We here address the task of **patent classification**, which while constituting a hierarchical multi-label text classification problem, is often addressed

[1] https://github.com/boschresearch/hierarchical_patent_classification_ecir2021.

using flat classifiers [11], though with several notable exceptions [4,5,36]. Patent documents are usually represented by transforming the text of their title and abstract into a feature vector for classification. Recent work uses the CPC scheme explained above, while some older datasets use the International Patent Categorization (IPC), which is roughly speaking a predecessor of CPC. In a recent shared task on patent classification [27], an approach training separate SVM classifiers per node using simple n-gram and POS-tag based features [5] performed comparably to a flat neural approach [12] based on the ULMFiT contextual language model [13]. The work of Li et al. [22], based on [18] and optimized by [1], proposes a convolutional neural network based on non-contextual word2vec [26] embeddings predicting IPC codes on subclass level. In this work, we compare to the state-of-the-art HARNN system [14] (see Sect. 5.3). HARNN generates document embeddings with a BiLSTM initialized using word2vec, feeds these through a hierarchical attention-based memory unit that learns different attention weights per category, and finally predicts categories by combining hidden *local* and *global* information. The former relates to level-wise predictions and the latter consists of predictions for the entire taxonomy. Further, neural work on patent classification [35] employs graph-convolutional networks using word embeddings inferred from a word-document co-occurrence graph. Hierarchical patent classification has also been addressed as a sequence generation problem using an attention-based neural network model [32].

Outside the context of patent classification, [40] uses a very similar approach to [20,35] and [38] address **neural hierarchical text classification** by training level-wise classifiers and chaining predictions top-down. Similar to our work, [29] propose a CNN model in which the hierarchy of labels is leveraged by regularizing the deep architecture with dependencies among labels. A weakly-supervised hierarchical classification approach is suggested in [24]. Given a few user-provided seeds, the system generates pseudo-documents that are used for bootstrapping a neural hierarchical classifier including an LSTM-based language model.

Recently, **transformer**-based **neural language models** such as BERT [8] have been shown to be highly effective for a variety of natural language processing tasks [33], following a "pre-train and fine-tune" approach. In the context of patent classification, PatentBERT [21] adds a single hidden layer on top of BERT, mapping the CLS embedding to a sigmoid output in order to predict CPC labels on subclass level. [17] employs the same idea, predicting relevance with regard to a pre-specified topic.

Our work differs from previous work in the area of hierarchical patent classification in the following aspects. First, instead of predicting labels at a single hierarchical level [1,5,21,22], we model predictions across the label taxonomy. Second, unlike the flat classification [21,22] model architectures, we make use of the taxonomy by transferring information from the parent label to child labels. Finally, to the best of our knowledge, our approach is the first to combine powerful transformer-based language models with an intrinsically hierarchical algorithm for patent classification.

3 Model Architecture

3.1 Overview

We propose a neural hierarchical classification architecture as illustrated in Fig. 2. We assume the label set $L = \{l_1, l_2, l_3, ..., l_n\}$ in which labels are arranged hierarchically. The task consists in assigning a subset of L to each input document. For each predicted label, the respective ancestors should also be contained in the output set.

We create distributed representations of the textual input using a pre-trained trans-former-based neural language model. For each label in the label set, we train a binary classifier that decides whether an instance belongs to the respective category or not. The ensemble of classifiers is trained in a multi-task setup and makes use of a single underlying SciBERT neural language model [3] for creating document representations. SciBERT has been trained on a corpus of scientific publications and is hence closer to the patent domain than the standard BERT model [8]. In the terminology of multi-task learning, each of these classification heads addresses one *task*. Hence, each label-specific binary classifier constitutes a classification head in our multi-task based neural network architecture. In other words, our **Transformer-based Multi-task Model (TMM)** consists of a single transformer model with n heads where n corresponds to the number of labels in the hierarchy. Parameters of the transformer model (and of optional CNN layers) are shared in a hard way. In addition, each classification head has its own set of parameters. Further, to analyze the impact of sharing information between hierarchically related tasks, we propose an extended architecture which adds links between the network components corresponding to nodes that are linked in the hierarchy. We call this latter model **Transformer-based Hierarchical Multi-task Model (THMM)**.

3.2 Transformer-Based Language Model Based Document Representation

Similar to prior work on neural patent classification [14,21,22], we use the patent's title and abstract as input to our model. We concatenate them, word-piece tokenize the text and prepend the special CLS token. We leverage the transformer's output embeddings for the two variants of our model in the following ways: (i) We use the embedding generated for the CLS token (left-hand side path in Fig. 2), which can be regarded as capturing the semantics of the entire input text sequence [8]. However, the CLS token has been designed for next sentence prediction and it is unclear how effective its embedding is for representing long sequences as in our case. Hence, we also test a second option that explicitly considers the entire sequence: (ii) We compute a document embedding from the embeddings generated for each word-piece token by feeding them into a CNN (right/dotted path in Fig. 2). For details on the latter, see Sect. 5.2.

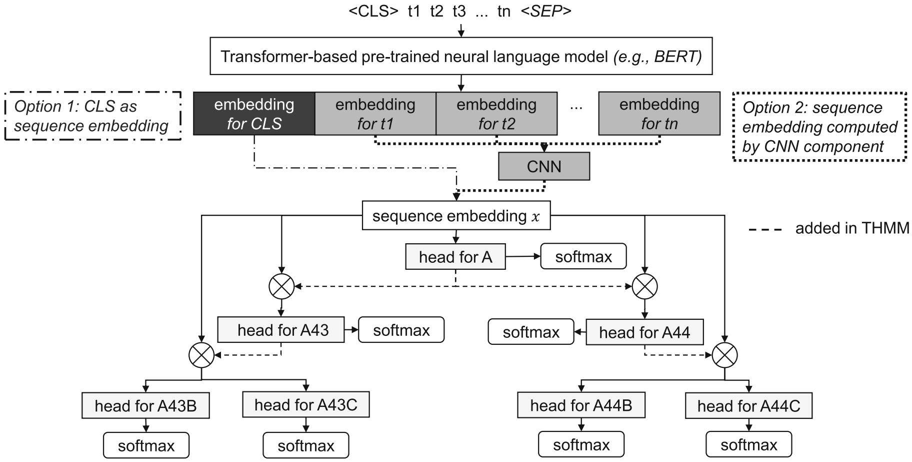


Fig. 2. Architecture of our **THMM** model for our running example from Fig. 1. The **TMM** version is the same but without the dashed connections between classification heads. Each classification head consists of three dense layers and predicts whether an instance belongs to the respective category or not.

3.3 Classification Heads

We next detail the architecture of the classification heads. For the **Transformer-based Multi-task Model (TMM)**, we create an independent head for each label. The input for each head is the document embedding x corresponding to the embedding of the CLS token or the CNN's output. Each head consists of two dense layers, both with ReLU activation, followed by a two-dimensional dense output layer producing logits. Finally, we perform classification by means of a softmax operation.

In the **Transformer-based Hierarchical Multi-task Model (THMM)**, we add connections between the classification heads as specified by the label taxonomy. As in the TMM, each classification head computes the logits for the binary decision using two fully connected dense layers. However, in this case (see Eq. (1)), the first hidden layer of the classification head for l_i additionally takes into account $h^2_{l_j}$, an output from the second (intermediate) dense layer of the head corresponding to l_i's parent l_j. It computes a hidden representation $h^1_{l_i}$ by performing a linear transformation on the concatenation ($\oplus$) of the sequence embedding x and $h^2_{l_j}$. If l_i does not have a parent in the taxonomy, the input to its classification head is simply x. The $parent(l_i, l_j)$ relation evaluates to $true$ if l_j is the parent of l_i, and to $false$ otherwise. ϕ is the ReLU activation function.

$$h^1_{l_i} = \begin{cases} \phi(W^1_{l_i}(h^2_{l_j} \oplus x) + b^1_{l_i}) & \text{if there is a } l_j \text{ with } parent(l_i, l_j) = true \\ \phi(W^1_{l_i} x + b^1_{l_i}) & \text{if } parent(l_i, ROOT) \end{cases} \tag{1}$$

$$h^2_{l_i} = \phi(W^2_{l_i}(h^1_{l_i}) + b^2_{l_i}) \qquad h^3_{l_i} = \phi(W^3_{l_i}(h^2_{l_i}) + b^3_{l_i}) \tag{2}$$

As in TMM, the hidden representation $h^1_{l_i}$ is passed through two further dense layers (Eq. (2)) and mapped to a two-dimensional logit vector $h^3_{l_i}$. This serves as input to a softmax layer that performs the prediction whether label l_i applies to the instance. For training our models, we use binary cross entropy loss and weight all "tasks" equally.

4 Patent Classification Datasets

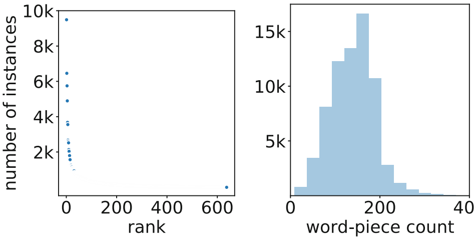
In this section, we give details on the two datasets we use for our experiments. To ensure comparability with prior work, we use WIPO-alpha, which contains 75k patents from 1998–2001 annotated with the IPC scheme. As domains, writing style and terminology of patents evolve over time, we also experiment with more recent data. We create a new dataset of 70k USPTO patents spanning the years 2006–2019, using the more recent CPC scheme. We release this dataset to ensure reproducibility of our study.

4.1 USPTO Dataset

We sample a dataset of 70k patents from the USPTO patents data dump[2] as follows. With the aim of creating a realistic setup in which models predict labels for newer patents based on older data, similarly to [9], we split the dataset temporally, assigning the documents from years 2006–2017 to the training set, 2018 to dev and 2019 to test. Our training sample contains 50k patents, and the dev and test sets 10k each.

	splits	total labels			average labels per patent		
level		1	2	3	1	2	3
USPTO	train	9	128	630	1.49	1.69	1.98
	dev	9	126	575	1.56	1.84	2.25
	test	9	127	573	1.56	1.89	2.32
WIPO	train	8	123	602	1.22	1.34	1.49
	dev	8	120	544	1.22	1.35	1.49
	test	8	128	576	1.22	1.35	1.51

(a)



(b)

(c)

Fig. 3. Corpus statistics of USPTO and WIPO-alpha datasets. (a) Label counts by level. (b) Label count distribution for USPTO. (c) Instance length distribution for USPTO.

We address label sparsity by up-sampling the least frequent labels by adding patents carrying the infrequent label such that each label occurs at least 10

[2] https://www.patentsview.org/download.

times in the training set. Dev and test distributions are not changed. Figure 3b shows that for some labels, there are many instances, but the distribution has a long tail. As shown in Fig. 3a, the total number of labels at leaf node, i.e., subclass, level is 630 for train, 575 in dev, and 573 for test. There is one label occurring in dev that does not have any associated training instances. In the test split, there are 7 such labels. The average number of labels per patent is around 1.5 on the first level of the hierarchy and up to 2.32 on the leaf level, with the latter increasing from 1.8 in 2014 to 2.3 in 2019. This reflects a tendency towards more interdisciplinary inventions and further demonstrates the need to take the temporal dimension into account when training and evaluating models.

4.2 WIPO-alpha

The WIPO-alpha dataset[3] contains about 46k training instances and 29k test instances. The patent documents were published between 1998 and 2002, with test instances sampled randomly.[4] There are 602 labels in train and 576 test labels at subclass level. As there is no pre-existing split, we sample a validation (dev) set from train by selecting 20% of the data points at random. There are 22 labels with instances in test but without examples in the training data at subclass level. The IPC code in the dataset is defined using the seventh edition of IPC which labels each patent with a main IPC code and a set of secondary IPC codes. Unlike prior work [1], which considers only the main IPC code and benchmarks the models in a single-label flat classification setting, we consider all IPC codes in a hierarchical multi-label classification setting.

5 Experiments

In this section, we first describe our experimental setup including evaluation metrics, baselines and implementation details. We then discuss our experimental results in detail.

5.1 Evaluation Metrics

For evaluating our models, we use *hierarchical* precision, recall and F1-score as proposed by [19] and defined as $hP = \frac{\sum |P_i \cap T_i|}{\sum P_i}$ and $hR = \frac{\sum |P_i \cap T_i|}{\sum T_i}$. For each test instance i, the set P_i consists of all predicted labels and their respective ancestors. T_i contains all true labels including ancestors. For all results and analyses reported in this section, we modify the set of predicted labels to include relevant ancestors.

Prior work [14] has focused on evaluating per-instance (*micro*) scores. As the distribution of instances per label is highly skewed (see Fig. 3b), we additionally report *macro*-scores that average across scores obtained per label.

[3] https://www.wipo.int/classifications/ipc/en/ITsupport/Categorization/dataset/.

[4] See WIPO-alpha readme and personal correspondence with authors.

We compute macro-F1 as the average over the macro-F1 scores per label. Unless otherwise stated, we consider a model to predict a label if the softmax score for the dimension "label applies" is at least 0.5. In addition, in line with previous work [14,38], we evaluate the predictions as a ranking task, which does not require defining a threshold. We compute the Area Under the Precision-Recall Curve (AUPRC) [7] as implemented in scikit-learn.[5] In the case of models outputting only leaf-level scores, we here use the maximum of the leaf-level scores for each intermediate-level label.

5.2 Implementation and Hyperparameter Settings

We implement our models in Python using TensorFlow 2.0[6] and Keras [6]. We use the HuggingFace Transformers library [39] for integrating SciBERT [3]. For efficiency reasons, we truncate the word-piece tokenized input sequences to a maximum length of 256. As illustrated in Fig. 3c, this covers the complete input text for almost all instances in USPTO (and also for WIPO-alpha, not shown).

We found the following hyperparameters to work best across our two benchmark datasets for our proposed TMM/THMM models. All dense layers have a hidden size of 256 and use ReLU activations. For training, we apply a learning rate of 10^{-5}, a dropout of 0.25 across layers, and a batch size of 64. In the case of the CNN model variant, we compute a single-vector document representation using a CNN whose architecture largely follows [22]. For each word-piece token, we compute an embedding by summing up the corresponding weights of the last four SciBERT layers. Then, we concatenate the embeddings of all word-piece tokens and apply convolution operations with kernel sizes {2, 3, 4, 5}. In contrast to [22], we add an extra kernel of size 2 to capture bigrams and we use a filter size of 256 instead of 512. The training of a single model takes approximately 300 h on a Nvidia Tesla V100 GPU with 80 GB VRAM.

5.3 Baselines

We compare our models to a wide range of non-neural and neural models. First, the **TwistBytes** system [4] constitutes a competitive *non-neural* baseline. We run a recently updated version[7] leveraging a TF-IDF vector of uni-gram features. The system is implemented using scikit-learn[8] and learns one support vector classifier [31] per node. During prediction, the model only tests for presence of labels if the respective parent's score is positive. Finally, the set of predicted labels is filtered using a threshold of −0.25.

HARNN. In order to compare to a recent state-of-the-art *neural* model for hierarchical patent classification, we run the Hierarchical Attention-based Recurrent Neural Network [14] on our datasets. We keep hyperparameter settings as

[5] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision_score.html.

[6] https://www.tensorflow.org.

[7] https://dublin.zhaw.ch/~benf/HPC.

[8] https://github.com/globality-corp/sklearn-hierarchical-classification.

proposed, representing each document using a 100-dimensional Word2Vec [25] model trained on train and dev, using 256 and 512 as the hidden sizes in the BiLSTM and for each fully connected layer, respectively. Local and global information are combined with a regulation parameter α with a value of 0.5. For a fair comparison with the other models, we tune the prediction threshold for macro-performance, resulting in 0.15 for both datasets. **HARNN-orig** [14] uses a prediction threshold of 0.5.

flat-*. In addition, we provide results for simplified versions of our own model, predicting only labels for the leaf level and inferring ancestors during post-processing. First, **flat-CNN** corresponds to DeepPatent [22], which uses a CNN with kernels of sizes $\{3, 4, 5\}$ and 512 filters on top of SciBERT. The outputs of all CNN layers are flattened and concatenated, resulting in a 1,536-dimensional document embedding. Second, **flat-CLS** is based on PatentBERT [21], using SciBERT's 786-dimensional CLS embedding directly as document embedding. The feature vectors of **flat-CNN** and **flat-CLS** are subsequently fed into a multi-layer perceptron with two dense layers, applying sigmoid activation to each logit. For both models, dense layers have size 512, the learning rate is set to 10^{-5}, dropout rate is 0.25 and batch size is 64.

5.4 Experimental Results

We now analyze the performance of our models and compare them to prior work. We find similar tendencies on the two datasets and show that our models perform better especially at deeper levels of the hierarchy and for less frequent labels.

Classification Performance. Table 1 and Table 2 show results obtained for the USPTO and WIPO-alpha datasets, respectively. Based on these, we can draw

Table 1. Hierarchical classification results on **USPTO** test set.

Model	macro-avg.			micro-avg.			AUPRC
	hP	hR	hF1	hP	hR	**hF1**	
TwistBytes [4]	0.423	0.203	0.257	0.651	0.534	0.587	0.407
HARNN-orig [14]	0.355	0.126	0.170	**0.781**	0.481	0.595	0.661
HARNN [14]	0.292	0.281	0.267	0.519	0.679	0.588	0.661
flat-CNN [22]	**0.486**	0.272	0.330	0.718	0.552	0.624	0.645
TMM-CNN	0.412	0.360	0.366	0.639	**0.636**	0.637	0.667
THMM-CNN	0.412	0.364	0.369	0.649	0.634	0.641	0.669
flat-CLS [21]	0.481	0.256	0.316	0.740	0.546	0.628	0.644
TMM-CLS	0.485	0.313	0.362	0.709	0.611	**0.656**	**0.678**
THMM-CLS	0.426	**0.367**	**0.377**	0.666	0.633	0.649	0.670

Table 2. Hierarchical classification results on **WIPO-alpha** test set.

Model	macro-avg.			micro-avg.			AUPRC
	hP	hR	hF1	hP	hR	hF1	
TwistBytes [4]	0.456	0.264	0.308	0.626	0.570	0.597	0.412
HARNN-orig [14]	0.089	0.021	0.027	**0.757**	0.248	0.373	0.505
HARNN [14]	0.206	0.269	0.206	0.373	0.652	0.474	0.505
flat-CNN [22]	0.466	0.348	0.382	0.707	0.578	0.636	0.641
TMM-CNN	0.408	0.400	0.389	0.636	0.684	0.659	0.681
THMM-CNN	0.377	0.413	0.380	0.620	0.686	0.651	0.674
flat-CLS [21]	**0.503**	0.328	0.377	0.737	0.598	0.660	0.674
TMM-CLS	0.462	0.376	0.399	0.682	0.679	**0.680**	**0.697**
THMM-CLS	0.409	**0.424**	**0.405**	0.651	**0.698**	0.674	0.690

the following conclusions. First, neural models generally perform better than the non-neural TwistBytes system, with SciBERT-based models outperforming HARNN. Our models achieve much higher recall while keeping precision high. When tuning HARNN for hF1 as in the original work, a high micro-hP can be achieved but at the cost of lower recall especially in the macro evaluation.[9] This implies that the original model focuses on the easy cases of highly frequent labels. Tuning HARNN for macro-scores changes the precision-recall tradeoff in the micro-setting and improves macro-F1, but still not approaching the performance of transformer-based models.

With the exception of macro-hP of flat-CNN on USPTO, the CLS-based models all outperform their CNN-based counterparts. However, the CLS-based models achieve the best results in terms of micro- and macro-F1 on both datasets. We conclude that there is no extra need for aggregating information across the sequence using a CNN layer. In most cases, adding hierarchical links between classification heads in TMM increases recall at the expense of precision. When comparing THMM-CLS with TMM-CLS on both datasets, the former does better in terms of macro-F1, while the latter has slightly higher micro-F1, i.e., adding the links helps especially for less frequent labels.

Finally, the flat strategy leads to good precision but is not competitive in terms of recall, illustrating that such models struggle with activating all relevant classifications to the required extent. The AUPRC scores also indicate that the TMM-CLS model performs best overall in terms of producing correct rankings of all labels for each patent, closely followed by THMM-CLS. Hence, our experiments confirm that when optimizing for a good trade-off between micro- and macro-average performance, hierarchical multi-label classification for patents is best approached using a fully hierarchical model.

[9] We double-checked the surprisingly low macro-scores of HARNN-orig and decided to present results of HARNN tuned for macro-performance as well.

Table 3. Analysis of coverage for USPTO dataset. *No Prediction*: number of test instances with no predicted labels at a given level. *False Positives (error analysis)*: average # hops between false positives and nearest true labels at third level.

Level	Avg. labels predicted			No prediction			False positives	
	1	2	3	1	2	3	# inst.	hops
Gold	1.56	1.89	2.32	0	0	0	0.0	0.0
TwistBytes [4]	1.65	1.51	1.56	147	891	1,575	3,788	4.17±1.77
HARNN-orig [14]	1.36	1.17	1.02	116	1,134	2,466	2,341	4.08±1.79
HARNN [14]	2.29	2.62	2.82	0	12	148	6,380	4.12±1.79
flat-CNN [22]	1.31	1.45	1.67	512	512	512	4,198	4.38±1.69
TMM-CNN	1.75	1.98	2.01	**1**	**42**	228	5,236	4.22±1.69
THMM-CNN	1.68	1.92	2.04	5	55	232	5,282	4.17±1.69
flat-CLS [21]	1.26	1.39	1.61	570	570	570	3,916	4.28±1.72
TMM-CLS	1.59	1.68	1.71	13	125	476	4,114	4.19±1.72
THMM-CLS	1.61	1.84	2.03	8	66	**204**	5,046	4.22±1.68

Performance Across Levels. Figure 4 shows an increase in macro-F1 for TMM and THMM compared to the baselines, resulting primarily from higher recall (not shown). Adding hierarchical links (THMM vs. TMM) results in better predictions mainly at level 3. Hence, the overall increase in F1 is a result of improved classification at the lower levels, and finer-grained labels benefit from passing on hierarchical information.

Coverage. The number of labels at the subclass (leaf) level varies strongly across instances from a single category to 20 or more, with a tendency of more recent patents having more labels. Hence, one difficulty of the task consists in outputting the right number of categories per instance [10]. Table 3 breaks down the average number of labels predicted by level of the hierarchy for USPTO (WIPO-alpha shows similar tendencies). At the top level of the hierarchy, all other models predict a roughly fitting number of labels. However, at levels 2 and 3, TwistBytes and the flat models predict markedly fewer labels. This effect is alleviated by the TMM and THMM models. While HARNN-orig strongly under-predicts the number of labels, our version of HARNN optimized for macro-F1 over-predicts, indicating that tuning the model either way is problematic. Next, we report the number of test instances for which a model did not make any prediction at a particular level ("No Prediction" in Table 3). This count is much lower for the TMM and THMM models, showing that the hierarchical models often can make predictions at intermediate levels even if the fine-grained class is unclear.

Error Analysis. Finally, we capture the models' mis-classification behavior by computing the number of hops in the label taxonomy to the nearest gold label

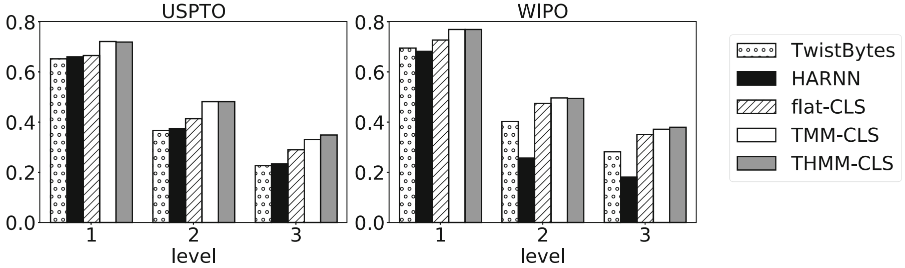


Fig. 4. Classification performance: **macro-avg. F1 by level of hierarchy**.

on the same level. For example, if a model incorrectly predicts B41F and A43C with the true label being A43B, the wrong predictions are 6 hops and 2 hops away from the true label, respectively. Table 3 shows the average number of hops between false positives and gold labels on level 3. The column titled **# inst.** denotes the number of test instances having at least one false positive label. In general, the wrong predictions of all models seem to be similarly far from the nearest gold label, usually within the correct section of the taxonomy. Again, the flat approach more often activates completely wrong labels.

Summary. Our experiments on two patent datasets have shown that our models based on pre-trained transformers strongly outperform both neural and non-neural prior work in terms of micro- and macro-scores. Recall increased considerably while keeping precision high. The coverage of our models is much better than the one of prior work; wrongly activated predictions usually are within the correct section of the taxonomy.

6 Conclusion and Outlook

In this work, we have proposed a novel Transformer-based Multi-task Model (TMM) for hierarchical patent classification. The strength of our architecture stems from integrating the highly effective local-classifier-per-node idea from traditional hierarchical classification algorithms with a large-scale pre-trained neural transformer language model, which is made computationally feasible by our novel multi-task based architecture. We have shown that this model architecture strongly outperforms previous work on hierarchical text classification, with a higher coverage of instances and addressing the long tail of less frequent labels more successfully.

Future Work. Further improvements for patent classification can be expected from integrating additional textual information, e.g., the description or claims sections, for computing the document embedding. In this work, we have focused on patents. Yet, our model should be easily adaptable to other genres and domains, e.g., by substituting the pre-trained language model with in-domain data. Improving the confidence estimation for classification decisions further may lead to more precise label activation while keeping recall high. Finally, as our model has the very practical application of patent categorization, improving the model in an active learning set-up may be a very promising direction.

Acknowledgements. We thank Mark Giereth and Jona Ruthardt for the fruitful discussions. We are grateful to Patrick Fievet for his support with the WIPO-alpha dataset. We also thank Alexander Müller for sharing his ideas on patent classification, and Lukas Lange, Trung-Kien Tran and the anonymous reviewers for their comments on this paper.

References

1. Abdelgawad, L., Kluegl, P., Genc, E., Falkner, S., Hutter, F.: Optimizing neural networks for patent classification. In: Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., Robardet, C. (eds.) ECML PKDD 2019. LNCS (LNAI), vol. 11908, pp. 688–703. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-46133-1_41
2. Aggarwal, C.C., Zhai, C.: Mining Text Data, chap. A Survey of Text Classification Algorithms, pp. 163–222. Springer (2012). https://doi.org/10.1007/978-1-4614-3223-4_6
3. Beltagy, I., Lo, K., Cohan, A.: SciBERT: a pretrained language model for scientific text. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, pp. 3615–3620. Association for Computational Linguistics, November 2019
4. Benites, F.: TwistBytes - hierarchical Classification at GermEval 2019: walking the fine line (of recall and precision). In: Proceedings of the 15th Conference on Natural Language Processing (KONVENS). Erlangen, Germany, October 2019
5. Benites, F., Malmasi, S., Zampieri, M.: Classifying patent applications with ensemble methods. In: Proceedings of the Australasian Language Technology Association Workshop 2018, pp. 89–92, Dunedin, New Zealand (2018)
6. Chollet, F., et al.: Keras (2015). https://github.com/fchollet/keras
7. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240. ICML 2006, Association for Computing Machinery, New York, NY, USA (2006)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (2019)
9. D'hondt, E., Verberne, S., Oostdijk, N., Beney, J., Koster, C., Boves, L.: Dealing with temporal variation in patent categorization: Inf. Retrieval **17**, 520–544 (2014)
10. Fall, C.J., Törcsvári, A., Benzineb, K., Karetka, G.: Automated categorization in the international patent classification. SIGIR Forum **37**(1), 10–25 (2003)

11. Gomez, J.C., Moens, M.-F.: A survey of automated hierarchical classification of patents. In: Paltoglou, G., Loizides, F., Hansen, P. (eds.) Professional Search in the Modern World. LNCS, vol. 8830, pp. 215–249. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12511-4_11
12. Hepburn, J.: Universal language model fine-tuning for patent classification. In: Proceedings of the Australasian Language Technology Association Workshop 2018, pp. 93–96, Dunedin, New Zealand (2018)
13. Howard, J., Ruder, S.: Universal Language Model Fine-tuning for Text Classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 328–339. Association for Computational Linguistics, Melbourne, Australia (2018)
14. Huang, W., et al.: Hierarchical multi-label text classification: an attention-based recurrent network approach. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1051–1060 (2019)
15. Jalan, R., Gupta, M., Varma, V.: Medical forum question classification using deep learning. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) ECIR 2018. LNCS, vol. 10772, pp. 45–58. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76941-7_4
16. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014), Baltimore, MD, USA, pp. 655–665 (2014)
17. Kang, D.M., Lee, C.C., Lee, S., Lee, W.: Patent prior art search using deep learning language model. In: Proceedings of the 24th Symposium on International Database Engineering & Applications. IDEAS 2020, Association for Computing Machinery, New York, NY, USA (2020)
18. Kim, Y.: Convolutional Neural Networks for Sentence Classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751. Association for Computational Linguistics, Doha, Qatar (2014)
19. Kiritchenko, S., Matwin, S., Famili, A.F.: Functional annotation of genes using hierarchical text categorization. In: Proceedings of BioLINK SIG: Linking Literature, Information and Knowledge for Biology (2005)
20. Kowsari, K., Brown, D.E., Heidarysafa, M., Meimandi, K.J., Gerber, M.S., Barnes, L.E.: Hdltex: hierarchical deep learning for text classification. In: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 364–371. IEEE (2017)
21. Lee, J.S., Hsiang, J.: PatentBERT: patent classification with fine-tuning a pretrained BERT model. World Patent Inf. **61**, 101965 (2020)
22. Li, S., Hu, J., Cui, Y., Hu, J.: DeepPatent: patent classification with convolutional neural networks and word embedding. Scientometrics **117**(2), 721–744 (2018)
23. Lu, Z., Du, P., Nie, J.-Y.: VGCN-BERT: augmenting BERT with graph embedding for text classification. In: Jose, J.M., et al. (eds.) ECIR 2020. LNCS, vol. 12035, pp. 369–382. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45439-5_25
24. Meng, Y., Shen, J., Zhang, C., Han, J.: Weakly-supervised hierarchical text classification. Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, 6826–6833 (2019)
25. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Bengio, Y., LeCun, Y. (eds.) 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, 2–4 May 2013, Workshop Track Proceedings (2013)

26. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
27. Mollá, D., Seneviratne, D.: Overview of the 2018 ALTA shared task: classifying patent applications. In: Proceedings of the Australasian Language Technology Association Workshop 2018, Dunedin, New Zealand, pp. 84–88. (2018)
28. Nanba, H., Kamaya, H., Takezawa, T., Okumura, M., Shinmori, A., Tanigawa, H.: Automatic translation of scholarly terms into patent terms. In: Proceedings of the 2nd International Workshop on Patent Information Retrieval, pp. 21–24. PaIR 2009, Association for Computing Machinery, New York, NY, USA (2009)
29. Peng, H., et al.: Large-scale hierarchical text classification with recursively regularized deep graph-CNN. In: Proceedings of the 2018 World Wide Web Conference, pp. 1063–1072 (2018)
30. Piroi, F., Hanbury, A.: Multilingual patent text retrieval evaluation: CLEF–IP. Information Retrieval Evaluation in a Changing World. TIRS, vol. 41, pp. 365–387. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-22948-1_15
31. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in Large Margin Classifiers, pp. 61–74. MIT Press (1999)
32. Risch, J., Garda, S., Krestel, R.: Hierarchical document classification as a sequence generation task. In: Proceedings of the Joint Conference on Digital Libraries (JCDL), pp. 147–155 (2020)
33. Rogers, A., Kovaleva, O., Rumshisky, A.: A primer in bertology: What we know about how bert works. arXiv preprint arXiv:2002.12327 (2020)
34. Silla, C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. Data Mining Knowl. Discov. **22**, 31–72 (2010)
35. Tang, P., Jiang, M., Xia, B.N., Pitera, J.W., Welser, J., Chawla, N.V.: Multi-label patent categorization with non-local attention-based graph convolutional network. In: Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2020) (2020)
36. Tikk, D., Biro, G.: Experiment with a hierarchical text categorization method on the wipo-alpha patent collection. In: Fourth International Symposium on Uncertainty Modeling and Analysis (ISUMA 2003), pp. 104–109 (2003)
37. Wang, P., Fan, Y., Niu, S., Yang, Z., Zhang, Y., Guo, J.: Hierarchical matching network for crime classification. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019), pp. 325–334. ACM (2019)
38. Wehrmann, J., Cerri, R., Barros, R.: Hierarchical multi-label classification networks. In: Proceedings of the 35th International Conference on Machine Learning (ICML 2018), pp. 5075–5084 (2018)
39. Wolf, T., et al.: Huggingface's transformers: State-of-the-art natural language processing. ArXiv abs/1910.03771 (2019)
40. Yao, L., Mao, C., Luo, Y.: Graph Convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2019), pp. 7370–7377 (2019)

Weakly-Supervised Open-Retrieval Conversational Question Answering

Chen Qu[1](✉), Liu Yang[1], Cen Chen[2], W. Bruce Croft[1], Kalpesh Krishna[1],
and Mohit Iyyer[1]

[1] University of Massachusetts Amherst, Amherst, USA
{chenqu,lyang,croft,kalpesh,m.iyyer}@cs.umass.edu
[2] Ant Financial Services Group, Hangzhou, China
chencen.cc@antfin.com

Abstract. Recent studies on Question Answering (QA) and Conversational QA (ConvQA) emphasize the role of retrieval: a system first retrieves evidence from a large collection and then extracts answers. This open-retrieval ConvQA setting typically assumes that each question is answerable by a single span of text within a particular passage (a span answer). The supervision signal is thus derived from whether or not the system can recover an exact match of this ground-truth answer span from the retrieved passages. This method is referred to as *span-match weak supervision*. However, information-seeking conversations are challenging for this span-match method since long answers, especially freeform answers, are not necessarily strict spans of any passage. Therefore, we introduce a *learned weak supervision* approach that can identify a paraphrased span of the known answer in a passage. Our experiments on QuAC and CoQA datasets show that the span-match weak supervisor can only handle conversations with span answers, and has less satisfactory results for freeform answers generated by people. Our method is more flexible as it can handle both span answers and freeform answers. Moreover, our method can be more powerful when combined with the span-match method which shows it is complementary to the span-match method. We also conduct in-depth analyses to show more insights on open-retrieval ConvQA under a weak supervision setting.

Keywords: Weak supervision · Open-retrieval · Conversational question answering

1 Introduction

Conversational search and Conversational Question Answering (ConvQA) have become one of the focuses of information retrieval research. Previous studies [5, 36] set up the ConvQA problem as to extract an answer for the conversation so far from a *given gold passage*. Recent work [30] has emphasized the fundamental role of retrieval by presenting an Open-Retrieval ConvQA (ORConvQA) setting.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 529–543, 2021.
https://doi.org/10.1007/978-3-030-72113-8_35

This setting requires the system to *learn* to retrieve top relevant passages from a large collection and then extract answers from the passages.

The open-retrieval setting presents challenges to training the QA/ConvQA system. Qu et al. [30] adopts a fully-supervised setting, which encourages the model to find the gold passage and extract an answer from it by manually including the gold passage in the retrieval results during training. This *full supervision* setting can be impractical since gold passages may not always be available. In contrast, other studies [2,8,23] assume no access to gold passages and identify weak answers in the retrieval results by finding a span that is an exact match to the known answer. We argue that the effectiveness of this *span-match weak supervision* approach is contingent on having only *span answers* that are short, or extractive spans of a retrieved passage. In information-seeking conversations, however, answers can be relatively long and are not necessarily strict spans of any passage. These *freeform answers* can be challenging to handle for span-match weak supervision.

In this work, we introduce a *learned weak supervision* approach that can identify a paraphrased span of the known answer in a retrieved passage as the weak answer. Our method is more flexible than span-match weak supervision since that it can handle both span answers and freeform answers. Moreover, our method is less demanding on the retriever since it can discover weak answers even when the retriever fails to retrieve any passage that contains an exact match of the known answer. By using a weakly-supervised training approach, our ConvQA system can discover answers in passages beyond the gold ones and thus can potentially leverage various knowledge sources. In other words, our learned weak supervision approach makes it possible for an ORConvQA system to be trained on natural conversations that can have long and freeform answers. The choice of the passage collection is no longer a part of the task definition. We can potentially combine different knowledge sources with these conversations since the weak answers can be discovered automatically.

Our learned weak supervisor is based on Transformers [41]. Due to the lack of training data to learn this module, we propose a novel training method for the learned weak supervisor by leveraging a diverse paraphraser [19] to generate the training data. Once the learned weak supervisor is trained, it is frozen and used to facilitate the training of the ORConvQA model.

We conduct experiments with the QuAC [5] and CoQA [36] datasets in an open-retrieval setting. We show that although a span-match weak supervisor can handle conversations with span answers, it is not sufficient for those with freeform answers. For more natural conversations with freeform answers, we demonstrate that our learned weak supervisor can outperform the span-match one, proving the capability of our method in dealing with freeform answers. Moreover, by combining the span-match supervisor and our method, the system has a significant improvement over using any one of the methods alone, indicating these two methods complement each other. Finally, we perform in-depth quantitative and qualitative analyses to provide more insight into weakly-supervised ORConvQA. Our data and model implementations will be available for research purposes.[1]

The rest of our paper is organized as follows. In Sect. 2, we present related work regarding question answering and conversational question answering. In Sect. 3, we formulate the research question of ORConvQA following previous work and present our weakly-supervised solution. In Sect. 4, we present our evaluation results on both span answers and freeform answers. Finally, Sect. 5 presents the conclusion and future work.

2 Related Work

Our work is closely related to question answering, conversational question answering, session search [26,27,56], and weak supervision and data augmentation [3,24]. We highlight the related works on QA and ConvQA as follows.

Question Answering. Most of the previous work formulates question answering either as an answer selection task [13,43,54] or a machine comprehension (MC) task [20,34,35,39]. These settings overlook the fundamental role of retrieval as articulated in the QA task of the TREC-8 Question Answering Track [42]. Another line of research on open-domain question answering addresses this issue by leveraging multiple documents or even the entire collection to answer a question [7,10,11,16,28]. When a large collection is given as a knowledge source, previous work [2,53] typically uses TF-IDF or BM25 to retrieve a small set of candidate documents before applying a neural reader to extract answers. More recently, neural models are being leveraged to construct learnable rerankers [14,18,22,44] or learnable retrievers [8,17,23] to enhance the retrieval performance. Compared to this work on single-turn QA, we focus on a conversational setting as a further step towards conversational search.

Conversational Question Answering. As an extension of the answer selection and MC tasks in single-turn QA, most research on conversational QA focuses on conversational response ranking [25,38,47–52] and conversational MC [4,5,15,29,31,32,36,55,57]. A recent paper [30] extends conversational QA to an open-retrieval setting, where the system is required to learn to retrieve top relevant passages from a large collection before extracting answers from the passages. Although this research features a learnable retriever to emphasize the role of retrieval in ConvQA, it adopts a fully-supervised setting. This setting requires the model to have access to gold passages during training, and thus is less practical in real-world scenarios. Instead, we propose a learned weakly-supervised training approach that can identify good answers in any retrieved documents. In contrast to the span-match weak supervision [2,8,23] used in single-turn QA, our approach is more flexible since it can handle freeform answers that are not necessarily a part of any passage.

[1] https://github.com/prdwb/ws-orconvqa.

3 Weakly-Supervised ORConvQA

In this section, we first formally define the task of open-retrieval ConvQA under a weak supervision setting. We then describe an existing ORConvQA model [30] and explain how we train it with our learned weak supervision approach.

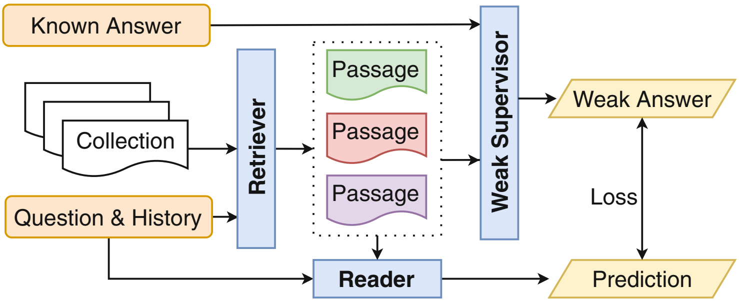


Fig. 1. Architecture of our full model. Given a question and its conversation history, the retriever first retrieves top-K relevant passages from the collection. The reader then reads the top passages and produces an answer. We adopt a weakly-supervised training approach. Given the known answer and one of the retrieved passages, the weak supervisor predicts a span in this passage as the weak answer to provide weak supervision signals for training the reader.

3.1 Task Definition

We define the ORConvQA task following Qu et al. [30]. Given the k-th question q_k in a conversation, and all history questions $\{q_i\}_{i=1}^{k-1}$ preceding q_k, the task is to predict an answer a_k for q_k using a passage collection C. Different from Qu et al. [30], we assume no access to gold passages when training the reader. The gold passage for q_k is the passage in C that is known to contain or support a_k.

3.2 An End-to-End ORConvQA System

We follow the same architecture of the ORConvQA model in Qu et al. [30].[2] Our approach differs from theirs in how we train the model. They use full supervision while we adopt weak supervision. We briefly describe the architecture of this ORConvQA model before introducing our weakly-supervised training approach.

As illustrated in Fig. 1, the ORConvQA model is composed of a passage retriever and a passage reader that are both learnable and based on Transformers [41]. Given a question and its history, the retriever first retrieves top-K relevant passages from the collection. The reader then reads the top passages and produces an answer. History modeling is enabled in both components by

[2] We disable the reranker in Qu et al. [30] since our preliminary experiments indicated the weak supervision signals seem to lead to degradation for reranker and retriever.

concatenating history questions. Since we do not have access to ground-truth history answers and gold passages, advanced history modeling approaches proposed in previous research [31,32] does not apply here. The training contains two phases, a pretraining phase for the retriever, and a concurrent learning phase for the reader and fine-tuning the question encoder in the retriever. Our weakly-supervised training approach is applied to the concurrent learning phase.

Retriever. The learnable retriever follows a dual-encoder architecture [1,8,23] that has a passage encoder and a question encoder. Both encoders are based on ALBERT [21] and can encode a question/passage into a 128-dimensional dense vector. The question is enhanced with history by prepending the initial question and other history questions within a history window. The retriever score is defined as the dot product of the representations of the question and the passage. The retriever pretraining process ensures the retriever has a reasonable initial performance during concurrent learning. A pretraining example contains a question and its gold passage. Other passages in the batch serve as sampled negatives. Using the passage encoder in the pretrained retriever, we encode the collection of passages to a collection of vectors. We then use Faiss[3] to create an index of these vectors for maximum inner product search [37] on GPU. The question encoder will be fine-tuned during concurrent learning using the retrieved passages. We refer our readers to Qu et al. [30] for further details.

Reader. The reader adapts a standard BERT-based extractive machine comprehension model [9] to a multi-document setting by using the shared-normalization mechanism [6] during training. First, the retrieved passages are encoded independently. Then, the reader maximizes the probabilities of the true start and end tokens among tokens from all the top passages. This step enables the reader to produce comparable token scores across all the retrieved passages for a question. The reader score is defined as the sum of the scores of the start token and the end token. The answer score is then the sum of its retriever score and reader score.

3.3 Weakly-Supervised Training

The reader component in Qu et al. [30] is trained with access to gold passages while our model is supervised by the conversation only. Our weakly-supervised training approach is *more practical* in real-world scenarios. Figure 1 illustrates the role the weak supervisor plays in the system. Given a known answer a_k and one of the retrieved passages p_j, the weak supervisor predicts a span in p_j as the *weak answer* a_k^{weak}. This weak answer is the weak supervision signal for training the reader. The weak supervisor can also indicate there is no weak answer contained in p_j. A question is skipped if there are no weak answers in any of the retrieved passages.

[3] https://github.com/facebookresearch/faiss.

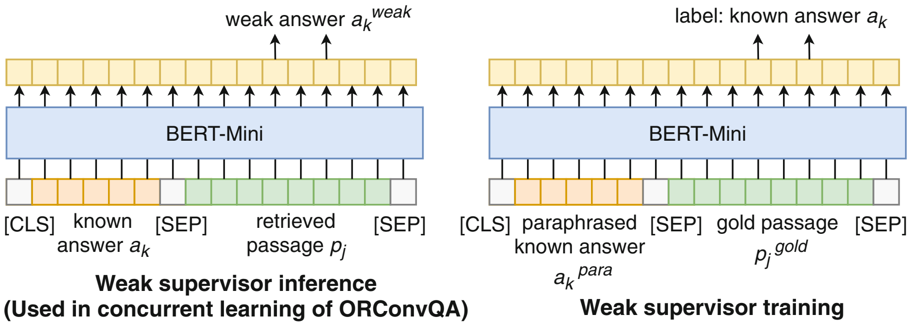

weak answer a_k^{weak}
BERT-Mini
[CLS]
known answer a_k
[SEP]
retrieved passage p_j
[SEP]
Weak supervisor inference (Used in concurrent learning of ORConvQA)
label: known answer a_k
BERT-Mini
[CLS]
paraphrased known answer a_k^{para}
[SEP]
gold passage p_j^{gold}
[SEP]
Weak supervisor training

Fig. 2. Learned weak supervisor. During the concurrent learning phase of ORConvQA, the weak supervisor conducts inference on a retrieved passage p_j (the left figure) to predict a passage span that is a paraphrase of the known answer a_k. When training of the weak supervisor (the right figure), the model is trained to predict the known answer a_k in the passage given a paraphrase of the known answer a_k^{para} and the passage.

Inspirations. Our learned weak supervision method is inspired by the classic span-match weak supervision. This method has been the default and only weak supervision method in previous open-domain QA research [2,8,23]. These works mainly focus on factoid QA, where answers are short. A span-match weak supervisor can provide accurate supervision signals since the weak answers are exactly the same as the known answers. In addition, the short answers can find matches easily in passages other than the gold ones. In information-seeking conversations, however, the answers can be long and freeform, and thus are more difficult to get an exact match in retrieved passages. Although the span-match weak supervisor can still provide accurate supervision signals in this scenario, it renders many training examples useless due to the failure to find exact matches. A straightforward solution is to find a span in a retrieved passage that has the maximum overlap with the known answer. Such overlap can be measured by word-level F1. This overlap method, however, can be intractable and inefficient since it has to enumerate all spans in the passage. This method also requires careful tuning for the threshold to output "no answer". Therefore, we introduce a learned weak supervisor based on Transformers [41] to predict a weak answer span directly in a retrieved passage given the known answer. This supervisor also has the ability to indicate that the retrieved passage does not have a good weak answer.

Learned Weak Supervisor. Given the known answer a_k and one of the retrieved passages p_j, the weak supervisor predicts a span in p_j as the weak answer a_k^{weak}. Intuitively, a_k^{weak} is a paraphrase of a_k. We use a standard BERT-based extractive MC model [9] here as shown in Fig. 2, except that we use a_k for the question segment. The best weak answer for all top passages is the one with the largest sum of start and end token scores.

Although theoretically simple, this model presents challenges in training because position labels of a_k^{weak} are not available. Therefore, we consider the known answer a_k as the weak answer we are seeking since we know the exact position of a_k in its gold passage p_j^{gold}. We then use a diverse paraphrase generation model (described in Sect. 3.3) to generate a paraphrase a_k^{para} for the known answer a_k. The paraphrase a_k^{para} simulates the known answer during the training of the weak supervisor, as shown in Fig. 2. The weak supervisor is trained before concurrent learning and kept frozen during concurrent learning. We train the weak supervisor to tell if the passage does not contain a weak answer by pairing a randomly sampled negative passage with the known answer.

We are aware of a dataset, CoQA [36], that provides both span answer and freeform answer for a given question q_k. In this case, we can take the freeform answer as a natural paraphrase a_k^{para} for the span answer (known answer) a_k when training the weak supervisor. For datasets that do not offer both answer types, our diverse paraphraser assumes the role of the oracle to generate the paraphrase answer. In other words, the use of the diverse paraphraser ensures that our weak supervision approach can be applied to a wide variety of conversation data that are beyond datasets like CoQA.

Diverse Paraphrase Model. We now briefly describe the diverse paraphraser [19] used in the training process of the learned weak supervisor. This model is built by fine-tuning GPT2-large [33] using encoder-free seq2seq modeling [46]. As training data we use PARANMT-50M [45], a massive corpus of back translated data [45]. The training corpus is aggressively filtered to leave sentence pairs with high lexical and syntactic diversity so that the model can generate diverse paraphrases. We refer our readers to Krishna et al. [19] for further details.

4 Experiments

We now describe the experimental setup and report the results of our evaluations.

4.1 Experimental Setup

Dataset. We select two ConvQA datasets, QuAC [5] and CoQA [36], with different answer types (span/freeform) to conduct a comprehensive evaluation of our weak supervision approach and to provide insights for weakly-supervised ORConvQA. We present the data statistics of both datasets in Table 1. We remove unanswerable questions in both datasets since there is no basis to find weak answers.[4]

OR-QuAC (span answers) We use the OR-QuAC dataset introduced in Qu et al. [30]. This dataset adapts QuAC to an open-retrieval setting. It contains information-seeking conversations from QuAC, and a collection of 11 million Wikipedia passages (document chunks).

[4] This difference in the data accounts for the discrepancies of the full-supervision results presented in Table 2.

OR-CoQA (freeform answers) We process the CoQA dataset [36] in the Wikipedia domain for the open-retrieval setting following Qu et al. [30], resulting in the OR-CoQA dataset. CoQA offers freeform answers generated by people in addition to span answers, resulting in more natural conversations. OR-CoQA and OR-QuAC share the same passage collection. Similar to QuAC, many initial questions in CoQA are also ambiguous and hard to interpret without the given gold passage (e.g., "When was the University established?"). OR-QuAC deals with this by replacing the *first question* of a conversation with its context-independent *rewrite* offered by the CANARD dataset [12] (e.g., "When was the University *of Chicago* established?"). This makes the conversations self-contained. Since we are not aware of any CANARD-like resources for CoQA, we prepend the document title to the first question for the same purpose (e.g., "*University of Chicago* When was the University established?"). Since the CoQA test set is not publicly available, we take the original development set as our test set and 100 dialogs from the original training set as our development set.

Table 1. Data statistics.

Items	OR-CoQA			OR-QuAC		
	Train	Dev	Test	Train	Dev	Test
# Dialogs	1,521	100	100	4,383	490	771
# Questions	23,027	1,494	1,611	25,824	2,808	4,406
# Avg. question tokens	5.8	5.7	5.8	6.8	6.6	6.8
# Avg. answer tokens	2.8	2.6	2.6	15.0	15.0	14.7
# Avg. dialog questions	15.1	14.9	16.1	5.9	5.7	5.7
# Avg./Max History turns per question	7.9/22	7.6/21	7.9/19	2.8/11	2.8/11	2.8/11

Competing Methods. Since this work focuses on weak supervision, we use the same ORConvQA model and vary the supervision methods. To be specific, the competing methods are:

- **Full supervision** (Full S): Manually add the gold passage to the retrieval results and use the ground-truth answer span [30]. This only applies to QuAC since we have no passage relevance for CoQA. This method serves as the upper bound of model performance and it is not comparable with other weak supervision methods that do not have access to the groundtruth answers in concurrent learning.
- **Span-match weak supervision** (Span-match WS): This method finds a weak answer span that is identical to the known answer in the retrieved passages. When there are multiple matched spans, we take the first one.
- **Learned weak supervision** (Learned WS): This is our method in Sect. 3.3 that finds a paraphrased span of the known answer as the weak answer.
- **Combined weak supervision** (Combined WS): This is the combination of the above two methods. We first use the span-match weak supervisor to try to

find a weak answer. If it fails, we take the weak answer found by the learned weak supervisor.

Evaluation Metrics. We use the word-level F1 and human equivalence score (HEQ) [5] to evaluate the performance of ConvQA. **F1** evaluates the overlap between the prediction and the ground-truth answer. **HEQ** is the percentage of examples for which system F1 $\geq$ human F1. This is computed on a question level (HEQ-Q) and a dialog level (HEQ-D).

In addition to the performance metrics described above, we define another set of metrics to reveal the impact of the weak supervisor in the training process as follows. **% Has answer** is the percentage of training examples that have a weak answer (in the last epoch). **% Hit Gold** is the percentage of training examples that have a weak answer identified in gold passages (in the last epoch). **Recall** is the percentage of training examples that have the gold passage retrieved (in the last epoch). **% From Gold** is the percentage of predicted answers that are extracted from the gold passages.

Implementation Details. Our models are based on the open-source implementation of ORConvQA[5], Diverse Paraphrase Model[6], and the HuggingFace Transformers repository.[7] We use the same pretrained retriever in Qu et al. [30] for both datasets. For concurrent learning of ORConvQA, we set the number of training epochs to 5 (larger than [30]) to account for the skipped steps where no weak answers are found. We set the number of passages to update the retriever to 100, and the history window size to 6 since these are the best settings reported in [30]. The max answer length is set to 40 for QuAC and 8 for CoQA. The rest of the hyper-parameters and implementation details for the ORConvQA model are the same as in [30].

For the weak supervisor, we use BERT-Mini [40] for better efficiency. We set the number of training epochs to 4, the learning rate to 1e–4, and the batch size to 16. As discussed in Sect. 3.3, the diverse paraphraser is used for OR-QuAC only. For OR-CoQA, we use the freeform answer provided by the dataset as a natural paraphrase to the span answer.

4.2 Evaluation Results on Span Answers

Given the different properties of span answers and freeform answers, we study the performance of our weak supervision approach on these answers separately. We report the evaluation results on the span answers in Table 2. Our observations can be summarized as follows.

The full supervision setting yields the best performance, as expected. This verifies the supervision signals provided by the gold passages and the ground-truth answer spans are more accurate than the weak ones. Besides, all supervision

[5] https://github.com/prdwb/orconvqa-release.

[6] https://github.com/martiansideofthemoon/style-transfer-paraphrase.

[7] https://github.com/huggingface/transformers.

Table 2. Evaluation results on OR-QuAC (span answers). The learned weak supervisor causes no statistical significant performance decrease compared span match.

Methods		Full S	Span-match WS	Learned WS	Combined WS
Train	% Has answer	100.00%	72.96%	75.98%	75.52%
Dev	F1	**22.8**	**20.8**	20.2	20.1
	HEQ-Q	**8.1**	**6.8**	6.0	6.4
	HEQ-D	0.6	0.6	0.2	0.6
Test	F1	**23.9**	**23.6**	23.1	23.2
	HEQ-Q	**14.0**	12.3	11.8	**12.5**
	HEQ-D	**2.2**	1.7	**1.9**	**1.9**

approaches have similar performance on span answers. This suggests that span-match weak supervision is sufficient to handle conversations with span answers. Ideally, if the known answer is part of the given passage, the learned weak supervisor should be able to predict the weak answer as exactly the same with the known answer. In other words, the learned weak supervisor should fall back to the span-match weak supervisor when handling span answers. In practice, this is not guaranteed due to the variance of neural models. However, our learned weak supervisor causes no statistical significant performance decrease compared with the span-match supervisor. This demonstrates that the learned weak supervision approach can cover span answers as well. Although we observe that the learned supervisor can identify more weak answers than span match, these weak answers could be false positives that do not contribute to the model performance. Finally, for the combined weak supervisor, our analysis shows that 96% of the weak answers are identified by span match, further explaining the fact that all weak supervision approaches have almost identical performance.

4.3 Evaluation Results on Freeform Answers

We then look at the evaluation results on freeform answers in Table 3. These are the cases where a span-match weak supervisor could fail. We observe that combining the learned weak supervisor with span match brings a statistically significant improvement over the span-match baseline on the test set, indicating these two methods complement each other. The test set has multiple reference answers per question, making the evaluation more practical. In addition, the learned supervisors can identify more weak answers than span match, these weak answers contribute to the better performance of our model. Further, for the combined weak supervisor, our analysis shows that 77% of the weak answers are identified by span match. This means that nearly a quarter of the weak answers are provided by the learned supervisor and used to improve the performance upon span match. This further validates the source of effectiveness of our model.

Table 3. Evaluation results on OR-CoQA (freeform answers). ‡ means statistically significant improvement over the span-match baseline with $p < 0.05$.

Methods		Span-match WS	Learned WS	Combined WS
Train	% Has answer	51.81%	65.75%	70.35%
Dev	F1	18.3	18.9	**19.7**
	HEQ-Q	11.6	9.0	**12.7**
	HEQ-D	0.0	0.0	0.0
Test	F1	24.3	26.0	**28.8**‡
	HEQ-Q	19.9	15.9	**22.5**
	HEQ-D	0.0	0.0	0.0

Table 4. A closer look at the training process for OR-QuAC.

Methods	Train			Dev	Test
	% Has Ans	% Hit Gold	Recall	% From Gold	% From Gold
Full S	100.00%	100.00%	1.0000	45.23%	27.46%
Span-match WS	72.96%	68.97%	0.7190	40.88%	28.80%
Learned WS	75.98%	67.24%	0.7187	39.89%	28.73%
Combined WS	75.52%	68.37%	0.7129	40.28%	28.39%

4.4 A Closer Look at the Training Process

We take a closer look at the training process, as shown in Table 4. We conduct this analysis on OR-QuAC only since we do not have the ground-truth passage relevance for CoQA. We observe that, "% Has Ans" are higher than "% Hit Gold" for all weak supervision methods, indicating all of them can identify weak answers in passages beyond the gold passages. In particular, our method can identify more weak answers than span match. We also notice that "% Hit Gold" is only slightly lower than "Recall", suggesting that most of the retrieved gold passages can yield a weak answer. This verifies the capability of weak supervisors. Finally, "% From Gold" are relatively low for all methods, indicating great potential for improvements.

4.5 Case Study and Error Analysis

We then conduct a qualitative analysis by presenting weak answers identified by the learned weak supervisor in Table 5 to better understand the weak supervision process. Example 1 and 2 show that our learned weak supervisor can find weak answers that are exactly the same or almost identical to the known answers when an exact match of the known answer exits, further validating our method can potentially cover span-match weak supervision. Example 3 shows that if an exact match does not exist, our method can find a weak answer that expresses

Table 5. Case study. Weak answers are found by the learned weak supervisor. Boldface denotes discrepancies and italic denotes paraphrasing.

	#	Questions and Answers	
Good	1	Question	Where was the album released?
		Known answer	On online forums and music sites
		Weak answer	On online forums and music sites
	2	Question	... mention anything else he starred in?
		Known answer	After starring ... the film adaptation of The Music Man
		Weak answer	After starring ... film adaptation of The Music Man **(1962)**
	3	Question	Where did he distribute the Cocaine?
		Known answer	Flying out planes several times, mainly between Colombia and Panama, along smuggling routes into the United States
		Weak answer	*He flew a plane himself several times, mainly between Colombia and Panama, in order to smuggle a load into the United States.*
Bad	4	Question	How long have people had clothes?
		Known answer	As long ago as 650 thousand years ago
		Weak answer	Around **170,000** years ago
	5	Question	What is data compression called?
		Known answer	Reducing the size of a data file
		Weak answer	**By using wavelets, a compression ratio**

the same meaning with the known answer. This is a case that a span-match weak supervisor would fail.

Example 4 shows that our method tends to focus on the lexical similarity only but get the fact wrong. Example 5 indicates our method sometimes finds a weak answer that is relevant to the known answer but cannot be considered as a good answer. These are the limitations of our method.

5 Conclusions and Future Work

In this work, we propose a learned weak supervision approach for open-retrieval conversational question answering. Extensive experiments on two datasets show that, although span-match weak supervision can handle span answers, it is not sufficient for freeform answers. Our learned weak supervisor is more flexible since it can handle both span answers and freeform answers. It is more powerful when combined with the span-match supervisor. For future work, we would like to enhance the performance of ORConvQA by studying more advanced history modeling methods and more effective weak supervision approaches.

Acknowledgments. This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF IIS-1715095. 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. The authors would like to thank Minghui Qiu for his constructive comments on this work.

References

1. Ahmad, A., Constant, N., Yang, Y., Cer, D.M.: ReQA: An Evaluation for End-to-End Answer Retrieval Models. ArXiv (2019)
2. Chen, D., Fisch, A., Weston, J., Bordes, A.: Reading wikipedia to answer open-domain uestions. In: ACL (2017)
3. Chen, L., Tang, Z., Yang, G.: Balancing reinforcement learning training experiences in interactive information retrieval. In: SIGIR (2020)
4. Chen, Y., Wu, L., Zaki, M.J.: GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension. ArXiv (2019)
5. Choi, E., et al.: QuAC: question answering in context. In: EMNLP (2018)
6. Clark, C., Gardner, M.: Simple and effective multi-paragraph reading comprehension. In: ACL (2017)
7. Cohen, D., Yang, L., Croft, W.B.: WikiPassageQA: a benchmark collection for research on non-factoid answer passage retrieval. In: SIGIR (2018)
8. Das, R., Dhuliawala, S., Zaheer, M., McCallum, A.: Multi-step retriever-reader interaction for scalable open-domain question answering. In: ICLR (2019)
9. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
10. Dhingra, B., Mazaitis, K., Cohen, W.W.: Quasar: Datasets for Question Answering by Search and Reading. ArXiv (2017)
11. Dunn, M., Sagun, L., Higgins, M., Güney, V.U., Cirik, V., Cho, K.: SearchQA: A New Q&A Dataset Augmented with Context from a Search Engine. ArXiv (2017)
12. Elgohary, A., Peskov, D., Boyd-Graber, J.L.: Can You Unpack That?. EMNLP/IJCNLP, Learning to Rewrite Questions-in-Context. In (2019)
13. Garg, S., Vu, T., Moschitti, A.: TANDA: transfer and adapt pre-trained transformer models for answer sentence selection. In: AAAI (2020)
14. Htut, P.M., Bowman, S.R., Cho, K.: Training a ranking function for open-domain question answering. In: NAACL-HLT (2018)
15. Huang, H.Y., Choi, E., tau Yih, W.: Flowqa: grasping flow in history for conversational machine comprehension. ArXiv (2018)
16. Joshi, M., Choi, E., Weld, D.S., Zettlemoyer, L.: TriviaQA: a large scale distantly supervised challenge dataset for reading comprehension. In: ACL (2017)
17. Karpukhin, V., et al.: Dense passage retrieval for open-domain question answering. In: EMNLP (2020)
18. Kratzwald, B., Feuerriegel, S.: Adaptive document retrieval for deep question answering. In: EMNLP (2018)
19. Krishna, K., Wieting, J., Iyyer, M.: Reformulating unsupervised style transfer as paraphrase generation. In: EMNLP (2020)
20. Kwiatkowski, T., et al.: Natural questions: a benchmark for question answering research. TACL **7**, 453–466 (2019)
21. Lan, Z.Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. ArXiv (2019)
22. Lee, J., Yun, S., Kim, H., Ko, M., Kang, J.: Ranking paragraphs for improving answer recall in open-domain question answering. In: EMNLP (2018)
23. Lee, K., Chang, M.W., Toutanova, K.: Latent retrieval for weakly supervised open domain question answering. In: ACL (2019)
24. Li, J., Qiu, L., Tang, B., Chen, D., Zhao, D., Yan, R.: Insufficient Data Can Also Rock!. AAAI, Learning to Converse Using Smaller Data with Augmentation. In (2019)

25. Lowe, R., Pow, N., Serban, I., Pineau, J.: The ubuntu dialogue corpus: a large dataset for research in unstructured multi-turn dialogue systems. In: SIGDIAL (2015)
26. Luo, J., Dong, X., Yang, G.: Learning to reinforce search effectiveness. In: ICTIR (2015)
27. Luo, J., Zhang, S., Yang, G.: Win-win search: dual-agent stochastic game in session search. In: SIGIR (2014)
28. Nguyen, T., et al.: MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. ArXiv (2016)
29. Qiu, M., et al.: Reinforced history backtracking for conversational question answering. In: AAAI (2021)
30. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.: Open-retrieval conversational question answering. In: SIGIR (2020)
31. Qu, C., et al.: Attentive history selection for conversational question answering. In: CIKM (2019)
32. Qu, C., Yang, L., Qiu, M., Croft, W.B., Zhang, Y., Iyyer, M.: BERT with history answer embedding for conversational question answering. In: SIGIR (2019)
33. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
34. Rajpurkar, P., Jia, R., Liang, P.: Know what you don't know: unanswerable questions for SQuAD. In: ACL (2018)
35. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100, 000+ questions for machine comprehension of text. In: EMNLP (2016)
36. Reddy, S., Chen, D., Manning, C.D.: CoQA: a conversational question answering challenge. TACL **7**, 249–266 (2018)
37. Shrivastava, A., Li, P.: Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In: NIPS (2014)
38. Tao, C., Wu, W., Xu, C., Hu, W., Zhao, D., Yan, R.: Multi-representation fusion network for multi-turn response selection in retrieval-based chatbots. In: WSDM (2019)
39. Trischler, A., et al.: NewsQA: a machine comprehension dataset. In: Rep4NLP-@ACL (2016)
40. Turc, I., Chang, M.W., Lee, K., Toutanova, K.: Well-Read Students Learn Better: On the Importance of Pre-training Compact Models. ArXiv (2019)
41. Vaswani, A., et al.: Attention is all you need. In: NIPS (2017)
42. Voorhees, E.M., Tice, D.M.: The TREC-8 question answering track evaluation. In: TREC (1999)
43. Wang, M., Smith, N.A., Mitamura, T.: What is the Jeopardy Model?. EMNLP-CoNLL, A Quasi-Synchronous Grammar for QA. In (2007)
44. Wang, S., et al.: R3: Reinforced ranker-reader for open-domain question answering. In: AAAI (2018)
45. Wieting, J., Gimpel, K.: ParaNMT-50M: pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In: ACL (2018)
46. Wolf, T., Sanh, V., Chaumond, J., Delangue, C.: Transfertransfo: a transfer learning approach for neural network based conversational agents. In: NeurIPS CAI Workshop (2018)
47. Wu, Y., Wu, W.Y., Zhou, M., Li, Z.: Sequential match network: a new architecture for multi-turn response selection in retrieval-based chatbots. In: ACL (2016)
48. Yan, R., Song, Y., Wu, H.: Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In: SIGIR (2016)

49. Yan, R., Song, Y., Zhou, X., Wu, H.: "Shall i be your chat companion?": towards an online human-computer conversation system. In: CIKM (2016)
50. Yang, L., et al.: Response ranking with deep matching networks and external knowledge in information-seeking conversation systems. In: SIGIR (2018)
51. Yang, L., et al.: A hybrid retrieval-generation neural conversation model. In: CIKM (2019)
52. Yang, L., et al.: IART: intent-aware response ranking with transformers in information-seeking conversation systems. In: WWW (2020)
53. Yang, W., et al.: End-to-end open-domain question answering with BERTserini. In: NAACL-HLT (2019)
54. Yang, Y., Yih, W.T., Meek, C.: WikiQA: a challenge dataset for open-domain question answering. In: EMNLP (2015)
55. Yeh, Y.T., Chen, Y.N.: FlowDelta: Modeling Flow Information Gain in Reasoning for Conversational Machine Comprehension. ArXiv (2019)
56. Zhou, J., Agichtein, E.: RLIRank: learning to rank with reinforcement learning for dynamic search. In: WWW (2020)
57. Zhu, C., Zeng, M., Huang, X.: SDNet: Contextualized Attention-based Deep Network for Conversational Question Answering. ArXiv (2018)

A Deep Analysis of an Explainable Retrieval Model for Precision Medicine Literature Search

Jiaming Qu, Jaime Arguello, and Yue Wang(✉)

University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA
jiaming@live.unc.edu, {jarguello,wangyue}@unc.edu

Abstract. Professional search queries are often formulated in a structured manner, where multiple aspects are combined in a logical form. The information need is often fulfilled by an initial retrieval stage followed by a complex reranking algorithm. In this paper, we analyze a simple, explainable reranking model that follows the structured search criterion. Different aspects of the criterion are predicted by machine learning classifiers, which are then combined through the logical form to predict document relevance. On three years of data from the TREC Precision Medicine literature search track (2017–2019), we show that the simple model consistently performs as well as LambdaMART rerankers. Furthermore, many black-box rerankers developed by top-ranked TREC teams can be replaced by this simple model without statistically significant performance change. Finally, we find that the model can achieve remarkably high performance even when manually labeled documents are very limited. Together, these findings suggest that leveraging the structure in professional search queries is a promising direction towards building explainable, label-efficient, and high-performance retrieval models for professional search tasks.

Keywords: Professional search · Precision medicine · Explainable IR

1 Introduction

Professional searchers often formulate complex information needs as a function of various concepts, or *aspects*. For example, in systematic reviews for evidence-based medicine, relevance criteria usually involve four elements, namely *p̲opulation*, *i̲ntervention*, *c̲omparison*, and *o̲utcome*, collectively known as the PICO elements [33]. In the TREC Precision Medicine track, a relevant research article should discuss *cancer treatment* and focus on subjects who had *the same cancer*, *the same genetic variation*, and *the same demographic* as the patient at hand [29]. In legal search, an inquiry often include multiple aspects such as *entity*, *event*, *time*, and *location* [18].

Professional searchers often use big, structured Boolean queries to express their relevance criteria. Each relevance aspect is encoded as a disjunctive clause of synonymous terms, which are then combined in a conjunctive clause to encode

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 544–557, 2021.
https://doi.org/10.1007/978-3-030-72113-8_36

the inclusion/exclusion criteria [31]. Boolean queries are popular among professional searchers because they provide an expressive and intuitive way of specifying complex logic. However, composing the "right" Boolean query takes experience and patience. Finding the right terms to include and exclude often requires extensive domain knowledge and many iterations of trial-and-error. As a result, searchers often use Boolean queries to retrieve an initial expansive set of results with high recall but low precision, and then manually sift through these results to identify relevant ones [25].

To refine Boolean search results and reduce a searcher's manual effort, machine learning and text mining approaches are proposed to rerank the initial search results. These include various learning-to-rank [8,20] and active learning [36,37] techniques. In these techniques, a ranking function is trained to distinguish relevant results from non-relevant ones, which helps prioritize relevant results for manual review [16]. However, building such a ranking function often requires a substantial amount of training data, which may not be available upfront. To create such training data, a searcher needs to manually assess the relevance of many initial search results. Once trained, the ranking function is often complex (e.g. gradient boosted decision trees or neural networks) and difficult to interpret, as it may not follow the searcher's decision logic expressed in the Boolean query. This lack of transparency is undesirable as many professional searchers value transparency more than pure ranking performance [31].

Our research is motivated by one overarching question: How can we support professional searchers with retrieval systems that leverage machine learning while preserving the transparency and interpretability of Boolean search? In a recent short paper, we explored an explainable retrieval model towards this goal [26]. The task setting is the TREC Precision Medicine (PM) track, where the relevance criteria involve multiple aspects combined in a decision logic [29]. The retrieval model learns separate machine learning classifiers to predict aspect-level relevance, and combines them through the decision logic to produce document-level relevance. Such a model can be easily explained as it closely resembles the searcher's relevance decision process. Preliminary results showed that the model performed as well as complex learning-to-rank models on 2018 PM track topics.

In this paper, we further investigate three research questions in order to gain deeper understanding of the proposed model beyond the preliminary work. Below we state these research questions and summarize the main findings.

1) *Is the proposed model generalizable to a broader range of PM topics?* By cross-validating on three years of data from the TREC PM track (2017–2019), we find that the model consistently performs as well as complex learning-to-rank models, confirming its generalizability beyond a specific year of data.
2) *How does the proposed model compare with those developed by teams in the PM track?* We find that the proposed model can replace competitive blackbox models submitted to the TREC PM track without significantly compromising retrieval performance, and sometimes even give performance gains.
3) *Does the proposed model require many labeled documents to learn well?* Through a learning curve analysis, we show that the model is substantially more

label-efficient than a conventional learning-to-rank model. It achieved remarkable retrieval performance even when trained on very few labeled documents.

2 Related Work

Professional Search Strategies. In professional search tasks, especially systematic literature reviews, searchers formulate structured information needs through complex Boolean queries, where concepts are encoded as disjunctive clauses of synonymous terms, and inclusion/exclusion criteria are built on top of these concepts [2,31]. Our approach aims to automate and assist in these tasks by replacing manually constructed query components with machine learning classifiers, and by logically explaining predictions using relevance aspects.

Explainable Information Retrieval. Recent works on explainable search and recommendation systems primarily focus on post-hoc explanation of highly complex ranking algorithms [12,34,39], where explanations are usually feature-based (e.g. highlighting query terms in search snippets [12]) and example-based (e.g. showing similar items that the user liked [39]). Our approach differs from these works in two ways. First, instead of explaining black-box models, we design inherently interpretable models. Second, the proposed approach can not only identify important high-level features, but also show intermediate decision steps.

Precision Medicine Literature Search. This work is inspired by the TREC PM track, where the task is to retrieve articles for cancer treatment planning. Most participating teams in this track employ a machine learning model to rerank documents retrieved by a simpler baseline. Some teams use the official relevance judgement criteria to fine-tune search results, e.g., filtering out documents that are not related to cancer treatment [9] or does not match the demographic information in the query [1]. High-performance reranking methods are often black-box models such as boosted decision trees [7,32], and deep neural networks [11,21,40], which means the decision logic is not interpretable. Here the proposed reranking model emulates the structured relevance judgment process in the TREC PM track, which is interpretable by design.

3 Structured Relevance in TREC PM Track

Since 2017, the TREC Precision Medicine (PM) track has been focusing on a specific type of professional search task in which relevance is *structured*, i.e., defined as a function of different aspects [28]. PM track organizers provided *structured* relevance judgements, where each document is assigned a relevance level (*not relevant*, *partially relevant*, *definitely relevant*) based on intermediate judgements on multiple aspects, as illustrated in Fig. 1. Each aspect takes a categorical outcome. For example, regarding the *Disease* aspect, a document may take one of four categories: (1) Exact (i.e., mentions the disease in the query), (2) More general (i.e., mentions a more general disease), (3) More specific (i.e., mentions a more specific disease), or (4) No disease (i.e., does not mention a

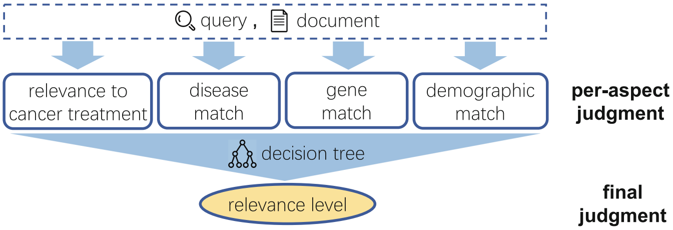


Fig. 1. Structured relevance judgment in the TREC Precision Medicine track.

Table 1. Relevance aspects and classifier features

Aspects	Outcomes	Classifier features
Relevance to cancer treatment	Human PM Animal PM Not PM	# "Human PM" keywords (n) # "Animal PM" keywords (n) # "Not PM" keywords (n)
Disease	Exact More general More specific No disease	# Query disease match (n) # Disease super-category match (n) # Disease sub-category match (n)
Gene	Exact Missing gene Missing variant Different variant	# Query gene and aliases match (n) Is variant in query (b) # Query variant match (n) # Other gene variants match (n) Is gene modification in query (b) # Gene modification match (n)
Demographic	Match Exclude Not discussed	Is gender mentioned in article (b) Is gender different in article (b) Is age mentioned in article (b) Difference in age (n) # Age group keywords match (n)

b: binary-valued, #: count of, n: real-valued, PM: precision medicine
keywords: terms with highest TF-IDF weights from each outcome

related disease). All aspects and corresponding outcomes are shown in the first two columns Table 1. Given a query, a document's *gold-standard* relevance level is determined by evaluating these intermediate judgements against a pre-defined cascade of rules (i.e., a decision tree). We refer the reader to Roberts et al. [29] for details about the judgment criteria and decision rules in the PM track.

The PM track released 30 queries with 22,642 judged documents in 2017, 50 queries with 22,429 judged documents in 2018, and 40 queries with 18,317 judged documents in 2019 for the subtask of PubMed abstract search. Aspect-level judgments were manually made by oncologists at the University of Texas MD Anderson Cancer Center. Then relevance levels were computed by passing intermediate aspect-level judgments through a pre-defined decision tree. We use these data in this work.

4 An Explainable Retrieval Model

The relevance judgment structure in Fig. 1 naturally inspires a new retrieval algorithm. The main idea is as follows. For each aspect, we train a multi-class classifier that predicts the categorical outcome (i.e., the second column in Table 1). Then we feed the predictions to the decision logic to compute document relevance. This approach has the potential to deliver good retrieval performance as it closely resembles the true relevance decision process. It is also highly explainable as its decision steps emulate those of human experts *by design*. Below we describe our implementation of the proposed retrieval algorithm. Its components – aspect classifiers and a decision tree – are learned from data.

4.1 Aspect Classifiers

Each classifier takes aspect-specific features extracted from a query-document pair. The model employs a small set of simple features per aspect (the third column in Table 1). All classifiers are *one-versus-rest* logistic regression models with regularization weight $C = 0.5$. SMOTE algorithm [6] is used to rebalance the severely skewed label distribution in each aspect (e.g., the majority of judged documents are non-relevant to cancer treatment, or *Not PM*).

4.2 Decision Tree

Building the Decision Tree. Instead of hand-coding the relevance decision logic into a decision tree, we learn the tree from structured relevance judgment data. The manually-assessed aspect outcomes are input features and the relevance level is the target category. We represent all outcomes as binary variables, so that each non-leaf node makes a binary decision on whether an outcome is true or false. Using information gain as the splitting criterion, we can learn a decision tree that achieves nearly 100% accuracy. This is not surprising, since aspect outcomes and relevance levels are known to be related through a simple decision logic. The learned tree structure is illustrated in Fig. 2.

Handling Predicted Outcomes. Such a decision tree assumes *manually-assessed* binary outcomes as inputs. To work as a retrieval component, the same tree should be able to handle classifier-predicted outcomes as inputs. In our context, these are confidence values predicted by our logistic regression models. The original decision process of the tree can be viewed as a 'walk' from the root to a leaf, making a binary decision at each non-leaf node. Now given confidence values predicted at each non-leaf node, we propose two ways of 'taking the walk':

- *Deterministic walk*: at each node, the walk follows the branch with confidence value of 50% or greater. In the end, the walk will reach a single leaf node, which determines a relevance level.
- *Probabilistic walk*: at each node, the walk will follow either branch with probability equal to the confidence value towards that branch. This (random) walk will reach every leaf node with non-zero probability, *i.e.* the product of all confidence values from the root to the leaf.

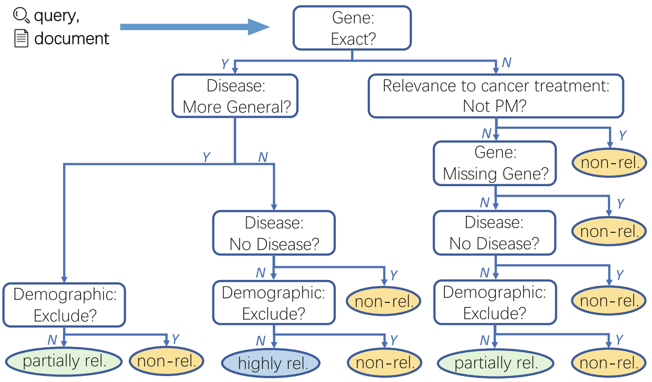


Fig. 2. The learned decision tree structure.

In terms of output, the decision tree predicts a probability distribution $p(r|q,d)$ over relevance levels $r \in \{$*not relevant*, *partially relevant*, *definitely relevant*$\}$ for a given query-document pair (q,d). The deterministic walk makes a *hard* prediction: $p(r^*|q,d) = 1$ for some r^* and 0 otherwise. We call this approach **Tree-hard**. The probabilistic walk makes a *soft* prediction: it predicts $p(r|q,d)$ as the probability of reaching any leaf associated with relevance level r. We call this approach **Tree-soft**.

Tree-hard and Tree-soft differ in their sensitivity to inaccurate predictions from our aspect classifiers. For Tree-hard, a single prediction error at any node will likely ‘sway’ the deterministic walk down a wrong path. For Tree-soft, when prediction errors occur, the probabilistic walk will still follow the right path with non-zero probability. In this regard, Tree-soft may have higher tolerance for inaccurate predictions.

These tree-based models offer natural ways of interpreting their decisions. To explain Tree-hard, one can show the single decision path it takes to predict relevance. To explain Tree-soft, one can show k most probable decision paths, each providing an alternative explanation. Upon close inspection, we found that the top-3 most probable paths down the tree account for an average of 80% of the total probability across all paths. In other words, while Tree-soft assigns a non-zero probability to each path, these probabilities tend to be *highly skewed* towards only a few.

Generating a Ranking Score. To rank documents, we need to generate a score for each (q,d). We use a variant of the approach in Li et al. [19]: $s(q,d) = \left[\sum_{r\in\{0,1,2\}} w_r \cdot p(r|q,d)\right] + b(q,d)$, where the weight w_r should increase with relevance level r. We define $r = 0, 1, 2$ as *not relevant*, *partially relevant*, and *definitely relevant*, respectively, and set $w_0 = 0$, $w_1 = 0.5$ and $w_2 = 1$. The first term $\left[\sum_{r\in\{0,1,2\}} w_r \cdot p(r|q,d)\right]$ is large if $p(r = 2|p,d)$ is large, *i.e.* the decision path unambiguously leads to a *definitely relevant* leaf. $b(q,d) \in [0,1]$ is the min-max scaled score generated from the initial retrieval stage (e.g. BM25). Overall, a large $s(q,d)$ indicates that d is relevant to q in a *clearly interpretable* manner.

5 Leave-One-Year-Out Cross-validation

In this section, we evaluate the proposed retrieval model on three years of PM track data (2017–2019). We perform leave-one-year-out cross validation, *e.g.*, training on 2017 and 2018 data combined and testing on 2019 data, and so on. Note that although all queries in three years share the same structure as shown in Fig. 1, no two queries are identical in all four aspects.

5.1 Initial Retrieval Stage

We first implement a simple initial retrieval stage by concatenating disease and gene terms to generate a search query, and then use the BM25 scoring function implemented in Apache Lucene to retrieve the top 500 documents from a Lucene index of the PubMed corpus. Then, we use tree-based and learning-to-rank models as rerankers after the initial retrieval stage.

5.2 Learning-to-Rank Baselines

To evaluate the reranking performances of the proposed models, we include classical learning-to-rank (LTR) approaches for comparison. LTR models are often highly complex (e.g. neural networks or ensemble models [4]) and make less explainable relevance predictions than the proposed approach.

LTR-High. The first baseline uses the prediction confidence values from our aspect classifiers as features. Again, each aspect has a fixed set of possible outcomes (2nd column in Table 1). Our aspect classifiers output a confidence value per outcome. The LTR-high approach uses a concatenation of these confidence values (14 total) as its input feature vector. We call this approach LTR-high because it uses *high*-level features that *directly* model the outcome of each aspect.

LTR-Low. The second baseline uses the raw features used by our four aspect classifiers (3rd column in Table 1). We call this approach LTR-low because it uses *low*-level features that *indirectly* model the outcome of each aspect.

We implement both LTR models with LambdaMART [4] in Lemur RankLib. To obtain the strongest baselines, we perform grid search for hyperparameters of each LTR model to maximize its precision@10 on 5-fold cross validation. These hyperparameters include the number of trees, the number of leaves in each tree, learning rate, and minimum leaf support.

5.3 Results

Three metrics were used to evaluate ranking performance: precision@10 (P@10), which focuses on precision at top ranks; R-precision (R-prec) and mean average precision (MAP), which emphasize both recall and precision. Table 2 shows evaluation results for the above algorithms when training on data from two out of three years and testing on the held-out year. We also included Lucene's BM25 baseline for comparison. When comparing approaches, we tested for statistical significance using Fisher's Randomization Test [35] (α = .05).

Table 2. Evaluation Results in terms of P@10, MAP and R-prec in three years. A ▲(▼), △(▽), and ∧(∨) denotes significantly better(worse) performance when comparing Tree-soft vs. Tree-hard, LTR-low vs. LTR-high, and Tree-soft vs. LTR-low, respectively.

Method	2017			2018			2019		
	P@10	MAP	R-prec	P@10	MAP	R-prec	P@10	MAP	R-prec
BM25	0.4100	0.1437	0.2374	0.5360	0.2273	0.3122	0.4550	0.1704	0.2394
LTR-high	0.4567	0.1433	0.2156	0.5080	0.2070	0.2864	0.4850	0.1533	0.2215
LTR-low	0.5330△	0.1780△	0.2616△	0.6240△	0.2530△	0.3281△	0.5200	0.1888△	0.2667△
Tree-hard	0.4200	0.1437	0.2321	0.5520	0.2284	0.3098	0.4400	0.1707	0.2422
Tree-soft	0.4333∨	0.1661▲	0.2551▲	0.6220▲	0.2622▲∧	0.3496▲∧	0.5100▲	0.1986▲	0.2736▲

Tree-Soft vs. Tree-Hard ▲(▼). First, we compare between two tree-based approaches (Sect. 4.2). Except for P@10 in 2017 (p = .555), Tree-soft outperformed Tree-hard across all years and metrics by a significant margin ($p < .001$). This result suggests an important trend—when traversing the "relevance decision tree" using *predicted* (vs. gold-standard) relevance aspects, it is better to traverse the tree *probabilistically* (i.e., using prediction confidence values) than to follow the single most confident path to a leaf node.

LTR-Low vs. LTR-High △(▽). Next, we compare between two LTR-based approaches (Sect. 5.2). Except for P@10 in 2019 (p = .248), LTR-low outperformed LTR-high across all years and metrics by a significant margin ($p < .001$). Interestingly, an LTR-based approach performed better with low-level features than high-level relevance aspects predicted by our aspect classifiers (Sect. 4.1).

Tree-Soft vs. LTR-Low ∧(∨). Finally, we compare between the better tree-based approach (Tree-soft) and the better LTR-based approach (LTR-low). In terms of P@10, Tree-soft performed significantly *worse* than LTR-low when testing on 2017 ($p < .005$). However, Tree-soft performed at the same level as LTR-low (i.e., no significant differences) when testing on 2018 and 2019. Note that LTR-low was expected to deliver a high P@10 because it was trained to optimize that metric, while Tree-soft was not. In terms of MAP and R-prec, Tree-soft performed at the same level as LTR-low across all years, and significantly better when testing on 2018 (MAP: $p < .005$; R-prec: $p < .005$).

Overall, the Tree-soft approach is consistently better than the Tree-hard approach, and its performance is comparable to LTR-low. This is no small feat considering that the Tree-soft approach is a much simpler (more interpretable) approach than the LTR-low approach which is a ensemble model using 500 decision trees in 2017 and 2018, and 1000 decision trees in 2019.

6 Replacing Black-Box Rerankers in TREC PM Track

A common approach in the TREC PM track has been to employ LTR models to rerank the top results produced by a simpler baseline [5,7,11,21,27,32,40]. While such an approach can effectively improve performance, the reranking model is often complex and not easily interpretable. Our goal in this section is to explore if these complex rerankers can be replaced with the proposed interpretable model without sacrificing performance.

6.1 Selecting Runs from Top-Ranked Teams

We compare the Tree-soft reranker (the better one in Sect. 5) against competitive TREC PM teams in 2019 (teams on the left side of Table 6 in [29]) that used a complex LTR model to rerank the retrieved results from a simpler baseline *and* submitted a "run" (i.e., result list) using that baseline. The initial retrieval baselines are denoted as **Original Baseline** and the corresponding reranking results are denoted as **Original Reranking**.

We identified six such teams. Table 3 shows the identified runs and reranking techniques. Teams that only submitted reranking results (e.g. BITEM_PM [5]) or did not use LTR reranking (e.g., imi_mug [22], CincyMedIR [38], ims_unipd [23]) were not selected. We also did not select runs from the julie-mug team as their baseline query has been extensively fine-tuned on previous relevance judgment data [10], and further reranking does not help [9].

For each selected team, we use the Tree-soft model (denoted as **Tree-soft Reranking**) to rerank the same initial results produced by the team's baseline. Components in Tree-soft are trained on 2017 and 2018 data combined, as all participants in 2019 had access to these data. This allows us to make head-to-head comparisons between Tree-soft and the rerankers used by these teams. In these comparisons, we vary the reranker but hold all other factors constant.

Table 3. Selected runs from PM 2019 submissions.

Team	Baseline run	Reranking run	Reranking technique
POZNAN [7]	SAsimpleLGD	SA_LGD_letor	LambdaMART in Terrier [24]
CSIROmed [32]	bm25_6801	Et_8435	Extremely randomized trees [13]
ECNU [40]	sa_base	sa_base_rr	Doc2Vec + cosine similarity [17]
CCNL [21]	ccnl_sa5	ccnl_sa4	SciBERT [3]
DUTIR [11]	DutirRun1	DutirRun3	Deep semantic matching [14,15]
UNC_SILS [27]	sils_run1	sils_run3	Logistic regression pointwise LTR

6.2 Results

Table 4 summarizes our results in terms of P@10, MAP, and R-precision. We compare the Tree-soft model against each team's original baseline and each team's reranker. When comparing approaches, we used Fisher's Randomization Test [35] ($\alpha = .05$) to test for statistical significance.

Tree-Soft Reranking vs. Original Baseline ▲(▼). In terms P@10, the Tree-soft model performed at the same level as all baselines. In terms MAP and R-prec, Tree-soft performed significantly better in six cases and significantly worse

Table 4. Reranking 2019 PM track submissions with Tree-soft. A $\blacktriangle(\blacktriangledown)$, and $\triangle(\triangledown)$ denotes significantly better (worse) performance when comparing Tree-soft vs. Original Baseline and Tree-soft vs. Original Reranking, respectively.

Team	Original baseline			Original reranking			Tree-soft reranking		
	P@10	MAP	R-prec	P@10	MAP	R-prec	P@10	MAP	R-prec
POZNAN	0.5400	0.2603	0.3092	0.5050	0.2117	0.2714	0.5700$^{\triangle}$	0.2542$^{\triangle}$	0.3240$^{\blacktriangle\triangle}$
CSIROmed	0.5250	0.2499	0.3029	0.5725	0.1279	0.1856	0.5375	0.2470$^{\triangle}$	0.3153$^{\blacktriangle\triangle}$
ECNU	0.5600	0.1767	0.2608	0.5600	0.1769	0.2610	0.5625	0.1848$^{\blacktriangle\triangle}$	0.2690$^{\blacktriangle\triangle}$
CCNL	0.5025	0.2197	0.2770	0.5775	0.2279	0.2886	0.5275	0.2154	0.2944$^{\blacktriangle}$
DUTIR	0.5800	0.2774	0.3266	0.5825	0.2775	0.3227	0.5775	0.2647$^{\blacktriangledown\triangledown}$	0.3261
UNC_SILS	0.5225	0.2216	0.2858	0.5925	0.2216	0.2757	0.5625	0.2278	0.3124$^{\blacktriangle\triangle}$

in one case (DUTIR in terms of MAP). It should be noted that the difference in MAP compared to the DUTIR baseline is significant but small.

Tree-Soft Reranking vs. Original Reranking $\triangle(\triangledown)$. In terms of P@10, the Tree-soft model performed significantly better than one reranker (i.e., POZNAN) and at the same level as the other rerankers. In terms of MAP and R-prec, the tree soft models performs significant better than seven cases and significantly worse in one case (DUTIR in terms of MAP). Again, it should be noted that the difference in MAP compared to the DUTIR reranker is significant but small.

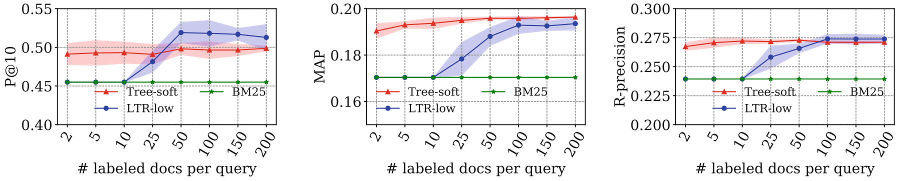
The results in Table 4 show more significant differences in MAP and R-prec than P@10. One possible explanation is that P@10 is a coarser metric (i.e., only 11 possible values). Another explanation is that P@10 considers only the top-10 results while MAP and R-prec consider additional results at lower ranks.

These results and those in Sect. 5.3 suggest that many black-box LTR models in this domain could be replaced by a simple and interpretable model without significant loss of performance, and in some cases even with performance gain. This finding is encouraging as medical literature search is a high-stake application that warrants an inherently interpretable retrieval model [30].

7 Learning Curve Analysis

The above experiments show that Tree-soft is an effective, explainable, and reusable reranker that performs as well as many black-box rerankers. However, such a model may seem more "data-hungry" than learning-to-rank approaches as it requires human experts to provide aspect-level judgments, which are more elaborate than relevance judgments. To label a document, an expert must first read it carefully to produce aspect-level labels, and then give the final relevance level. This is a time- and effort-consuming task. Therefore it would be ideal if the Tree-soft approach can achieve high performance without requiring a medical expert to label many documents. In this section, we investigate this problem through a learning curve analysis. Here, a learning curve shows the performance of a model when the amount of training data varies from small to large.

We sampled training documents from the 80 topics in 2017 and 2018 datasets combined, and tested on the 40 topics in 2019 dataset. For Tree-soft, we held the tree structure fixed as the decision logic is a piece of prior knowledge. We only re-trained our four aspect-level classifiers with newly sampled data. We include LTR-low for comparison as it is the stronger LTR model in Sect. 5. Note that Tree-soft and LTR-low may sample different documents for the same training data size. The training data for aspect-level classifiers in Tree-soft were sampled from the official relevance judgments, while those for LTR-low are sampled from initial retrieval results. Every time a new training dataset was sampled, we performed grid search of optimal hyperparameters for LTR-low with the same steps described in Sect. 5.2. At each data size, We took the mean and standard deviation of performance metrics over 8 random samples. To simulate settings where labeled documents are extremely scarce, we sampled two documents per query (one relevant and one non-relevant).

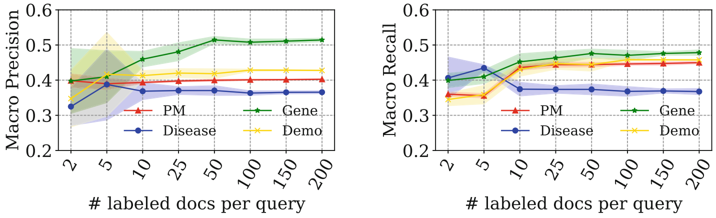


(a) P@10 (b) MAP (c) R-precision

Fig. 3. Learning curves of Tree-soft and LTR-low in terms of P@10, MAP, and R-prec. Color regions correspond to ±1 standard deviation around the mean.

Figure 3 shows the average P@10, MAP, and R-precision of Tree-soft and LTR-low when they are trained on increasing amounts of data. BM25 is included for comparison. Although Tree-soft and LTR-low learns from two kinds of labels (aspect-level labels for Tree-soft and document-level labels for LTR-low), both were generated through the same structured relevance judgment process (shown in Fig. 1) for every document. Therefore the number of labeled documents per query can be used as a unified measure of learning cost.

We observe that despite small training data sizes (# labeled documents per query = 2, 5, 10), Tree-soft performs surprisingly well and enjoys a large margin in all metrics compared to BM25 and LTR-low. In contrast, LTR-low does not perform well when the training data is small, due to severe overfitting of the LamdaMART reranker. In recall-oriented metrics (R-precision and MAP), Tree-soft performed on par with (if not better than) LTR-low across all training data sizes. In terms of P@10 (for which LTR-low was optimized), LTR-low outperformed Tree-soft only when more than 25 documents/query (2,000 documents for 80 queries) are labeled, which is a huge burden for manual annotation.



(a) Macro-averaged precision

(b) Macro-averaged recall

Fig. 4. Learning curves of aspect classifiers in terms of macro-averaged precision and recall. Color regions correspond to ±1 standard deviation around the mean.

We provide two explanations for this phenomenon. First, our aspect classifiers are simple linear models using a small set of features and therefore can be efficiently trained. The learning curves in Fig. 4 show that the classifiers reach stable performance after receiving only 10 labeled documents per query. The large performance variance in the early stage is due to small training data. In particular, the *Disease* classifier suffered a performance drop because its outcome label distribution is severely imbalanced, and increasing the data size exacerbates label imbalance. A second possible explanation for the Tree-soft model being robust with less training data stems from its pre-specified decision tree structure. Importantly, the structure encodes a piece of prior knowledge that LTR-low is completely unaware of: the correct decision-making procedure to combine (possibly noisy) aspect-level relevance into document-level relevance.

Overall, these results highlight the promise of leveraging the structure of professional search queries in a retrieval model – it makes the model not only more interpretable, but also more robust in handling noisy inputs and less hungry for training data.

8 Conclusion

In this paper, we analyzed a recently proposed explainable retrieval model that closely resembles a structured relevance judgment process, where the search query involves multiple aspects and document relevance is decided by a logical function of these aspects. Extensive experiments on TREC precision medicine track data show that the simple, explainable model can perform as well as many complex, black-box learning-to-rank models, and achieve high performance with much fewer labeled documents. These results point to a promising direction towards building effective, explainable, label-efficient retrieval algorithms for professional search tasks. In future work, we will evaluate the interpretability of the proposed retrieval model in prototype systems and user studies.

Acknowledgment. UNC SILS Kilgour Research Grant supported this work.

References

1. Agosti, M., Nunzio, G.M.D., Marchesin, S.: The university of Padua IMS research group at TREC 2018 precision medicine track (2018)
2. Aromataris, E., Riitano, D.: Systematic reviews: constructing a search strategy and searching for evidence. Am. J. Nurs. **114**(5), 49–56 (2014)
3. Beltagy, I., Lo, K., Cohan, A.: Scibert: a pretrained language model for scientific text. arXiv preprint arXiv:1903.10676 (2019)
4. Burges, C.J.: From ranknet to lambdarank to lambdamart: an overview. Learning **11**(23–581), 81 (2010)
5. Caucheteur, D., Pasche, E., Gobeill, J., Mottaz, A., Mottin, L., Ruch, P.: Designing retrieval models to contrast precision-driven ad hoc search vs. recall-driven treatment extraction in precision medicine. In: TREC (2019)
6. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
7. Cieslewicz, A., Dutkiewicz, J., Jedrzejek, C.: Poznan contribution to TREC-PM 2019. In: TREC (2019)
8. Dang, V., Bendersky, M., Croft, W.B.: Two-stage learning to rank for information retrieval. In: European Conference on Information Retrieval, pp. 423–434 (2013)
9. Faessler, E., Hahn, U., Oleynik, M.: Julie lab & med uni graz@ TREC 2019 precision medicine track. In: TREC (2019)
10. Faessler, E., Oleynik, M., Hahn, U.: What makes a top-performing precision medicine search engine? tracing main system features in a systematic way. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 459–468 (2020)
11. Feng, J., Yang, Z., Liu, Z., Luo, L., Lin, H., Wang, J.: Dutir at TREC 2019: Precision medicine track. In: TREC (2019)
12. Fernando, Z.T., Singh, J., Anand, A.: A study on the interpretability of neural retrieval models using deepshap. In: SIGIR 2019. pp. 1005–1008. ACM, New York, NY, USA (2019)
13. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**(1), 3–42 (2006)
14. Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep relevance matching model for adhoc retrieval. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 55–64 (2016)
15. Hui, K., Yates, A., Berberich, K., de Melo, G.: PACRR: a position-aware neural IR model for relevance matching. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1049–1058. Association for Computational Linguistics, Copenhagen, Denmark (September 2017)
16. Kanoulas, E., Li, D., Azzopardi, L., Spijker, R.: Clef 2019 technology assisted reviews in empirical medicine overview. In: CEUR Workshop Proceedings, vol. 2380 (2019)
17. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
18. LexisNexis: Developing a search with lexisnexis. Accessed September 2020). http://www.lexisnexis.com/bis-user-information/docs/developingasearch.pdf
19. Li, P., Wu, Q., Burges, C.J.: Mcrank: learning to rank using multiple classification and gradient boosting. In: Advances in Neural Information Processing Systems, pp. 897–904 (2008)

20. Liu, T.Y.: Learning to Rank for Information Retrieval. Springer, Beijing (2011) https://doi.org/10.1007/978-3-642-1467-3
21. Liu, X., Li, L., Yang, Z., Dong, S.: SCUT-CCNL at TREC 2019 precision medicine track. In: TREC (2019)
22. López-Úbeda, P., Vera-Ramos, J.A., López-García, P.: TREC 2019 precision medicine - medical university of Graz. In: TREC (2019)
23. Nunzio, G.M.D., Marchesin, S., Agosti, M.: Exploring how to combine query reformulations for precision medicine. In: TREC (2019)
24. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Johnson, D.: Terrier information retrieval platform. In: Losada, D.E., Fernández-Luna, J.M. (eds.) ECIR 2005. LNCS, vol. 3408, pp. 517–519. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31865-1_37
25. O'Mara-Eves, A., Thomas, J., McNaught, J., Miwa, M., Ananiadou, S.: Using text mining for study identification in systematic reviews: a systematic review of current approaches. Syst. Rev. **4**(1), 5 (2015)
26. Qu, J., Arguello, J., Wang, Y.: Towards explainable retrieval models for precision medicine literature search. In: Proceedings of the 43rd ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1593–1596 (2020)
27. Qu, J., Wang, Y.: UNC SILS at TREC 2019 precision medicine track. In: TREC (2019)
28. Roberts, K., et al.: Overview of the TREC 2017 precision medicine track (2017)
29. Roberts, K., et al.: Overview of the TREC 2019 precision medicine track. In: TREC (2019)
30. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Mach. Intell. **1**(5), 206–215 (2019)
31. Russell-Rose, T., Chamberlain, J., Azzopardi, L.: Information retrieval in the workplace: a comparison of professional search practices. Inf. Process. Manage. **54**(6), 1042–1057 (2018)
32. Rybinski, M., Karimi, S., Paris, C.: Csiro at 2019 TREC precision medicine track. In: TREC (2019)
33. Schardt, C., Adams, M.B., Owens, T., Keitz, S., Fontelo, P.: Utilization of the pico framework to improve searching pubmed for clinical questions. BMC Med. Inf. Decis. Making **7**(1), 16 (2007)
34. Singh, J., Anand, A.: Posthoc interpretability of learning to rank models using secondary training data. arXiv:1806.11330 (2018)
35. Smucker, M.D., Allan, J., Carterette, B.: A comparison of statistical significance tests for information retrieval evaluation. In: CIKM, pp. 623–632 (2007)
36. Tian, A., Lease, M.: Active learning to maximize accuracy vs. effort in interactive information retrieval. In: Proceedings of the 34th ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 145–154 (2011)
37. Wallace, B.C., Trikalinos, T.A., Lau, J., Brodley, C., Schmid, C.H.: Semi-automated screening of biomedical citations for systematic reviews. BMC Bioinform. **11**(1), 1–11 (2010)
38. Wu, D.T.Y., Su, W., Lee, J.J.: Retrieving scientific abstracts using venue- and concept-based approaches: Cincymedir at TREC 2019 precision medicine track. In: TREC (2019)
39. Zhang, Y., Chen, X.: Explainable recommendation: a survey and new perspectives. arXiv preprint: 1804.11192 (2018)
40. Zheng, Q., Li, Y., Hu, J., Yang, Y., He, L., Xue, Y.: ECNU-ICA team at TREC 2019 precision medicine track. In: TREC (2019)

A Transparent Logical Framework for Aspect-Oriented Product Ranking Based on User Reviews

Firas Sabbah(✉) and Norbert Fuhr

University of Duisburg-Essen, Duisburg, Germany
firas.sabbah@uni-due.de, norbert.fuhr@uni-due.de

Abstract. Customer reviews play a major role in online shopping, but there is hardly any support for aggregating the opinions of multiple reviewers, especially when the user is interested in certain aspects only. Current retrieval methods cannot handle the issues of limited credibility, contradictions and information omission when dealing with this type of documents. For addressing these problems, we investigate two multi-valued logic retrieval models. Subjective logic was specifically developed for considering uncertainty and subjective opinions. As an alternative, we regard a probabilistic version of a 4-valued logic addressing missing and inconsistent information. For an aspect-product pair, we get a probability distribution over the truth values and use them for ranking the search results. Our experimental results on a data set from the hotel domain show that our proposed approaches outperform the traditional keyword-based methods for the task of ranking items based on reviews. Moreover, the logic-based methods are more transparent than other approaches.

Keywords: User reviews · IR · Four-valued logic · Subjective logic

1 Introduction

Online shopping has become a usual scenario of internet users. Most of the online shops offering products or services have been converted to user-driven platforms [2] where users are able to not only buy online, but also to contribute and add their reviews to the items listed by the shops. These reviews are valuable because they express real user experiences and thus provide valuable information for others to make decisions [11]. However, when there are more than a few reviews, users would need an aggregated view of the set of reviews of a product – especially when they want to compare different products. Besides showing the distribution of star ratings, current systems provide no appropriate functionality. Moreover, users might be interested only in some of the aspects addressed in reviews, while the star rating summarizes all aspects a review talks about.

In this paper, we present a framework which starts with a set of predefined aspects for a product (or service) category, like e.g. performance, connectivity, battery etc. for laptops, or cleanliness, staff, comfort etc. for hotels. The system

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 558–571, 2021.
https://doi.org/10.1007/978-3-030-72113-8_37

then computes a score for each aspect-product pair, based on the product's reviews. In product search, these scores can be shown for each item in the result list, giving the user a quick survey over the strengths and weaknesses of each product according to its reviews. As with overall scores, the user can select aspects for being considered in the ranking or filtering of items, or s/he can navigate to reviews giving positive or negative comments on specific aspects. Moreover, review passages referring to the selected aspect(s) can be highlighted (see Table 1).

Aggregating the opinions of a set of reviews regarding specific aspects brings several challenges that need to be addressed: 1) The set of reviews of a product is likely to contain contradictions. 2) Some reviews have questionable credibility (e.g. from paid-for reviewers). 3) Most product reviews do not cover all aspects of a product, and thus we have to deal with missing information.

For addressing these problems, we investigate two approaches based on multivalued logics, which explicitly address the issues of contradiction, credibility and missing information. More specifically, we regard a) probabilistic 4-valued logic with additional truth values for *inconsistent* and *unknown*, and b) subjective logic, which defines the subjectivity of opinions as probabilities.

We compute a truth value for each aspect-review pair and then aggregate these values over all reviews of a product, in order to compute probabilistic truth values for each aspect-product pair. This model is tested on a dataset from a hotel booking site, and the results are compared to that of traditional keyword-based methods.

The remainder of this paper is structured as follows: After giving a survey over related work, Sect. 3 presents our basic approach by first briefly introducing the two logics used and then describing our methods for deriving the probabilistic truth values from reviews. The experimental setting is presented in Sect. 4, followed by the discussion of results in Sect. 5. The paper concludes with a summary of the findings and an outlook on follow-up work.

2 Related Work

User reviews are one of the most popular ways for consumers to exchange their experiences on products [5,6], but they represent mostly subjective opinions of strangers [4]. Moreover, some reviews might be erroneous, or intentionally misleading, e.g. for commercial reasons. [24] gives a good survey over this issue and describes methods for estimating credibility. For example, [10] presents a credibility assessment method for recommender systems based on the user profile or user expertise. Other works like e.g. [17] demonstrate powerful linguistic features for assessing credibility.

Positive and well-reasoned reviews have a positive influence on the likelihood of customer purchase decision [19]. However, negative reviews are not always of negative influence. For instance, [21] found that not just positive reviews have a positive impact on sales, but also negative ones. [22] showed that negative reviews are more powerful in reducing sales than positive ones in increasing them.

Four-valued logic was proposed in [8] to deal with contradictions that appear when aggregating the information from different documents [20]. Subjective logic [12–14] has been widely discussed and used for fusing information of different sources. Trust assessment and network security are common domains where subjective logic is used. However, it is rarely discussed in the domains of information retrieval or recommender systems [9].

Ranking products based on an analysis of reviews has gained attractiveness as a research topic as the online products and its related reviews are increasing enormously. In [25], researchers introduced feature-based approaches to rank products by analyzing the sentiments of reviews and considering the helpfulness of votes, review date and other features. [16] followed a user-based approach to build weighted and directed graphs in order to rank the products. Similarly, in [26] product features were manually identified within categories in order to construct directed and weighted graphs. [18] proposed a method based on sentiment analysis and intuitionistic fuzzy set theory to rank products based on their reviews. These approaches are mainly based on feature extraction and machine learning, but provide only a single overall ranking, while lacking transparency. In contrast, our logic-based models distinguish between different aspects of a product category and implement an explicit and transparent treatment of contradiction, missing information and credibility.

3 Logic-Based Approaches for Review Indexing

In this section, we first introduce the theories that we use as bases for our retrieval approaches, namely four-valued logic (4vL) and subjective logic (SL), and then describe the estimation of their probabilistic parameters based on reviews.

Four-Valued Logic. Belnap's relevance logic [3] is a 4vL designed to aggregate information from multiple information sources, like the different reviews for a product in our case. Belnap complemented the two standard truth values *true* and *false* by *inconsistent* and *unknown. Inconsistent* means that we have both *true* and *false* values from different sources (e.g. reviews on one aspect), while *unknown* refers to the fact that we are missing information.

For applying 4vL, we assign truth values to each pair of an aspect a and a review r: *true* if the review talks positively about the aspect, and *false* in the contrary case; *unknown* is assigned if the aspect is not mentioned in the review. Below, we denote these three truth values by t, f, and u, respectively. In addition, we compute probabilities for the truth values assigned, which reflect the strength of the sentiment of the reviewer's comment on the specific aspect. $P(t|a,r)$ reflects the positivity of r wrt. a and $P(f|a,r)$ the negativity, respectively. Normally, only one of these values will be different from 0, expressing a clearly positive or negative opinion. If both of these values are 0, the aspect is not mentioned at all in the review, and if both are greater than 0, we would have mixed feelings in a single review (e.g. 'brilliant display, but low resolution').

Furthermore, we always have $P(u|r,a) = 1 - P(t|a,r) - P(f|a,r)$. This way, our method clearly distinguishes between the case when an aspect is not mentioned in a review, and the case when there are negative comments. This is in stark contrast to standard IR methods which ignore negation, and only distinguish between absence and presence of a term. In 4vL, absence of an aspect in a review leads to $P(u|r,a) = 1$ and $P(t|a,r) = P(f|a,r) = 0$.

Given the probabilities for the three truth values (*true*, *false* and *unknown*) for each review, we need a method to aggregate these values for a set of reviews. Here we also have to consider the credibility $cr(r)$ of a review, the probability that the claims in r are actually true.

For aggregating the reviews for an aspect, we regard two possibilities

- The reviews actually refer to different instances of an item, e.g. some buyers of a hard disk might complain that they experienced a disk crash after a short time. In case other users have no such problems, there is no contradiction between the reviews – there is just a certain percentage of bad devices. In probabilistic terms, the different reviews can be modelled as *disjoint* events.
- The reviews are regarded as independent comments on the same instance (e.g. the content of a book), and there may be contradictory views. In this case, we regard the reviews as *independent* events which may overlap in event space, and in case they have different truth values, this contradiction leads to the truth value *inconsistent*.

Figure 1 illustrates the aggregation of probabilities from two reviews for both cases, for which we now give the precise definitions.

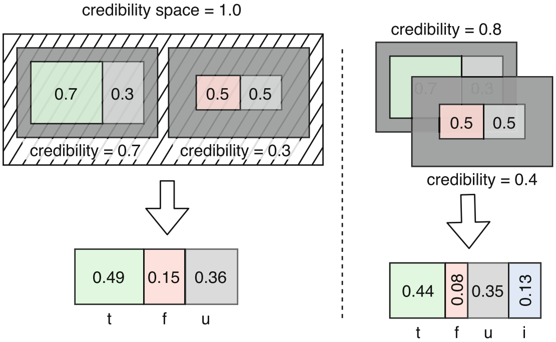


Fig. 1. Disjoint (left) and independent (right) 4vL credibility spaces

Disjoint Case. Here we have to transform the original credibility values so that their sum does not exceed 1. (If the sum is less than 1, we do not completely trust in reviews, e.g. when there are only a few of them.) Let R denote the set of reviews for a specific item, and β the overall trust in reviews, then we compute

$$cr_d(r) = \beta \cdot cr(r) / \sum\nolimits_{r' \in R} cr(r').$$

Assuming disjointness of reviews, we can now compute the truth values for an aspect of an item by aggregating over the set of reviews R in the following way:

$$\begin{aligned} P(t|a,R) &= \sum\nolimits_{r \in R} P(t|a,r) \cdot cr_d(r) \\ P(f|a,R) &= \sum\nolimits_{r \in R} P(f|a,r) \cdot cr_d(r) \\ P(u|a,R) &= 1 - P(t|a,R) - P(f|a,R) \end{aligned}$$

Independent Case. Here we have to consider all possible combinations of the truth values of the reviews. For two reviews, we get overall *true* for the combinations (t,t), (t,u) or (u,t); overall *false* results from (f,f), (f,u) or (u,f); furthermore, *inconsistent* results from (t,f) or (f,t) and *unknown* from (u,u). As a simple example, assume that for some aspect, we have a positive review r_1 with $P(t|a,r_1) = 0.6$ and a negative one r_2 with $P(f|a,r_2) = 0.7$. Assuming that the reviews are independent events, we would get $P(t|a,R) = 0.6 \cdot (1-0.3)$, $P(f|a,R) = (1-0.6) \cdot 0.7$, $P(i|a,R) = 0.6 \cdot 0.7$ and $P(u|a,R) = (1-0.6) \cdot (1-0.7)$ (assuming that both reviews have credibility 1).

For the general case, we have to regard all paths through all reviews, distinguish between true, false, unknown and inconsistent paths[1], and then sum up the probabilities of all paths for a specific truth value.

The paths for the four truth values are defined according to the following rules:

- *True*: At least one *true* review, no *false* reviews, and zero or more *unknown*s.
- *False*: At least one *false* review, no *true* reviews, and zero or more *unknown*s.
- *Unknown*: All *unknown* reviews.
- *Inconsistent*: At least one *true* and one *false* review, in addition to zero or more *unknown*s.

As a final step, the probabilities of the truth values are adjusted by removing the accumulated unknown knowledge from the accumulated *true* and *false* reviews. The following formulas summarise the process of creating probabilities of the truth values in the independent case:

[1] A path here is the combination of one truth value from each review.

$$P(u|a,R) = \prod_{r \in R} 1 - (P(t|a,r) \cdot cr(r) + P(f|a,r) \cdot cr(r)) \tag{1}$$

$$P(t|a,R) = \left(\prod_{r \in R} 1 - (P(f|a,r) \cdot cr(r)) \right) - P(u|a,R) \tag{2}$$

$$P(f|a,R) = \left(\prod_{r \in R} 1 - (P(t|a,r) \cdot cr(r)) \right) - P(u|a,R) \tag{3}$$

$$P(i|a,R) = 1 - P(t|a,R) - P(f|a,R) - P(u|a,R) \tag{4}$$

Subjective Logic. SL [12–14] is a probabilistic logic that considers uncertainty and subjective opinions. It provides definitions for binomial and multinomial cases. For information retrieval tasks, query matching is regarded as a binomial case (positive and negative).

In SL, the truth of a binomial opinion about proposition x is defined as a tuple $\omega_x = (b, d, u, a)$, with $b, d, u, a \in [0, 1]$ and $b + d + u = 1$.

- b represents the belief mass in support of x being true.
- d is the disbelief mass in support of x being false.
- u is the uncertainty mass about the probability of x.
- a is the prior probability of x.

The probability projection a.k.a. expected probability of a binomial proposition x is defined as $P(x) = b + a \cdot u$. This value represents the degree of certainty wrt. the truth of proposition x.

To combine opinions from various sources, SL provides many fusion operators for binomial opinions. Here we make use of two common operators: *cumulative* and *averaging fusion* which are comparable to the independent and disjoint cases in 4vL, respectively. If the observations i.e. opinions are about the same state of an object, cumulative fusion is used. Averaging fusion should be applied if the observations are about different states of an object (like e.g. reviewing different instances of a device, as mentioned above). Let x be a proposition and $\omega_x^A = (b^A, d^A, u^A, a^A)$ and $\omega_x^B = (b^B, d^B, u^B, a^B)$ be source A and B's respective opinions over the same proposition x. Furthermore, let us assume that there is some uncertainty, i.e. $u^A \neq 0$ or $u^B \neq 0$. The following table shows the definitions of the cumulative opinion $\omega_x^{A \oplus B}$ and the average opinion $\omega_x^{A \varnothing B}$:

	$\omega_x^{(A \oplus B)}$	$\omega_x^{(A \varnothing B)}$
b	$\frac{b^A u^B + b^B u^A}{u^A + u^B - u^A u^B}$	$\frac{b^A u^B + b^B u^A}{u^A + u^B}$
d	$\frac{d^A u^B + d^B u^A}{u^A + u^B - u^A u^B}$	$\frac{d^A u^B + d^B u^A}{u^A + u^B}$
u	$\frac{u^A u^B}{u^A + u^B - u^A u^B}$	$\frac{2u^A u^B}{u^A + u^B}$
a	$\frac{a^A u^B + a^B u^A - (a^A + a^B) u^A u^B}{u^A + u^B - 2u^A u^B}$	$\frac{a^A + a^B}{2}$

In the absence of uncertainty ($u^A = 0$ and $u^B = 0$), different formulas are used to handle dogmatic opinions. As IR is hardly ever about certain information, we do not regard this case here and refer the interested reader to the original papers [12,14,23].

As a concrete example for these fusion operators, let us assume a user is searching for a high-performance laptop. One of the offered laptops has two reviews. Review X reports the high-performance of the laptop with 0.9 confidence, while review Y reports the low-performance with 0.7 confidence. Assuming the prior knowledge about this aspect to be 0.5, cumulative fusion yields $\omega^{X \oplus Y}_{high-performance} = (0.73, 0.19, 0.08, 0.5)$; on the other hand, averaging fusion would lead to $\omega^{X \varnothing Y}_{high-performance} = (0.67, 0.17, 0.15, 0.5)$.

For considering the trustworthiness (credibility) of information, SL has proposed trust networks [15]. Here we only regard the so-called referral trust, for modelling a user's trust in a review. In SL terminology, an agent A's referral trust about agent B (i.e. B's credibility in the eyes of A) is represented as a subjective opinion and denoted by ω^A_B. The projected probability of ω^A_B is defined as: $P^A_B = b^A_B + u^A_B \cdot a^A_B$. The opinion ω^B_x is B's opinion on the proposition x (functional trust) as x is recommended by B to A. The function that yields the trust-discounted opinion $\omega^{[A;B]}_x = \omega^A_B \otimes \omega^B_x$ is defined with the following components:

$$\begin{aligned} b^{[A;B]}_x &= \qquad P^A_B b^B_x & \qquad d^{[A;B]}_x &= P^A_B d^B_x \\ u^{[A;B]}_x &= 1 - b^{[A;B]}_x - d^{[A;B]}_x & \qquad a^{[A;B]}_x &= a^B_x \end{aligned}$$

As a follow-up of the high-performance laptop example, let us assume that user A searching for a laptop trusts the opinion of reviewer X by 70% as a "belief" and the remaining 30% as "unsure". Here, the user's overall trust of the information of review X is $\omega^A_X \otimes \omega^X_{high-performance} = (0.7, 0, 0.3, 0.5) \otimes (0.9, 0, 0.1, 0.5) = (0.765, 0, 0.235, 0.5)$.

Computing Aspect-Item Scores. After having presented the logical foundations of our approach, we now describe the actual application of these concepts for the purpose of aspect-wise aggregation of reviews. Figure 2 gives an example of the whole indexing process for the two logics.

Each aspect a is represented by a set of keywords K_a terms, which can be derived e.g. by application of current NLP methods (see below)

For a specific review r, let r^+ denote the set of terms occurring with positive sentiment, and r^- those with negative sentiment. Furthermore, $w(k|r)$ is the term weight of keyword k in r. Then we can compute the positive and negative sentiment of r wrt. aspect a as follows:

$$sl^+(a,r) = \alpha \sum_{k \in K_a \cap r^+} w(k|r), \qquad sl^-(a,r) = \alpha \sum_{k \in K_a \cap r^-} w(k|r) \tag{5}$$

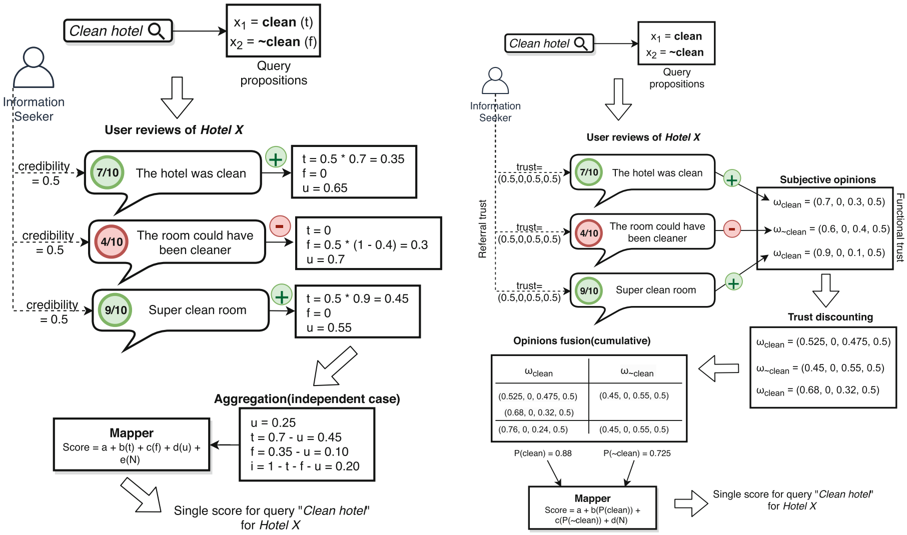


Fig. 2. An example of information extraction from reviews to evaluate a user query through 4-valued and subjective logic approaches.

Here α is a normalization constant depending on the actual term weighting method used, which ensures that both sl^+ and sl^- can be interpreted as probabilities such that $0 \leq sl^+(a,r) + sl^-(a,r) \leq 1$.

In the 4vL case, we can directly assign these two values to the corresponding probabilities: $P(t|a,r) = sl^+(a,r)$ and $P(f|a,r) = sl^-(a,r)$, respectively; we also set $P(u|r,a) = 1 - P(t|a,r) - P(f|a,r)$, and $P(i|r,a) = 0$ (a review cannot contradict itself).

For SL, we extract a positive-supportive opinion $\omega_P = (sl^+(a,r), 0, 1 - sl^+(a,r), 0.5)$ and a negative-supportive opinion $\omega_N = (sl^-(a,r), 0, 1 - sl^-(a,r), 0.5)$.

For credibility estimation, one of the methods cited in Sect. 2 can be used.

4 Experiments

In order to evaluate the aspect-specific weighting of items based on reviews, we need a dataset that includes aspect scores for products. Fortunately, there is a hotel booking site (Booking.com) containing these values, and we used the subset from [7] for our evaluation. This dataset contains 839K reviews of 11.5K hotels in Berlin, Brussels, Barcelona, London and Rome. Each review consists of a title/summary, a section of positive and a section of negative points. For our experiments described in the following, we only considered the latter two parts. This allows us to do a proper evaluation of the logic-based weighting formulas, which are the focus of this paper; otherwise, if we had to apply sentiment analysis, the intrinsic difficulties of this method (e.g., ‘low price’ vs. ‘low comfort’) could have had an unknown effect on the experimental results.

The review form in Booking.com asks the users to rate the following seven aspects of the hotel they stayed in: cleanliness, comfort, staff, value for money, location, wifi, and facilities. Users rate each of these aspects on a 4-point Likert scale, but these individual judgments are not available in the dataset; for each aspect, we only have the overall score aggregated over all reviews of a hotel. In our experiments, we use these aspect ratings as ground truth for the aspect scores to be estimated by the methods regarded here.

Table 1. Sample reviews(b) mapped to aspects based on an aspects dictionary(a).

Aspect	Keywords(including misspelled terms)
cleanliness	clean, unclean, smelly, cleaning, neat, clen, dirty, ...
location	lokation, locatio, locstion, centrality, situated, ...
staff	reciption, fiendly, staff, owner, managers, crew, ...
	...
facilities	gym, bathroom, cooking, shower, garden, entrance, ...

(a)

Review			Mapped aspects	
Positives(+)	Negatives(-)	score	Positives(+)	Negatives(-)
Man on **reception** was smiley and very helpful. Breakfast was ok.	**Bathroom** **smelt** like sewers had to keep the door closed. Ants in the room.	5	staff	facilities, cleanliness
Staff were great.	Could do with a small **gym** on site.	9.6	staff	facilities

(b)

For mapping the texts from the positive and negative sections onto aspects, a word2vec model was trained using the text from the reviews. This method generated a set of related terms for each aspect label (see Table 1). These are the aspect-specific keywords K_a of our approach.

For term weighting, we used three methods: 1) raw term frequencies (tf), 2) tfidf, and 3) Okapi's BM25.

The logic-based models regarded here all consider review-specific credibility values. However, Booking.com allows only confirmed customers to write reviews. It also verifies the authenticity of reviews before publishing them[2]. Thus, we assume the credibility values being equal for all reviews: For 4vL, we choose a

[2] https://partner.booking.com/en-us/help/guest-reviews/what-are-guest-reviews-and-who-can-write-one, last accessed on Sep. 10th 2020.

credibility of 1.0 in the independent case, and distribute a value of 1.0 equally over all reviews in the disjoint case. For SL, we use a fixed value 1.0 for trust of each proposition, which is represented by a subjective logic opinion $trust = (1.0, 0, 0, 0.5)$. In this case, no trust discounting operation is necessary, as the trust is neutralised.

Both logic methods yield a vector of probabilities for the different truth values of a hotel-aspect pair. As we want to relate these estimates to the corresponding aspect scores in the dataset, we have to map the probability vectors onto a single value, i.e. we want to predict the corresponding aspect ratings. For this purpose, we tested linear regression, kNN, SVR and random forest, and found that simple linear regression gave the best results; so we used this method for all experiments described in the following, and applied it in the same 10-fold cross validation setup for all methods tested.

Baselines and Evaluation Metric. We compare our logic-based method to different baselines where in each case, we index the positive and the negative sections of each review separately. Then, we make use of the term weights of traditional approaches (tf, tfidf and BM25) to compute positive and negative scores for each hotel. These scores are computed as follows:

$$Sc^{+}(h,a) = \frac{\sum_{r \in R} \sum_{k \in K_a \cap r^{+}} w(k^{+}|r)}{N_a^{+}}, \quad Sc^{-}(h,a) = \frac{\sum_{r \in R} \sum_{k \in K_a \cap r^{-}} w(k^{-}|r)}{N_a^{-}}$$

Here $Sc^{+}(h,a)$ and $Sc^{-}(h,a)$ are the positive and negative scores of a hotel h for an aspect a. R is the set of reviews of hotel h, and N_a^{+} and N_a^{-} are the numbers of reviews with positive/negative comments on aspect a, respectively. K_a, r^{+}, r^{-} as well as the term weighting function $w(.)$ are defined as in Eq. 5.

We used these scores and the number of positive and negative reviews in a feature vector and then applied linear regression for predicting the aspect ratings of hotels, in the same setting as with the logic-based methods.

For measuring the quality of the prediction, we adopt the well-known Coefficient of Determination (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{6}$$

Here $\hat{y}_i$ is the predicted value of a (hotel,aspect) pair i, and y_i is the corresponding ground truth value. $\bar{y}$ is the mean of all ground truth values.

Table 2. R^2 scores of logic-based approaches compared with baselines

				4-valued logic						Subjective logic					
	Baselines			Independent			Disjoint			Cumulative fusion			Averaging fusion		
Aspect	tf	tfidf	bm25	tf	tfidf	bm25	tf	tfidf	bm25	tf	tfidf	bm25	tf	tfidf	bm25
Cleanliness	0.312	0.349	0.278	**0.369**	0.298	0.333	0.141	0.136	0.127	**0.369**	0.336	0.329	0.09	0.11	0.076
Comfort	0.111	0.108	0.078	**0.173**	0.113	0.132	0.092	0.091	0.072	0.145	0.123	0.109	0.047	0.051	0.021
Staff	0.313	0.342	0.257	**0.368**	0.306	0.335	0.111	0.111	0.116	0.36	0.349	0.352	0.145	0.154	0.109
v. f. money	0.076	0.085	0.07	0.09	**0.114**	0.073	0.038	0.037	0.034	0.089	0.113	0.074	0.044	0.046	0.041
Location	0.274	0.272	0.193	**0.305**	0.174	0.279	0.172	0.164	0.124	0.286	0.22	0.247	0.151	0.158	0.067
Wifi	0.155	0.157	0.128	0.169	0.156	0.133	0.057	0.058	0.05	**0.175**	0.168	0.147	0.074	0.073	0.06
Facilities	0.197	0.221	0.145	**0.279**	0.239	0.22	0.088	0.091	0.077	0.259	0.244	0.216	0.031	0.036	0.02
Avg. score	0.206	0.219	0.164	**0.25**	0.2	0.215	0.1	0.098	0.086	0.241	0.222	0.211	0.083	0.09	0.056

5 Results and Discussion

Table 2 presents the R^2 scores of the linear models in each tested aspect. The results show that 4vL in the independent case is outperforming the baseline approaches in all aspects. Subjective logic in the cumulative fusion case also achieves close results. As we can see, the most appropriate term weighting method to be used for this task is the term frequency alone. This might be due to the aspect-oriented word embedding method used for determining the meaningful terms, which cannot be improved any further by idf weighting.

Table 3. Regression factors of the two logics for independent / cumulative fusion with tf weights.

Aspect	4vL			SL(b+ua)	
	True	False	Unkn.	Pos.	Neg.
Cleanliness	2.15	−3.91	1.08	4.21	−7.86
Comfort	2.17	−7.57	1.51	2.62	−7.97
Staff	1.97	−4.78	0.83	4.39	−7.49
V. f. money	0.54	−0.89	0.63	0.35	−2.68
Location	1.81	−6.37	0.88	3.85	−6.89
Wifi	1.75	−3.99	−0.64	7.90	−8.02
Facilities	1.73	−4.54	0.17	5.87	−8.06

The figures for the disjoint and averaging fusion cases show that their performance is below that of the baselines. The performance difference between independent and disjoint 4vL as well as between cumulative and averaging fusion in SL suggests that in each case, the former method is more appropriate here. This can be explained by noticing that all users are more or less reviewing the same item – possible differences between rooms or the staff behavior on different days seem to be minor and have no effect.

Looking at the regression factors of the logical models shown in Table 3, we see that with the exception of the aspect 'value for money' (overall rating is the best predictor for this aspect), the weighting factors are very similar for all aspects: the false/negative values have higher weights than the true/positive ones. The weight for *unknown* is mostly positive, indicating that the default for reviews is on the positive side. Overall, negative reviews have the biggest influence on the score of an item, thus confirming the findings from [22]. (There are no coefficients for *inconsistent*, since its probability is linearly dependent on the other three probabilities.)

A powerful characteristic of our models is that they consider creating both-polarity interpretations of the unknown information. As we have seen in the model construction, 4-valued logic constructs the *true* or *false* knowledge by the assumption that at least one of the reviews is classified either as *true* or *false* and the other reviews are classified as *unknown*. The *unknown* probability reduces the values of *true* and *false*; however, this gives the model an opportunity for different interpretations of the reviews classified as *unknown*. On the other hand, subjective logic also distinguishes between positive and negative evidence, and it also assumes that the *uncertainty* about an event can be interpreted as *belief* probability regarding the aspect under consideration. The major difference between the two approaches is the handling of missing information: in 4vL, this is modelled via an explicit truth value for the aggregated reviews, which also could be made transparent to the end user, especially for differentiating between missing and positive information. In contrast, SL handles missing information via the prior beliefs for positive and negative opinions; thus, especially for items with belief values in the medium range, we do not know if these scores are supported by actual comments in the reviews, or if they are mainly the result of the prior beliefs.

6 Conclusion and Future Work

Product reviews influence the decision of customers who utilize them in choosing the best product or service among the different available alternatives. A simple ranking by overall ratings might often not be helpful, as there are usually several aspects, which are not all of equal importance for a specific user. Current systems offer ranking, filtering and navigation to reviews with specific scores, but only for the overall scores, whereas our approach allows to implement the same functionality per aspect.

The logic-based models presented in this paper demonstrate superior performance in comparison to traditional keyword-based methods. While it might be possible to achieve slightly better one-dimensional rankings via tuning deep learning methods (in case there is enough training data available), the advantage of the logic-based approaches is that they handle contradictions, omission and credibility in a transparent way, which also can be made visible for the end-user. For researchers, this transparency allows for a better understanding of the problems, identifying the major influencing factors and spotting possible improvements (e.g. base rate probability of observations for different weighting methods and its possible effect on uncertainty interpretation). As IR research is paying more attention to the transparency of the methods employed [1], our work is a contribution along these lines.

This experimental study has been applied to a single specific collection, where we have pre-defined aspects and aspect-specific ratings for each item. For new applications, first, the aspects have to be defined manually, and then associated keywords are classified (either manually or automatically) into positive and negative ones. We are currently working on the application of our approach for such new applications (e.g. for specific product categories in online shops). The aspect-specific ratings in the Booking.com case are mainly needed for evaluation; here we also used them for tuning the mapping of the probability vectors onto a linear scale.

This paper has focused on the general suitability of logic-based approaches for handling credibility, contradictions and omissions in product reviews. Further work will address the improvement of the indexing and weighting methods, as well as the estimation and integration of credibility values. Extending our aspect-based methods for answering arbitrary queries referring to multiple aspects and containing additional conditions is straightforward, given the logic foundation. We will also perform user studies with real user queries, and explore multi-dimensional ranking.

Acknowledgements. This work was supported by the German Research Foundation (DFG) under grant No. GRK 2167, Research Training Group “User-Centred Social Media”.

References

1. Research frontiers in information retrieval - report from the third strategic workshop on information retrieval in lorne (swirl 2018). SIGIR Forum, vol. 52, no. 1, pp. 34–90 (2018). http://sigir.org/wp-content/uploads/2018/07/p034.pdf
2. Akehurst, G.: User generated content: the use of blogs for tourism organisations and tourism consumers. Serv. Bus. **3**(1), 51 (2009)
3. Belnap, N.D.: A useful four-valued logic. In: Modern Uses of Multiple-valued Logic, pp. 5–37. Springer, Berlin (1977)
4. Burgess, S., Sellitto, C., Cox, C., Buultjens, J.: Trust perceptions of online travel information by different content creators: some social and legal implications. Inform. Syst. Front. **13**(2), 221–235 (2011)
5. Curien, N., Fauchart, E., Laffond, G., Moreau, F.: Online Consumer Communities: Escaping the Tragedy of the Digital Commons. NA, New York (2006)
6. Dellarocas, C.: Strategic manipulation of internet opinion forums: implications for consumers and firms. Manag. Sci. **52**(10), 1577–1593 (2006)
7. Feuerbach, J., Loepp, B., Barbu, C.M., Ziegler, J.: Enhancing an interactive recommendation system with review-based information filtering. IntRS **17**, 2–9 (2017)
8. Fuhr, N., Rölleke, T.: HySpirit — a probabilistic inference engine for hypermedia retrieval in large databases. In: Schek, H.-J., Alonso, G., Saltor, F., Ramos, I. (eds.) EDBT 1998. LNCS, vol. 1377, pp. 24–38. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0100975

9. Haydar, C., Boyer, A.: A new statistical density clustering algorithm based on mutual vote and subjective logic applied to recommender systems. In: Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, pp. 59–66 (2017)
10. Hu, S., Kumar, A., Al-Turjman, F., Gupta, S., Seth, S., et al.: Reviewer credibility and sentiment analysis based user profile modelling for online product recommendation. IEEE Access **8**, 26172–26189 (2020)
11. Jiménez, F.R., Mendoza, N.A.: Too popular to ignore: the influence of online reviews on purchase intentions of search and experience products. J. Interact. Mark. **27**(3), 226–235 (2013)
12. Jøsang, A.: The consensus operator for combining beliefs. Artif. Intell. **141**(1–2), 157–170 (2002)
13. Jøsang, A.: Subjective Logic. Springer, Berlin (2016)
14. Jøsang, A., Hankin, R.: Interpretation and fusion of hyper opinions in subjective logic. In: 2012 15th International Conference on Information Fusion, pp. 1225–1232. IEEE (2012)
15. Josang, A., Hayward, R.F., Pope, S.: Trust network analysis with subjective logic (2006)
16. Kong, R., Wang, Y., Xin, W., Yang, T., Hu, J., Chen, Z.: Customer reviews for individual product feature-based ranking. In: 2011 First International Conference on Instrumentation, Measurement, Computer, Communication and Control, pp. 449–453. IEEE (2011)
17. Krishnamoorthy, S.: Linguistic features for review helpfulness prediction. Expert Syst. Appl. **42**(7), 3751–3759 (2015)
18. Liu, Y., Bi, J.W., Fan, Z.P.: Ranking products through online reviews: a method based on sentiment analysis technique and intuitionistic fuzzy set theory. Inform. Fusion **36**, 149–161 (2017)
19. Park, D.H., Lee, J., Han, I.: The effect of on-line consumer reviews on consumer purchasing intention: the moderating role of involvement. Int. J. Electron. Commer. **11**(4), 125–148 (2007)
20. Rölleke, T., Fuhr, N.: Retrieval of complex objects using a four-valued logic. In: SIGIR, pp. 206–214. ACM (1996)
21. Sorensen, A.T., Rasmussen, S.J.: Is any publicity good publicity? a note on the impact of book reviews. NBER Working paper, Stanford University (2004)
22. Sun, M.: How does the variance of product ratings matter? Manag. Sci. **58**(4), 696–707 (2012)
23. Van Der Heijden, R.W., Kopp, H., Kargl, F.: Multi-source fusion operations in subjective logic. In: 2018 21st International Conference on Information Fusion (FUSION), pp. 1990–1997. IEEE (2018)
24. Viviani, M., Pasi, G.: Credibility in social media: opinions, news, and health information-a survey. Wiley Interdisc. Rev. Data Min. Knowl. Discovery **7**(5), e1209 (2017)
25. Zhang, K., Cheng, Y., Liao, W.k., Choudhary, A.: Mining millions of reviews: a technique to rank products based on importance of reviews. In: Proceedings of the 13th International Conference on Electronic Commerce, p. 12. ACM (2011)
26. Zhang, K., Narayanan, R., Choudhary, A.N.: Voice of the customers: mining online customer reviews for product feature-based ranking. WOSN **10**, 11–11 (2010)

On the Instability of Diminishing Return IR Measures

Tetsuya Sakai(✉)

Waseda University, Tokyo, Japan
tetsuyasakai@acm.org

Abstract. The diminishing return property of ERR (Expected Reciprocal Rank) is highly intuitive and attractive: its user model says, for example, that after the users have found a highly relevant document at rank r, few of them will continue to examine rank $(r+1)$ and beyond. Recently, another IR evaluation measure based on diminishing return called iRBU (intentwise Rank-Biased Utility) was proposed, and it was reported that nDCG (normalised Discounted Cumulative Gain) and iRBU align surprisingly well with users' SERP (Search Engine Result Page) preferences. The present study conducts offline evaluations of diminishing return measures including ERR and iRBU along with other popular measures such as nDCG, using four test collections and the associated runs from recent TREC tracks and NTCIR tasks. Our results show that the diminishing return measures generally underperform other graded relevance measures in terms of system ranking consistency across two disjoint topic sets as well as discriminative power. The results generalise a previous finding on ERR regarding its limited discriminative power, showing that the diminishing return user model hurts the stability of evaluation measures regardless of the utility function part of the measure. Hence, while we do recommend iRBU along with nDCG for evaluating adhoc IR systems from multiple user-oriented angles, iRBU should be used under the awareness that it can be much less statistically stable than nDCG.

Keywords: Diminishing return · Discriminative power · Evaluation measures · Statistical significance · System ranking consistency

1 Introduction

IR researchers use IR evaluation measures in their offline (i.e., test collection-based) experiments in the hope of improving their systems for real users. The measures should therefore (a) serve as surrogates of users' perspectives so that IR systems evolve towards the right direction; and (b) be statistically stable so that reliable offline experiments can be conducted. Regarding (a), Sakai and Zeng [36–38] recently reported that nDCG (normalised Discounted Cumulative Gain) [18] and their new measure called iRBU (intentwise Rank-Biased Utility) were the two best measures among the ones they examined from the viewpoint of how they align with users' SERP (Search Engine Result Page) preferences.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 572–586, 2021.
https://doi.org/10.1007/978-3-030-72113-8_38

iRBU is a component of a recently-proposed diversity evaluation measure called RBU (Rank-Biased Utility) [3], and inherits a highly intuitive and unique feature of ERR (Expected Reciprocal Rank) [9], namely, the *diminishing return* property [9,29].

As discussed in Chapelle *et al.* [9], ERR can be regarded as an instance of the NCU (Normalised Cumulative Utility) family of measures [33].[1] NCU defines each evaluation measure instance by specifying the user's *abandoning probability distribution* over the ranked documents and the *utility function* for each group of users who abandon the ranked list at a given rank. NCU thus represents the expected utility over all users. ERR and iRBU use the diminishing return probability distribution, while other instances of NCU such as AP (Average Precision) [7] and RBP (Rank-Biased Precision) [25] use distributions that do not consider diminishing return (See Sect. 3).

We examine the diminishing return measures along with non-diminishing-return measures primarily from the viewpoint of (b) mentioned above. More specifically, we compare the measures in terms of *system ranking consistency* when two disjoint topic sets are used for computing the mean scores, as well as discriminative power [28,29]. In addition, we compare the system rankings for every pair of measures. As IR researchers are primarily interested in evaluating and advancing the state-of-the-art, we use four recently-constructed test collections and their associated runs from TREC and NTCIR in our experiments.

2 Related Work

As we have discussed in Sect. 1, we want IR evaluation measures to (a) align well with user perception or performance; and (b) behave reliably in offline experiments. Concerning (a), prior art includes the studies by Turpin and Scholer [43], Al-Maskari *et al.* [1], Sanderson *et al.* [39], and Sakai and Zeng [36–38]. Among them, the most recent work of Sakai and Zeng [36–38] reported that nDCG and iRBU were the top two measures in terms of how often they align with users' SERP preferences, and this is what motivated the present study: *how do iRBU (and other diminishing return measures) perform from Viewpoint (b)?* While Sakai and Zeng [37] reported that nDCG and iRBU have similar discriminative power, we view that particular result as preliminary, as it relied on one relatively small data set, namely, the NTCIR-9 INTENT data from 2011 [34], with 43 topics and only 15 runs. Moreover, as the INTENT test collection was originally constructed for diversified search, it was not clear how these measures perform with modern *adhoc* IR test collections.

Among the methods for evaluating the reliability of IR evaluation measures in offline experiments, *discriminative power* [29] is probably the most widely-used today (e.g., Anelli *et al.* [4], Ashkan and Metzler [5], Chuklin *et al.* [10], Clarke *et al.* [11,12], Dou *et al.* [15], Golbus *et al.* [17], Kanoulas and Aslam [19], Leelanupab *et al.* [20], Lu *et al.* [21], Luo *et al.* [22], Robertson *et al.* [26], Smucker and Clarke [42], Valcarce *et al.* [45], Wang *et al.* [49], Zhou *et al.* [52]). Given a set

[1] Section 2 discusses an alternative framework for defining a family of measures [24].

S of runs ($|S| = K$), this method obtains a p-value for every system pair. Sakai's original method [28] repeated $K(K-1)/2$ bootstrap tests without correction (since correction is applied uniformly to all evaluation measures and therefore it would not affect the relative comparison of measures), and others have repeated t-tests in a similar manner. However, a multiple comparison procedure such as Tukey's HSD test is generally recommended for significance testing of multiple systems [30]. In the present study, we employ a distribution-free, randomised version of Tukey's HSD test [8,30] to discuss discriminative power.

Using TREC 2003–2004 robust track data and NTCIR-7 crosslingual task data from 2008, Sakai and Kando [32] showed that RBP substantially underperforms "deep" measures such as nDCG, Q, and AP when the measurement depth is 1,000. This was the *de facto* standard document cutoff for adhoc IR in early TRECs. However, recent IR tasks have shifted the focus towards smaller document cutoffs, as many researchers are interested in the search engine quality "above the fold." In this context where recall is not a central question, RBP is expected to perform better, as it is a purely precision-oriented measure. The present study re-examines IR evaluation measures including RBP, using four modern test collections and runs with small cutoffs, namely, 10 and 20.[2]

Before discriminative power was popularised, the *swap method* was often used for similar purposes. Zobel [53] split the original topic set in half and examined whether a conclusion regarding the comparison of two systems using the first subset can be confirmed on the other subset. However, his primary concern was the reliability of different significance tests. Voorhees and Buckley [48] considered repeated trials for breaking the topic set in half for the purpose of empirically determining reliable topic set sizes, but did not consider statistical significance testing.[3] Follow up studies on this swap method include Sanderson and Zobel [40] and Voorhees [47]; they considered statistical significance along with repeated topic set splits. On the other hand, Voorhees [46] used Kendall's τ to compare system rankings produced by two different versions of qrels (i.e., relevance assessments) to discuss test collection reliability. Since τ is a measure of system swaps across the entire ranking, this approach can naturally be combined with the aforementioned idea of repeatedly and randomly splitting the topic set in half. We thus examine *system ranking consistency*: if an evaluation measure produces a system ranking based on topic set A and another based on topic set B, how similar are the two rankings? While this is not a new idea,[4] the present study describes a simple procedure to systematically address system ranking consistency with multiple topic set splits, complete with distribution-free statistical significance testing for the difference in mean τ's (See Sect. 6).

[2] For example, the TREC 2014 Web Track used 20 as the document cutoff [13]; the NTCIR We Want Web tasks haved used 10 [23].

[3] Topic set sizes can also be theoretically determined based on statistical power, given some pilot data for variance estimation [30].

[4] For example, Amigó *et al.* [2] refer to the correlation of system rankings across data sets as *robustness*.

Another potentially useful approach to evaluating evaluation measures is to examine what mathematical axioms the measures satisfy [3,24]. This is beyond the scope of the present study. Evaluating evaluation measures is also closely related to evaluating test collection reliability [16,44].

ERR and iRBU are not the only measures that features the diminishing return property. Sakai [29] points out that Time-Biased Gain [42] and U-measure [31] also embody diminishing return. A few *adaptive* measures defined in the *C/W/L framework* [24], namely, INST (INSQ with T) [24], IFT (Information Foraging Theory measure) [6], BPM (Bejeweled Player Model measure) [50,51] can also accommodate diminishing return.[5] These measures generally require tuning of multiple parameters using document statistics, user behaviour data, and/or user satisfaction ratings, and are outside the scope of the present study.

3 Measures

Table 1. Seven ranked-retrieval measures considered in this study.

Measure	$P_A(r)$	$U(r)$	Parameters
AP (Average Precision) [7]	$P_{uniform}(r)$	$prec$(r)	–
Q (Q-measure) [27]	$P_{uniform}(r)$	$BR(r)$	–
RBP (Rank-Biased Precision) [25]	$(1-p)p^{r-1}$	$g(r)/2^{x_{max}}$	$p = 0.85$
ERR (Expected Reciprocal Rank) [9]	$P_{ERR}(r)$	$1/r$	–
EBR (Expected Blended Ratio) [36]	$P_{ERR}(r)$	$BR(r)$	–
iRBU (intentwise Rank-Biased Utility) [36]	$P_{ERR}(r)$	p^r	$p = 0.99$
nDCG (normalised Discounted Cumulative Gain) [18]	–	–	–

Table 1 provides a summary of the seven ranked retrieval measures examined in the present study. We selected these measures because (a) how they align with users' SERP preferences have been clarified by Sakai and Zeng [36–38]; (b) we wanted to compare diminishing return measures (indicated in the table and hereafter in **bold**) with non-diminishing-return measures in offline experiments; and (c) all of these measures are available in a single publicly-available evaluation toolkit, namely, NTCIREVAL.[6]

With the exception of nDCG, the measures shown in Table 1 are instances of NCU [33], given by $NCU = \sum_{r=1}^{L} P_A(r)U(r)$. Here, $P_A(r)$ is the probability that a user group abandons the SERP at rank r, and $U(r)$ is the utility of the

[5] Not all adaptive measures are diminishing return measures. Moffat *et al.* [24] classify Reciprocal Rank (RR) as adaptive, but RR does not accommodate diminishing return: once a relevant document is found, there is no further return.

[6] http://research.nii.ac.jp/ntcir/tools/ntcireval-en.html (version 200626).

top r documents of the SERP for that user group. L is the document cutoff: we consider $L = 10, 20$ in the present study as we have mentioned earlier.[7] Let x be a relevance level, and let x_{max} be the highest relevance level for an entire test collection. AP and Q use a uniform user distribution over all relevant documents, $P_A(r) = P_{uniform}(r) = I(r)/\min(L, R)$, where $I(r)$ is 0 if the document at rank r is nonrelevant (i.e., $x = 0$) and 1 otherwise (i.e., $0 < x \leq x_{max}$). The min operator ensures that the maximum attainable value of AP (Q) is 1 even for small document cutoffs [29].

Let $g(r)$ denote the gain value of the document at rank r. Following popular practice, we let $g(r) = 2^x - 1$ if the document at r is x-relevant for all graded relevance measures. **ERR**, **EBR**, and **iRBU** use the same diminishing return distribution [9]:

$$P_{ERR}(r) = P_{sat}(r) \prod_{k=1}^{r-1} (1 - P_{sat}(k)) \ , \tag{1}$$

where $P_{sat}(r) = g(r)/2^{x_{max}}$. It is clear from Eq. 1 that diminishing return measures would be "shallow" if there is a highly relevant document near the top of the SERP: it is assumed that very few users will go beyond that rank. In such a case, diminishing return measures will pay little attention to lower ranks, and may produce statistically unstable scores. As for RBP, its distribution assumes that the probability of the users transitioning from rank r to $(r+1)$ is a constant, namely, p. We consider $p = 0.85$ in the present study, as this setting aligned well with users' SERP preferences in the experiment of Sakai and Zeng [36–38]. This particular choice of p originates from the work of Moffat *et al.* [24].

Regarding the utility functions $U(r)$ shown in Table 1, AP uses *precision*, while Q and **EBR** use the *blended ratio* [29], which combines precision with the idea of *normalised cumulative gain* [18]. RBP uses $U(r) = g(r)/2^{x_{max}}$ ($= P_{sat}(r)$ in Eq. 1). Note that **ERR** uses $U(r) = 1/r$, which is equivalent to "precision at r when only the document at r is considered relevant." Finally, **iRBU** uses $U(r) = p^r$, where $p = 0.99$ throughout our experiments based on the user-based results of Sakai and Zeng [36–38]. Note that this utility function completely disregards the relevance of the top r documents: it only reflects the user effort spent so far in viewing the documents.

4 Data

Table 2 presents an overview of the data that we used in our experiments, with a short name we gave to each data set for brevity. These four data sets were chosen based on the following criteria: (I) they should be recent, since IR researchers are often interested in evaluating and advancing the state-of-the-art; (II) neither the number of topics nor the number of submitted runs should be small, since we want to obtain reliable experimental results; (III) they have graded relevance

[7] The relevance assessments of the four test collections we use in our experiments are expected to be incomplete: see the "rel. per topic" column in Table 2. Hence, using a large cutoff L probably would not give us reliable results.

Table 2. Overview of the data sets used in our experiments.

Our short name	Track/Task	#Topics	rel. levels	#rel. per topic	#runs used	(all runs)
TR19DL	TREC 2019 Deep Learning Track document retrieval task	43	4	153.4	37	(38)
TR18Core	TREC 2018 Common Core Track	50	3	79.0	63	(72)
STC2	NTCIR-13 (2017) Short Text Conversation 2 Chinese subtask	100	7	256.7	99	(120)
WWW3	NTCIR-15 (2020) We Want Web with CENTRE, English subtask	160	4	159.0	36	(37)

assessments, since modern evaluation measures accommodate graded relevance; and (IV) they should be diverse, representing different information access tasks and evaluation venues, since we want our results to generalise. The target corpus for TR19DL is an MS MARCO corpus (3.2 million documents) [14]; that for TR18Core is the TREC Washington Post Corpus (608,180 documents).[8] STC2 is not based on a traditional adhoc IR task: the task was to either retrieve or generate appropriate responses for a given Weibo post (i.e., a Chinese "tweet"). However, the returned responses were judged through the traditional pooling approach, and the systems were evaluated by ranked retrieval measures [41]. WWW3 is from the NTCIR-15 WWW-3 English subtask whose target corpus is clueweb12-B13 (about 50 million web pages) [35].[9]

The rightmost column of Table 2 shows the number of runs used in our experiments. From TR19DL, we excluded the worst performing run as this was a clear outlier: its mean AP (at $L = 10$) was below 0.1 whereas the second worst performer achieved over 0.4. Similarly, for TR18Core and STC2, we excluded 9 runs and 21 runs, respectively, whose mean AP (at $L = 10$) scores were below 0.1. As for WWW3, although all the runs achieved relatively high scores, we excluded one run as this was found to have a problem: two different systems were used to produce this single run.[10]

5 System Ranking Similarity

Table 3 shows how the system rankings according to different evaluation measures resemble one another in terms of Kendall' τ. Correlation strengths are visualised in colour ($\tau \geq 0.8$, $0.7 \leq \tau < 0.8$, $\tau < 0.7$). 95%CIs are omitted in the table due to lack of space, but are shown in brackets with the τ values mentioned below. The general trends are consistent across the four data sets, and they are

[8] https://trec-core.github.io/2018/.

[9] https://lemurproject.org/clueweb12/.

[10] The search results for the first 80 topics (i.e., the reused WWW-2 topics) were copied from a run from the NTCIR-14 WWW-2 task [23] and the other 80 topics (i.e., the new WWW-3 test topics) were processed by a new system.

Table 3. System ranking similarity (τ) for every pair of measures.

(I) cutoff $L = 10$							(II) cutoff $L = 20$						
(a) TR19DL (37 runs)													
	Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**		Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**
AP	0.877	0.754	0.736	0.613	0.628	0.631	AP	0.928	0.790	0.742	0.495	0.529	0.538
Q	–	0.763	0.751	0.640	0.649	0.598	Q	–	0.826	0.772	0.538	0.571	0.550
nDCG	–	–	0.892	0.823	0.844	0.763	nDCG	–	–	0.874	0.658	0.685	0.700
RBP	–	–	–	0.817	0.826	0.727	RBP	–	–	–	0.718	0.745	0.652
ERR	–	–	–	–	0.937	0.748	**ERR**	–	–	–	–	0.943	0.628
EBR	–	–	–	–	–	0.775	**EBR**	–	–	–	–	–	0.661
(b) TR18Core (63 runs)													
	Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**		Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**
AP	0.953	0.888	0.906	0.749	0.751	0.611	AP	0.963	0.900	0.917	0.735	0.758	0.635
Q	–	0.899	0.920	0.751	0.757	0.611	Q	–	0.908	0.925	0.739	0.764	0.631
nDCG	–	–	0.944	0.836	0.842	0.698	nDCG	–	–	0.931	0.794	0.818	0.703
RBP	–	–	–	0.806	0.802	0.666	RBP	–	–	–	0.783	0.804	0.665
ERR	–	–	–	–	0.951	0.780	**ERR**	–	–	–	–	0.950	0.757
EBR	–	–	–	–	–	0.790	**EBR**	–	–	–	–	–	0.762
(c) STC2 (99 runs)													
	Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**		Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**
AP	0.831	0.802	0.803	0.739	0.721	0.746	AP	0.833	0.795	0.804	0.739	0.722	0.746
Q	–	0.941	0.931	0.870	0.857	0.777	Q	–	0.936	0.931	0.871	0.857	0.778
nDCG	–	–	0.975	0.890	0.885	0.821	nDCG	–	–	0.965	0.894	0.885	0.820
RBP	–	–	–	0.877	0.877	0.825	RBP	–	–	–	0.877	0.877	0.825
ERR	–	–	–	–	0.947	0.788	**ERR**	–	–	–	–	0.947	0.788
EBR	–	–	–	–	–	0.786	**EBR**	–	–	–	–	–	0.786
(d) WWW3 (36 runs)													
	Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**		Q	nDCG	RBP	**ERR**	**EBR**	**iRBU**
AP	0.895	0.879	0.883	0.841	0.854	0.873	AP	0.908	0.860	0.810	0.746	0.765	0.806
Q	–	0.965	0.975	0.927	0.940	0.914	Q	–	0.946	0.902	0.838	0.857	0.867
nDCG	–	–	0.978	0.949	0.962	0.937	nDCG	–	–	0.943	0.879	0.898	0.895
RBP	–	–	–	0.933	0.952	0.927	RBP	–	–	–	0.917	0.937	0.857
ERR	–	–	–	–	0.981	0.905	**ERR**	–	–	–	–	0.975	0.832
EBR	–	–	–	–	–	0.917	**EBR**	–	–	–	–	–	0.844

particularly clear with the TREC data (Parts (a) and (b)). More specifically, the following can be observed in terms of system ranking similarity.

- AP, Q, nDCG, and RBP are similar to each other (e.g., $\tau = 0.742$ [0.623, 0.828], ..., 0.928 [0.889, 0.954] in Part (II)(a)), and they are less similar to the three diminishing return measures (e.g., $\tau = 0.495$ [0.307. 0.646], ..., 0.745 [0.627, 0.830] in Part (II)(a)).
- Within the diminishing return measures, while **ERR** and **EBR** are extremely similar to each other (e.g., $\tau = 0.943$ [0.912, 0.963] in Part (II)(a)), **iRBU** behaves a little differently from these two (e.g., $\tau = 0.628$ [0.472, 0.746], 0.661 [0.515, 0.770] in Part (II)(a)). In fact, in several cases (Parts (II)(a) (I)(d), and (II)(d)), **iRBU** behaves more similarly to nDCG than to any other measure.

The above observations on system ranking similarities are generally in line with the user-based results of Sakai and Zeng [36–38]: they reported that nDCG,

iRBU, RBP (with $p = 0.85$), and Q were the best measures in terms of agreement with users' SERP preferences on the NTCIR-9 INTENT data, in this exact order.

6 System Ranking Consistency Across Two Topic Sets

This section compares the seven ranked retrieval measures in terms of system ranking consistency across two disjoint topic sets. To be more specific, given a test collection whose topic set is T and a set of K runs associated with it, we compare a set $\{M\}$ of candidate evaluation measures as follows.

1. For each measure M, evaluate the K runs with T, and thereby obtain a $|T| \times K$ topic-by-run score matrix S_M.
2. From each S_M, obtain a τ score B times using the algorithm shown in Fig. 1 (or alternatively Fig. 2 for considering a smaller topic subset size $n < |T|/2$), where each τ quantifies the system ranking consistency when the K runs are ranked according to two disjoint subsets of T. We thus obtain a $B \times |\{M\}|$ matrix C containing the consistency τ scores.
3. To see if any of the differences in mean consistency τ scores are statistically significant, apply a paired, randomised Tukey HSD test [8,30] to C.

```
$n_1 = \texttt{truncate}(|T|/2)$;   $n_2 = |T| - n_1$;
for $b = 1$ to $B$; do
    $T_1^b$ = a random subset of the original topic set $T$ s.t. $|T_1| = n_1$;
    $T_2^b = T - T_1$; /* $|T_2| = n_2$ */
    $r_1^b$ = run ranking according to mean $M$ over $T_1^b$;
    $r_2^b$ = run ranking according to mean $M$ over $T_2^b$;
    $\tau^b$ = Kendall's $\tau$ score for run rankings $r_1^b$ and $r_2^b$;
done
```

Fig. 1. Pseudocode for sampling a consistency τ score B times for an evaluation measure M, given a set of runs for Topic Set T. The function `truncate` returns the integer part of an argument.

```
for $b = 1$ to $B$; do
    $T_1^b$ = a random subset of the original topic set $T$ s.t. $|T_1| = n$;
    $T_2^b$= a random subset of $T - T_1$ s.t. $|T_2| = n$;
    $r_1^b$ = run ranking according to mean $M$ over $T_1^b$;
    $r_2^b$ = run ranking according to mean $M$ over $T_2^b$;
    $\tau^b$ = Kendall's $\tau$ score for run rankings $r_1^b$ and $r_2^b$;
done
```

Fig. 2. A variant of Fig. 1 that uses a specified sample size n ($< |T|/2$) for both topic subsets.

Our method enables us to argue, for example, "measure M_1 statistically significantly outperforms M_2 ($p \approx 0.000$) in terms of mean system ranking consistency," and to discuss effect sizes (standardised mean differences) [30]. Note that the randomised Tukey HSD test is distribution-free: it can be applied regardless of what distribution the τ scores obey. Moreover, as this test is a multiple comparison procedure, we can ensure that the familywise Type I error rate is no more than α, which we set to 5% throughout our study. We use the `Random-test` script of the `Discpower` tool[11] for the randomised Tukey HSD test with 5,000 trials [30].

Table 4 summarises the results of our system ranking consistency experiments with $B = 1,000$ topic subset pairs in each case. "Full split" means, for example, 43 topics are split into 21 and 22 topics to measure the consistency (Fig. 1), while "10 vs. 10" means only 10 topics were used in both topic subsets (Fig. 2). As indicated in the table caption, randomised Tukey HSD test results are indicated with symbols, and effect sizes can be computed from the numbers presented here. For example, in Table 4(a)(I), RBP statistically significantly outperforms all other measures; the effect size for the difference between RBP and nDCG is $(0.635 - 0.598)/\sqrt{0.00245} = 0.748$. That is, the two measures are about 0.75 common standard deviations apart.

It can be observed that, with a few exceptions, the three diminishing return measures (indicated in **bold**) underperform the other measures. The results demonstrate that the diminishing return property hurts the statistical stability of the measures, regardless of the utility function $U(r)$ employed (See Table 1). On the other hand, the rankings within the diminishing return measures and those within the non-diminishing-return measures are not consistent across the data sets and cutoffs, suggesting that the differences are data-dependent, not inherent.

7 Discriminative Power

Finally, we evaluate our seven measures based on the widely-used discriminative power method [28,29], For significance testing, again we used the randomised version of the paired Tukey HSD test, using the `Discpower` tool with 5,000 trials (See Sect. 6). Figure 3 shows the results with cutoff $L = 20$; similar results with $L = 10$ are omitted due to lack of space. We can observe the following.

- The trends for the two TREC data sets are quite clear (Fig. 3(a) and (b)): the three diminishing return measures are substantially less discriminative than the other measures. The differences are more pronounced for TR19DL.
- The results with STC2 (Fig. 3(c)), which is not an adhoc IR data set, are different from the above in two aspects: first, **iRBU** does exceptionally well; second, the binary AP does equally well, which means that graded relevance is not essential for discriminating between the STC2 runs even though this particular data set has 7-point relevance levels (See Table 2).

[11] http://research.nii.ac.jp/ntcir/tools/discpower-en.html (version 160507).

Table 4. System ranking consistency in terms of mean τ over $B = 1,000$ trials. For each section, a paired randomised Tukey HSD test at the 5% significance level was conducted: ♠/♣/♡/◇/ ‡ /† means statistically significantly outperforms the worst 6/5/4/3/2/1 measure(s), respectively. V_{E2} is the residual variance computed from each 1000×7 matrix of τ scores, which can be used for computing effect sizes [30].

Cutoff $L = 10$				Cutoff $L = 20$			
(a) TREC19DL (43 topics)							
(I) full split		(II) 10 vs. 10		(III) full split		(IV) 10 vs. 10	
$V_{E2} = 0.00245$		$V_{E2} = 0.00579$		$V_{E2} = 0.00268$		$V_{E2} = 0.00681$	
RBP	0.635 ♠	RBP	0.530 ♠	nDCG	0.693 ♠	nDCG	0.559 ♡
nDCG	0.598 ♡	nDCG	0.513 ♣	Q	0.679 ♣	Q	0.553 ♡
iRBU	0.597 ♡	Q	0.486 ♡	AP	0.653 ◇	RBP	0.547 ♡
ERR	0.567 ‡	AP	0.452 ◇	RBP	0.652 ◇	AP	0.531 ◇
Q	0.561 †	**iRBU**	0.424 ‡	**iRBU**	0.639 ‡	**iRBU**	0.454 ‡
EBR	0.558 †	**EBR**	0.386	**ERR**	0.563 †	**ERR**	0.381
AP	0.541	**ERR**	0.386	**EBR**	0.550	**EBR**	0.375
(b) TREC18Core (50 topics)							
(I) full split		(II) 10 vs. 10		(III) full split		(IV) 10 vs. 10	
$V_{E2} = 0.00149$		$V_{E2} = 0.00328$		$V_{E2} = 0.00158$		$V_{E2} = 0.00369$	
Q	0.660 ♣	Q	0.547 ♣	Q	0.689 ♣	Q	0.579 ♣
AP	0.658 ♣	AP	0.543 ♣	AP	0.684 ♣	AP	0.578 ♣
RBP	0.626 ◇	RBP	0.521 ◇	nDCG	0.652 ◇	nDCG	0.558 ◇
nDCG	0.620 ◇	nDCG	0.517 ◇	RBP	0.645 ◇	RBP	0.549 ◇
EBR	0.586 †	**EBR**	0.438 †	**EBR**	0.588 †	**EBR**	0.444 †
ERR	0.584 †	**ERR**	0.432 †	**ERR**	0.584 †	**ERR**	0.433 †
iRBU	0.549	**iRBU**	0.386	**iRBU**	0.532	**iRBU**	0.375
(c) NTCIR13STC2 (100 topics)							
(I) full split		(II) 10 vs. 10		(III) full split		(IV) 10 vs. 10	
$V_{E2} = 0.000404$		$V_{E2} = 0.00132$		$V_{E2} = 0.000405$		$V_{E2} = 0.00132$	
AP	0.788 ♠	AP	0.598 ♠	AP	0.788 ♠	AP	0.598 ♠
Q	0.765 ♣	nDCG	0.569 ◇	Q	0.765 ♣	nDCG	0.572 ♡
nDCG	0.757 ◇	Q	0.567 ‡	nDCG	0.757 ◇	Q	0.567 ‡
RBP	0.753 ◇	RBP	0.563 ‡	RBP	0.753 ◇	RBP	0.563 ‡
iRBU	0.736 ‡	**iRBU**	0.562 ‡	**iRBU**	0.736 ‡	**iRBU**	0.562 ‡
ERR	0.726 †	**ERR**	0.529 †	**ERR**	0.726 †	**ERR**	0.529 †
EBR	0.697	**EBR**	0.487	**EBR**	0.697	**EBR**	0.487
(d) NTCIR15WWW3 (160 topics)							
(I) full split		(II) 10 vs. 10		(III) full split		(IV) 10 vs. 10	
$V_{E2} = 0.000597$		$V_{E2} = 0.00369$		$V_{E2} = 0.000707$		$V_{E2} = 0.00421$	
nDCG	0.851 ♣	RBP	0.566 ♡	RBP	0.866 ♣	Q	0.673 ♣
RBP	0.851 ♣	nDCG	0.564 ♡	Q	0.866 ♣	nDCG	0.624 ♣
Q	0.845 ♡	Q	0.561 ♡	nDCG	0.853 ♡	RBP	0.598 ♡
EBR	0.813 ‡	**EBR**	0.503 †	AP	0.836 ◇	AP	0.576 ◇
iRBU	0.813 ‡	**iRBU**	0.492 †	**EBR**	0.803 ‡	**EBR**	0.054 ‡
ERR	0.783 †	**ERR**	0.446 †	**ERR**	0.779 †	**iRBU**	0.446
AP	0.777	AP	0.416	**iRBU**	0.757	**ERR**	0.442

– The results with WWW3 (Fig. 3(d)) is similar to the TREC ones, except that AP performs very poorly. That is, in contrast to the results with STC2, graded relevance works very effectively to differentiate between the WWW3 runs.

The overall picture is that, with the exception of **iRBU** on STC2 (which is not an adhoc IR data set), diminishing return measures tend to suffer in terms of discriminative power. While it was already known that **ERR** has low discriminative power due to its diminishing return probability distribution $P_{ERR}(r)$ (Eq. 1) [29], our results generalises the observation for all three diminishing return measures on modern, diverse data sets. In other words, we have demonstrated that diminishing return measures may suffer in terms of discriminative power regardless of the utility function $U(r)$ employed (See Table 1).

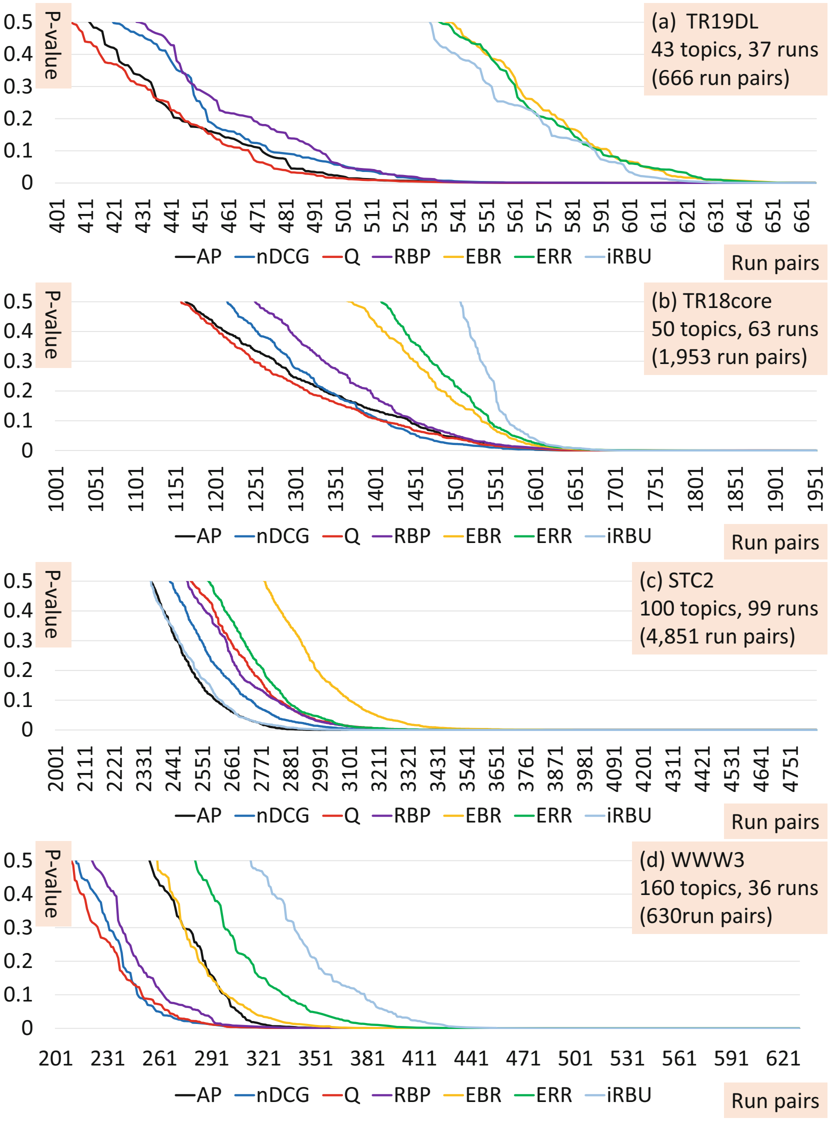


Fig. 3. Discriminative power curves of the seven measures (with cutoff $L = 20$) on the four data sets (Randomised Tukey HSD tests for paired data with 5,000 trials).

8 Conclusions

The present study compared some properties of three diminishing return measures (**ERR**, **EBR**, and **iRBU**) with other adhoc IR measures (nDCG, Q, RBP, and the binary AP), using diverse ranked retrieval data sets from TREC and NTCIR. **iRBU** was of particular interest, as Sakai and Zeng [36–38] have reported that it performed surprisingly well in terms of agreement with users' SERP preferences, along with nDCG. Our findings can be summarised as follows.

System Ranking Similarity Between Two Measures. The three diminishing return measures rank systems substantially differently compared to the other measures. However, while **ERR** and **EBR** behave very similarly, **iRBU** behaves more similarly to nDCG than to other measures with some data. These results are in line with the aforementioned results of Sakai and Zeng [36–38], who reported that nDCG, **iRBU**, RBP (with $p = 0.85$), and Q performed best in terms of agreement with users' SERP preferences, in this exact order.

System Ranking Consistency. With a few exceptions, the three diminishing return measures statistically significantly underperform the other measures in terms of system ranking consistency across two disjoint topic sets.

Discriminative Power. Similarly, the diminishing return measures generally perform poorly in terms of discriminative power. These results generalise a previous finding regarding the low discriminative power of **ERR** [29].

Both the consistency and discriminative power results demonstrate that the diminishing return property hurts the statistical stability of the measures, regardless of the utility function ($U(r)$) employed.

A practical recommendation for researchers working on adhoc IR would be to use both nDCG and **iRBU**, the top two measures in the user-based experiments of Sakai and Zeng [36–38], under the awareness that **iRBU** may be substantially less stable than nDCG due to its intuitive diminishing return model. Hence, a statistically significant difference in terms of nDCG may often not be significant in terms of **iRBU**. The NTCIR-15 WWW-3 task [35] has already used iRBU as an official evaluation measure in addition to nDCG, Q, and normalised ERR (nERR).

The fact that **iRBU** is statistically unstable despite its high agreement with users' SERP preferences serves as a reminder that evaluation measures should also be examined from multiple viewpoints. We argue that both offline evaluation approaches such as the ones we took in the present study and the user-based validations are necessary.

The present study did not aim to cover all existing diminishing-return IR measures for the reason discussed in Sect. 2. To help other researchers reproduce and extend our work, we have made all of our topic-by-run score matrices publicly available.[12]

[12] https://waseda.box.com/ECIR2021PACK.

References

1. Al-Maskari, A., Sanderson, M., Clough, P., Airio, E.: The good and the bad system: does the test collection predict users' effectiveness. In: Proceedings of ACM SIGIR 2018, pp. 59–66 (2008)
2. Amigó, E., Gonzalo, J., Mizzaro, S., de Albornoz, J.C.: An effectiveness metric for ordinal classification: formal properties and experimental results. In: Proceedings of ACL 2020 (2020)
3. Amigó, E., Spina, D., de Albornoz, J.C.: An axiomatic analysis of diversity evaluation metrics: introducting the rank-biased utility metric. In: Proceedings of ACM SIGIR 2018, pp. 625–634 (2018)
4. Anelli, V.W., Di Noia, T., Di Sciascio, E., Pomo, C., Ragone, A.: On the discriminative power of hyper-parameters in cross-validation and how to choose them. In: Proceedings of ACM RecSys 2019, pp. 447–451 (2019)
5. Ashkan, A., Metzler, D.: Revisiting online personal search metrics with the user in mind. In: Proceedings ACM SIGIR 2019, pp. 625–634 (2019)
6. Azzopardi, L., Thomas, P., Craswell, N.: Measuring the utility of search engine result pages. In: Proceedings of ACM SIGIR 2018, pp. 605–614 (2018)
7. Buckley, C., Voorhees, E.M.: Retrieval system evaluation. In: Voorhees, E.M., Harman, D.K. (eds.) TREC: Experiment and Evaluation in Information Retrieval, pp. 53–75. The MIT Press (2005)
8. Carterette, B.: Multiple testing in statistical analysis of systems-based information retrieval experiments. ACM TOIS **30**(1), 1–34 (2012)
9. Chapelle, O., Metzler, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: Proceedings of ACM CIKM 2009, pp. 621–630 (2009)
10. Chuklin, A., Serdyuov, P., de Rijke, M.: Click model-based information retrieval metrics. In: Proceedings of ACM SIGIR 2013, pp. 493–502 (2013)
11. Clarke, C.L., Craswell, N., Soboroff, I., Ashkan, A.: A comparative analysis of cascade measures for novelty and diversity. In: Proceedings of ACM WSDM 2011, pp. 75–84 (2011)
12. Clarke, C.L., Vtyurina, A., Smucker, M.D.: Offline evaluation without gain. In: Proceedings of ICTIR 2020, pp. 185–192 (2020)
13. Collins-Thompson, K., Macdonald, C., Bennett, P., Diaz, F., Voorhees, E.M.: TREC 2014 web track overview. In: Proceedings of TREC 2014 (2015)
14. Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Voorhees, E.M.: Overview of the TREC 2019 deep learning track. In: Proceedings of TREC 2019 (2020)
15. Dou, Z., Yang, X., Li, D., Wen, J.R., Sakai, T.: Low-cost, bottom-up measures for evaluating search result diversification. Inform. Retrieval J. **23**, 86–113 (2020)
16. Ferro, N., Kim, Y., Sanderson, M.: Using collection shards to study retrieval performance effect sizes. ACM TOIS **37**(3), 1–40 (2019)
17. Golbus, P.B., Aslam, J.A., Clarke, C.L.: Increasing evaluation sensitivity to diversity. Inform. Retrieval **16**, 530–555 (2013)
18. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inform. Syst. **20**(4), 422–446 (2002)
19. Kanoulas, E., Aslam, J.A.: Empirical justification of the gain and discount function for nDCG. In: Proceedings of ACM CIKM 2009, pp. 611–620 (2009)
20. Leelanupab, T., Zuccon, G., Jose, J.M.: A comprehensive analysis of parameter settings for novelty-biased cumulative gain. In: Proceedings of ACM CIKM 2012, pp. 1950–1954 (2012)

21. Lu, X., Moffat, A., Culpepper, J.S.: The effect of pooling and evaluation depth on IR metrics. Inform. Retrieval J. **19**(4), 416–445 (2016)
22. Luo, J., Wing, C., Yang, H., Hearst, M.A.: The water filling model and the cube test: multi-dimensional evaluation for professional search. In: Proceedings of ACM CIKM 2013, pp. 709–714 (2013)
23. Mao, J., Sakai, T., Luo, C., Xiao, P., Liu, Y., Dou, Z.: Overview of the NTCIR-14 we want web task. In: Proceedings of NTCIR-14, pp. 455–467 (2019). http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings14/pdf/ntcir/01-NTCIR14-OV-WWW-MaoJ.pdf
24. Moffat, A., Bailey, P., Scholer, F., Thomas, P.: Incorporating user expectations and behavior into the measurement of search effectiveness. ACM TOIS **35**(3), 1–38 (2017)
25. Moffat, A., Zobel, J.: Rank-biased precision for measurement of retrieval effectiveness. ACM TOIS **27**(1), 1–27 (2008)
26. Robertson, S.E., Kanoulas, E., Yilmaz, E.: Extending average precision to graded relevance judgements. In: Proceedings of ACM SIGIR 2010, pp. 603–610 (2010)
27. Sakai, T.: Ranking the NTCIR systems based on multigrade relevance. In: Myaeng, S.H., Zhou, M., Wong, K.-F., Zhang, H.-J. (eds.) AIRS 2004. LNCS, vol. 3411, pp. 251–262. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31871-2_22
28. Sakai, T.: Evaluating evaluation metrics based on the bootstrap. In: Proceedings of ACM SIGIR 2006, pp. 525–532 (2006)
29. Sakai, T.: Metrics, statistics, tests. In: Ferro, N. (ed.) PROMISE 2013. LNCS, vol. 8173, pp. 116–163. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54798-0_6
30. Sakai, T.: Laboratory Experiments in Information Retrieval. TIRS, vol. 40. Springer, Singapore (2018). https://doi.org/10.1007/978-981-13-1199-4
31. Sakai, T., Dou, Z.: Summaries, ranked retrieval and sessions: a unified framework for information access evaluation. In: Proceedings of ACM SIGIR 2013, pp. 473–482 (2013)
32. Sakai, T., Kando, N.: On information retrieval metrics designed for evaluation with incomplete relevance assessments. Inform. Retrieval **11**(5), 447–470 (2008)
33. Sakai, T., Robertson, S.: Modelling a user population for designing information retrieval metrics. In: Proceedings of EVIA 2008, pp. 30–41 (2008)
34. Sakai, T., Song, R.: Diversified search evaluation: lessons from the NTCIR-9 INTENT task. Inform. Retrieval **16**(4), 504–529 (2013)
35. Sakai, T., et al.: Overview of the NTCIR-15 we want web with CENTRE task. In: Proceedings of NTCIR-15, pp. 219–234 (2020)
36. Sakai, T., Zeng, Z.: Which diversity evaluation measures are "good"? In: Proceedings of ACM SIGIR 2019, pp. 595–604 (2019)
37. Sakai, T., Zeng, Z.: Good evaluation measures based on document preferences. In: Proceedings of ACM SIGIR 2020, pp. 359–368 (2020)
38. Sakai, T., Zeng, Z.: Retrieval evaluation measures that agree with users' serp preferences: traditional, preference-based, and diversity measures. ACM TOIS **39**(2), 1–35 (2020)
39. Sanderson, M., Paramita, M.L., Clough, P., Kanoulas, E.: Do user preferences and evaluation measures line up? In: Proceedings of ACM SIGIR 2010, pp. 555–562 (2010)
40. Sanderson, M., Zobel, J.: Information retrieval evaluation: effort, sensitivity, and reliability. In: Proceedings of ACM SIGIR 2005, pp. 162–169 (2005)

41. Shang, L., et al.: Overview of the NTCIR-13 short text conversation task. In: Proceedings of NTCIR-13, pp. 194–210 (2017)
42. Smucker, M.D., Clarke, C.L.: Time-based calibration of effectiveness measures. In: Proceedings of ACM SIGIR 2012, pp. 95–104 (2012)
43. Turpin, A., Scholer, F.: User performance versus precision measures for simple search tasks. In: Proceedings of ACM SIGIR 2006, pp. 11–18 (2006)
44. Urbano, J.: Test collection reliability: a study of bias and robustness to statistical assumptions via stochastic simulation. Inform. Retrieval J. **19**(3), 313–350 (2016)
45. Valcarce, D., Bellogín, A., Parapar, J., Castells, P.: Assessing ranking metrics in top-N recommendation. Inform. Retrieval J. **23**(4), 411–448 (2020). https://doi.org/10.1007/s10791-020-09377-x
46. Voorhees, E.M.: Variations in relevance judgments and the measurement of retrieval effectiveness. Inform. Process. Manag. **36**, 697–716 (2000)
47. Voorhees, E.M.: Topic set size redux. In: Proceedings of ACM SIGIR 2009, pp. 806–807 (2009)
48. Voorhees, E.M., Buckley, C.: The effect of topic set size on retrieval experiment error. In: Proceedings of ACM SIGIR 2002, pp. 316–323 (2002)
49. Wang, X., Wen, J.R., Dou, Z., Sakai, T., Zhang, R.: Search result diversity evaluation based on intent hierarchies. IEEE Trans. Knowl. Data Eng. **30**(1), 156–169 (2018)
50. Zhang, F., Liu, Y., Li, X., Zhang, M., Xu, Y., Ma, S.: Evaluating web search with a bejeweled player model. In: Proceedings of ACM SIGIR 2017, pp. 425–434 (2017)
51. Zhang, F., et al.: Models versus satisfaction: towards a better understanding of evaluation metrics. In: Proceedings of ACM SIGIR 2020, pp. 379–388 (2020)
52. Zhou, K., Lalmas, M., Sakai, T., Cummins, R., Jose, J.M.: On the reliability and intuitiveness of aggregated search metrics. In: Proceedings of ACM CIKM 2013, pp. 689–698 (2013)
53. Zobel, J.: How reliable are the results of large-scale information retrieval experiments? In: Proceedings of ACM SIGIR 1998, pp. 307–314 (1998)

Studying the Effectiveness of Conversational Search Refinement Through User Simulation

Alexandre Salle[1(✉)], Shervin Malmasi[2], Oleg Rokhlenko[2], and Eugene Agichtein[2,3]

[1] Institute of Informatics, Federal University of Rio Grande do Sul,
Porto Alegre, Brazil
alex@alexsalle.com

[2] Amazon, Seattle, WA, USA
{malmasi,olegro,eugeneag}@amazon.com

[3] Emory University, Atlanta, GA, USA

Abstract. A key application of conversational search is refining a user's search intent by asking a series of clarification questions, aiming to improve the relevance of search results. Training and evaluating such conversational systems currently requires human participation, making it unfeasible to examine a wide range of user behaviors. To support robust training/evaluation of such systems, we propose a simulation framework called CoSearcher (Information about code/resources available at https://github.com/alexandres/CoSearcher.) that includes a parameterized user simulator controlling key behavioral factors like cooperativeness and patience. Using a standard conversational query clarification benchmark, we experiment with a range of user behaviors, semantic policies, and dynamic facet generation. Our results quantify the effects of user behaviors, and identify critical conditions required for conversational search refinement to be effective.

Keywords: Conversational search · User simulation for conversational search · Conversational query clarification

1 Introduction

As personalized information agents become ubiquitous, people increasingly expect to engage them in information-seeking dialogues, instead of having to formulate a precise query. A user's query to a search system often under-specifies the search intent (or *facet* of the information need, as is often referred to in the literature). A conversational system could elicit a more precise information need from a user, by asking her a series of *clarification questions* to narrow down the set of possible intents, ultimately to improve the relevance of the search results.

A. Salle–Work conducted during an internship at Amazon, Seattle, WA, USA.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 587–602, 2021.
https://doi.org/10.1007/978-3-030-72113-8_39

Recent work [7] has shown the theoretical value of obtaining answers to such clarification questions to improve the final retrieval.

Search refinement is also critical in practice, namely for voice-based agents like Alexa or Siri. Generally, only a small number of results can be returned to the user via a voice modality, and matching the correct search intent is critical [31]. Furthermore, in applications such as e-commerce, successive search refinement is natural for narrowing down the choice of products using facets of the target item.

Unfortunately, conversational search refinement is highly challenging due to the reliance on human participation for developing, training, and evaluating system variants or parameters. Furthermore, some users may not be willing to provide additional information to the search system after the initial request, while others might be willing to collaborate with the system by engaging in a dialogue. To address these issues, training and evaluating such conversational systems with a large number of users or crowd workers has been the dominant strategy. This has two shortcomings: (1) High cost, especially when different variations of a search system must be tested; (2) The pool of human participants might not be representative of future participants, who might, for example, be less *cooperative* and/or *patient*. A key contribution of this paper is re-examining the underlying assumptions of conversational search, to quantify the effects of user *cooperativeness*, i.e., willingness to provide clarification information, and user *patience*, i.e., willingness to engage in a long dialogue with a search system. We quantify this intuition by developing a simple, yet powerful, stochastic user simulator CoSearcher for conversational search refinement, and investigate the implications of cooperativeness and patience of users by extensive simulation experiments that would not be feasible with human participants. This proposed simulator provides a way to better understand the effectiveness and limitations of the a given conversational search system, for a wider range of potential future users, without degrading their search experience.

Although our user simulator has only two parameters (cooperativeness and patience), and might thus be deemed *unrealistically simple* because humans have far more "variables", we argue that these are the characteristics directly responsible for the user behavior *observable* by a search system, and thus form an acceptable proxy for scalable evaluation of a conversational search system under a wide range of *realistic* configurations of complex latent search behavior "variables".

In summary, our contributions include:

- We systematically investigate the task of conversational search intent clarification, comparing facet identification and ranking methods, for both static and dynamically generated candidate intents.
- We present a simple yet powerful conversational search simulator, CoSearcher, with key parameters of cooperativeness and patience, to enable systematic and scalable experimentation with conversational search refinement (Sect. 3.4).

- Using CoSearcher, we for the first time demonstrate using extensive simulation experiments, that modeling cooperation and patience of the searcher is fundamental for the success of conversational search, and identify the conditions where conversational search can be effective. This required evaluating results for hundreds of thousands parameter combinations for conversational experiments, which would not be feasible with human participants. (Sect. 5).

Broadly, our work adds to the growing evidence of the importance of engaging in conversations with users to improve search performance, and provides the critical building block, the CoSearcher user simulator, for scalable evaluation of a given conversational search system under a variety of conditions. Next, we briefly review related work to place our contributions in context.

2 Related Work

There is a large body of work in NLP and IR that addresses conversational systems [8,14,34,37]. Advances in NLP and IR in the last few years have also been accompanied by a surge in research of conversational systems.

Within the sub-field, understanding user behavior is an important research direction. [31] and [21] performed user studies to understand what kind of user behavior is useful for conversational search, but they did not explicitly model the results for use in simulations. Additionally, [30,39] perform user simulation, but unlike our work focus solely on recommender systems and use a fixed user model. For chat systems and task-completion dialogues, developing user simulators has also been shown to be an effective way to reduce the required training data [11,16,24], which inspired our efforts to adapt that general idea to search-oriented conversational systems. To the best of our knowledge, our paper is the first to propose a user simulator for *conversational search*.

A parallel line of work focuses on learning to ask clarification questions to fill in missing information [25–28,38]. None of these, however, focus on intent refinement, nor do they make use of a variable user model for evaluation. Another related direction is faceted search, where a user reacts to the proposed facets to refine the information need or to restrict or change the set of results [17–19,22,23,32,33,36].

Most similar to our work is that of [7], which uses human annotation of clarification questions which are then used within an IR system to evaluate how they could help retrieval performance. They release the resulting dataset, called Qulac, which we use as the basis of our paper. Qulac makes use of the 198 topics, corresponding facets and relevance judgements from the TREC09-12 diversity track [12,13], supplemented by crowdsourced human clarification questions and answers for each facet. For each topic, there are multiple human generated clarification questions corresponding to the each of the topic's facets, and for each *(topic,facet,question)* triple, there is an human generated answer where the human assumes the role of a searcher looking for the facet and answers the given question. Very recently, the Qulac dataset was expanded into ClariQ

[6] via the addition of new data, including synthetic multi-turn conversations. Our work is evaluated using the original Qulac dataset which is sufficient to investigate the research questions posed here. Our other, expanded facet dataset constructed from Bing query suggestions and manual annotations, complements Qulac and allows us to investigate additional challenges that arise with numerous query facets.

The Qulac paper [7] presents the Neural Question Selection (NeuQS) model, which given a conversation context (a series of questions/answers), selects the next question to ask from a candidate question database (the Qulac dataset). The human answer is then used to simulate the end of the conversation and the whole conversation is used as input to a query-likelihood IR system to evaluate the utility of the clarification question.

We differ from this work by focusing on intent refinement—the goal of our system is to narrow a set of candidate intents down to a specific intent—and by creating a user model and simulator, CoSearcher, which allows us to evaluate the utility of clarification questions not just on a specific set of human annotators, but rather a large set of simulated parameterized users. CoSearcher also enables the possibility of scalable training of conversational search systems, optimized for different types of users, and supporting sophisticated, yet data hungry, end-to-end deep learning approaches for conversational search, e.g., via Reinforcement Learning [9,35].

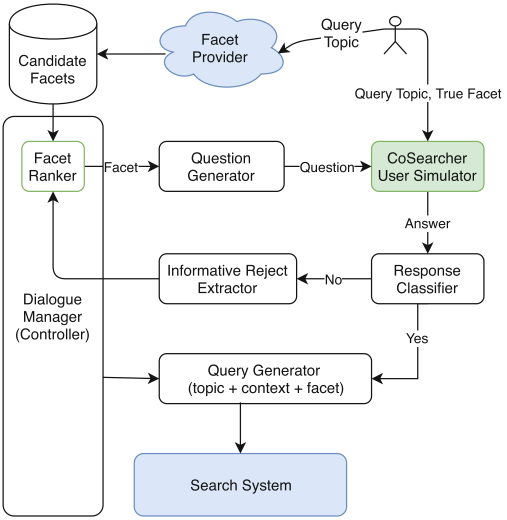
3 Modeling Conversational Search Intent Refinement Through User Simulation

We now overview the conversational search intent refinement setting, following the recent formulation in [7], and our simulation-based approach for investigating this topic.

3.1 Problem Setting: Conversational Search Refinement

Often, a searcher (user) provides an under-specified query to the search system, which may reflect multiple information needs, or different facets of the same intent. A conversational search refinement system attempts to pinpoint the user's search intent via a series of *clarification questions*, which the Searcher can *choose* to answer cooperatively (by volunteering additional information about their intent), lazily ("yes/no") or not respond to the system at all, e.g., if the Searcher ran out of time or patience. After each turn, the search system may chose to ask additional clarification questions, or return search results, or both. An example conversational search dialogue is shown in Fig. 3a, for the initial under-specified query, where the system follows with a sequence of clarification questions to generate the result ranking using the expanded/refined query.

Formally, we assume that the searcher has an information need (topic) t (i.e., the initial search query), and a true information need facet or aspect f_t, which the system has to infer to properly rank the search results. We also assume that



(a)

USER: Korean language *(the topic)*

BOT: **Are you looking to find resources on translating Korean to other languages?**

COSEARCHER: no

BOT: **Ok, are you looking to find resources on learning and teaching Korean?**

COSEARCHER: no

BOT: **Ok, are you looking to find a Korean/English bilingual dictionary?**

COSEARCHER: no i just want to know a few common korean phrases

BOT: **Are you looking for what are some common Korean phrases?**

COSEARCHER: YES

True intent: "What are some common Korean phrases?"

(b)

Fig. 1. (a) System overview, illustrating CoSearcher instantiated with (topic, intent facet), and a Facet Provider, which provides candidate facets that the search refinement system uses to converse with the CoSearcher to identify the intended facet; (b): An actual simulated conversation with a partially cooperative CoSearcher instance.

candidate facets C for the topic t is either known (e.g., from a knowledge base if the query is an entity), or can be dynamically generated (e.g., from query refinement logs of a search engine, or from popular entity attributes). The goal of the search system, then, is to identify the intended topic facet f_t by asking clarification questions, and return a list of results relevant to f_t. Specifically, the search system picks the first candidate facet $c \in C$ and asks a clarification question: "Are you looking for c?". The user can respond with either "Yes" or "No". If the answer is "Yes", the agent stops, accepting c as its best guess for the searcher's true information need. If the answer is "No", the agent selects the next candidate facet c from the list of candidate facets. If the user's "No" is *informative* (has additional information which might guide refinement, such as "No, I'm looking for...") we add the answer to the current context to be used for re-ranking. Candidates facets are then *re-ranked*, as described below, and this process repeated until either there are no more candidate facets or the user's patience runs out. Note that in our setup, we choose to model neutral responses (when the proposed facet is related to intended facet but not quite the same) as "No", since the intended facet has not yet been identified.

3.2 Candidate Facet Ranking Strategies

We consider two facet ranking strategies: **(1) Rand**: a random baseline that orders facets randomly. **(2) Sim**: a semantic similarity strategy, which assigns a score for every candidate facet by computing the mean cosine similarity between the facet and each informative "No" in the conversation context, using mean bag-of-vectors as sentence embeddings. We use the LexVec n-gram subword vectors [29] to represent each word.

3.3 Dynamic Facet Generation

The previous state of the art approach—NeuQS [7] and similar methods—require knowing *a priori* a set of candidate questions and answers for a given facet, which is not realistic for most search topics or information needs. We now investigate how to abstract and generalize this approach to *dynamically generated a set of candidate facets* using a *facet provider*.

One example of such a facet provider is a search engine query suggestion mechanism, e.g., the Bing search engine Autosuggest available via an API,[1] which, given an initial query, returns a set of 8 query completions. The query topic is used the initial query and the returned set of completions as candidate facets. We experiment with two variants of this facet provider: (1) **S-Bing**, which uses a single call to Autosuggest, resulting in at most 8 facets per topic and (2) the superset **B-Bing**, which makes makes 26 additional calls for a topic by appending to the query each letter of the alphabet, resulting in $8 + 8 * 26 = 216$ candidate facets per topic. Note that this can be seen as a breadth-first-search of the Autosuggest API, where nodes are expanded by this letter-appending technique. Though we restrict ourselves to a single level, this search can go deeper, to allow for more in-depth and comprehensive exploration of the user intent refinement task, using a simulator described next.

3.4 CoSearcher: User Simulator for Conversational Search

Our core contribution, CoSearcher, is the parameterized modeling of conversation search system users. The model is general, and is applicable to a broad set of conversational search tasks. It has two key components: (1) User Intent: a task-specific representation of the user's goals; and (2) User Parameters: values representing levels of cooperativeness and patience;

User Intent: In our search intent refinement use case, the goal is a search intent known only to the user, and the goal of a system is to discover this intent through a series of questions. The simulator returns a Boolean response depending on whether the question matches the intent.

Formally, the user model has a function $g(\textit{topic}, \textit{intent}, \textit{question})$ that returns a similarity score between the topic/intent and the question. The "Yes"/"No" is

[1] https://azure.microsoft.com/en-us/services/cognitive-services/autosuggest/.

then decided using a threshold that be chosen using downstream performance, or intrinsically evaluated if there is labeled “Yes”/“No” data.

CoSearcher Behavior Parameters: CoSearcher has two core parameters: cooperativeness and patience. Cooperativeness is a key user characteristic which has been *assumed* by conversational systems, and represents the users willingness to help the agent. Patience, representing the maximum number of interactions a user is willing to have with the conversational system, is based on the observation that user willingness to examine results diminishes over time [20]. Manipulating these two parameters via simulations enables us to expose the direct relationship between these key user behavior factors and conversational system results.

Cooperativeness: A user of a conversational system can be more cooperative by providing extra information (an *informative answer*) in addition to a minimal response. The informative answer can be task agnostic, by leaking the score from $g(\cdot)$ via answers such as “No, not even close”/“No, but you're close”, or directly leaking *intent* (with or without rewording), such as “No, I'm looking for *$intent*”. We define Cooperativeness as a Bernoulli random variable where p is the level of cooperativeness (i.e., a user with cooperativeness=0 only gives boolean answers, and a user with cooperativeness=1 always gives informative answers). Task-specific informative responses can be provided by making use of labeled data from human annotated informative answers, or by training a generative model using this data.

Patience: A user also has a patience level p, such that the conversation ends when the conversation exceeds a predefined number of turns p. This corresponds to the maximum amount of effort this user is willing to expand by interacting with the search system.

While in this paper we fix a user's patience and cooperativeness parameters throughout a conversation session, CoSearcher can also be configured to update these values dynamically, which can increase or decrease cooperativeness or patience of the user as the session progresses. In this work, we explore a wide range of these values through simulation, thus exhaustively testing the effect of user behavior on the success of a conversational search refinement system.

4 Experimental Setup

4.1 Resources and Evaluation

Our study uses only publicly available resources. The main dataset used is the previously described Qulac benchmark dataset [7]. Our “Yes”/“No” classifier fine-tunes the BERT-large uncased model from [15]. The similarity rankers use the LexVec [29] n-gram embeddings.[2] The IR search system is the same query-likelihood model used by [7][3] indexed on ClueWeb09b.

[2] https://github.com/alexandres/lexvec.

[3] Implementation distributed by authors at https://github.com/aliannejadi/qulac.

We measure the success of a dialogue by evaluating the relevance of the results retrieved using the enhanced query with identified user intent (topic + facet), using standard IR evaluation metrics: Mean Reciprocal Rank (MRR), Precision@k (P@k), and normalized Discounted Cumulative Gain@k (nDCG@k).

4.2 Conversational Intent Refinement Simulations

We now describe the concrete implementation of CoSearcher used to evaluate a conversational search refinement system under variety of conditions. Figure 1 shows the flow of an experiment for a given query topic and (hidden) true intent facet. For these experiments, the user intent is represented as a combination of topic and true intent facet, as described in Sect. 3.

To simulate cooperative users, we need a mechanism to provide informative answers that incorporate feedback. We achieve this through implementing for function $g(.)$ a simple heuristic to allow us to use a dataset such as Qulac (described above) to train CoSearcher. Specifically, we automatically label each instance *(topic, facet, question, answer)* in the Qulac dataset as follows: if *answer* contains "Yes"/"No" in its first three words, label *(topic, facet, question, answer, 1/0)* accordingly, else ignore it.

For CoSearcher to respond to a clarification question, we experiment with a variety of lexical and semantic matching mechanisms to determine a match between a question and a user's intended topic facet. We adapt the work on Semantic Textual Similarity (STS) for this task [1–5,10]. Specifically, we fine-tune the BERT-large model [15] which achieves state of the art performance on the STS Benchmark (STS-B) [10]. We use the same setup as used in [15] for the STS-B task, but train a binary classifier rather than a regressor. The input to BERT model is *"topic . intent [SEP] question"* using WordPiece tokenization, and the output is a match score - if a threshold is exceeded, CoSearcher returns "Yes" to indicate that the correct facet was proposed, and "No" otherwise (potentially with additional information as described above).

We split Qulac's 198 topics into 100 training, 25 validation, and 73 test topics, using only training and validation topics for the intent match classifier training/evaluation, and reserving the test topics as hidden for the full conversational system evaluations. At threshold 0.5, which we use throughout this paper, the classifier achieves an 0.63 F1 score. Figure 2a shows the resulting Precision/Recall curve of our trained classifier. This setup allows us to test our system with a fully configurable user. The system is run via a controller that selects the user parameters, including topic/facet and also initializes the interaction with the agent.

5 Results and Discussion

We formulate the IR query with the topic and the first facet to which the user model answers "Yes", or only the topic if no "Yes" is received before user patience runs out. We use the exact same Query-Likelihood IR model/data as in the

NeuQS paper [7].[4] Although submitting the entire dialogue could potentially improve search performance, since it includes human user responses which often contain paraphrases of the search facet, we opt to use only the system's best guess of what the correct facet is, as it excludes the previous (likely incorrect) facets discussed in the conversation.

We were not completely successful at adapting NeuQS to our exact problem setting (*explicit* intent refinement), so we compare our system using the Sim facet ranker to the results reported by [7] on the same overall IR task and dataset. We mimick the combinatorially-generated dialogue used as input to NeuQS by setting CoSearcher cooperativeness to 1 and patience to 3. Results are given in Table 1. Our system using Qulac facets has a larger gap to the Topic-only baseline (+.1061) than NeuQS to its Topic-only baseline (+.0910). Dynamic facet generation outperforms the topic-only baseline; we see that having a large number of candidate facets is important: B-Bing has 26x more facets than S-Bing, allowing for *finer matching.*

5.1 Effects of Patience and Cooperativeness

We set cooperativeness to 1 and vary the patience of the user model. Results are shown in Fig. 2c. We note that similarity based ranking always outperforms random selection, and retrieval improves as patience increases. Random facet selection is feasible when the set of candidate facets is small, as is the case with Qulac and S-Bing. The performance degrades substantially, however, for the larger B-Bing facet generator, remaining close to the baseline topic MRR (see Table 1). In contrast, semantic similarity ranking shows clear improvements as the conversation progresses.

We repeat these experiments, but this time vary the cooperativeness rather than patience (which is now fixed at 3). Results are shown in Fig. 2d. The Sim ranker clearly benefits from higher cooperativeness, while Rand shows no improvement, as expected. The considerable gap between B-Bing and S-Bing has a simple explanation: the user intent is less likely to be present in the small S-Bing set of facets than in the B-Bing superset, so additional cooperativeness helps one but no the other.

We next investigate the interaction between cooperativeness and patience, repeating the same setup from the previous IR experiments but this time varying both patience and cooperativeness. We study only B-Bing facets since these pose the hardest facet identification problem, requiring a deeper conversation to narrow down candidates. Results shown in Fig. 2b clearly indicate that *both* cooperativeness *and* patience are required to achieve maximal IR performance.

In sum, we showed that different CoSearcher configurations (user config, facet providers, etc.) led to a wide range of IR performances, demonstrating the functionality and applicability of our framework.

[4] Note that since they do not perform explicit intent refinement, they submit the entire dialogue context as a query to the IR system, whereas we submit only the topic and the refined facet.

Table 1. Performance comparison between prior state of the art methods, including [7] (top) and CoSearcher (bottom). "Topic-only" refers to the baseline method issuing only the topic as the query to the search system, ignoring any facet information obtained through conversation.

Method	MRR	P@1	nDCG@1	nDCG@5	nDCG@20
Topic-only	0.2715	0.1842	0.1381	0.1451	0.1470
σ-QPP	0.3570	0.2548	0.1960	0.1938	0.1812
LambdaMART	0.3558	0.2537	0.1945	0.1940	0.1796
RankNet	0.3573	0.2562	0.1979	0.1943	0.1804
NeuQS	**0.3625**	**0.2664**	**0.2064**	**0.2013**	**0.1862**
Topic-only	0.2938	0.1900	0.1329	0.1456	0.1525
CoSearcher- Qulac	**0.3999**	**0.3025**	**0.2263**	**0.2110**	**0.1908**
CoSearcher- S-Bing	0.3136	0.2010	0.1415	0.1653	0.1597
CoSearcher- B-Bing	0.3444	0.2366	0.1781	0.1769	0.1703

6 Analysis and Discussion

6.1 Characterization of Successful Conversational Refinement

Using the conversations generated with a wide range of behavior simulator features, we can explore what makes for a successful conversational search session. It is clear that the topic of the query has some effect on the difficulty of the task. We attempt to quantify this intuition through semantic analysis of the properties of search topics and facets to gain insight into the system performance.

We observe that ambiguous entities are associated with lower success rates across all facet providers. Examples of such entities with multiple senses include: *iron* (chemical element, clothing iron, nutritional supplement), *Euclid* (person, multiple businesses), *Rice* (food, person name, e.g., Rice university). Conversely, unambiguous entities are associated with much higher success rates, e.g., *Universal Animal Cuts* (a product), or *solar panels.* To quantify this we simulate 100 dialogues for each facet and measure the ratio of successful conversations. Using a sample of 20 topics (10 ambiguous entities, 10 non-ambiguous) we observe an average success rate of 55% for the ambiguous ones, compared to 72% for the non-ambiguous entities.

Similarly, topic ambiguity is a key factor. Topics that are broad in nature, with a large number of potential facets, yield poorer results. One such example is the topic *cass county missouri* with the facet "What was the 2008 budget for Cass County, MO?'. For a sample of 10 topics with $\geq$ 5 Qulac facets, we observe a mean success rate of 58%, against 66% for 10 topics with $\leq$ 3 facets. We hypothesize that it can be difficult to refine the query to such a specific facet within a reasonable number of turns.

Finally, facets containing multiple entities and entities that are complex noun phrases were often associated with poorer performance. For a sample of 10 topics

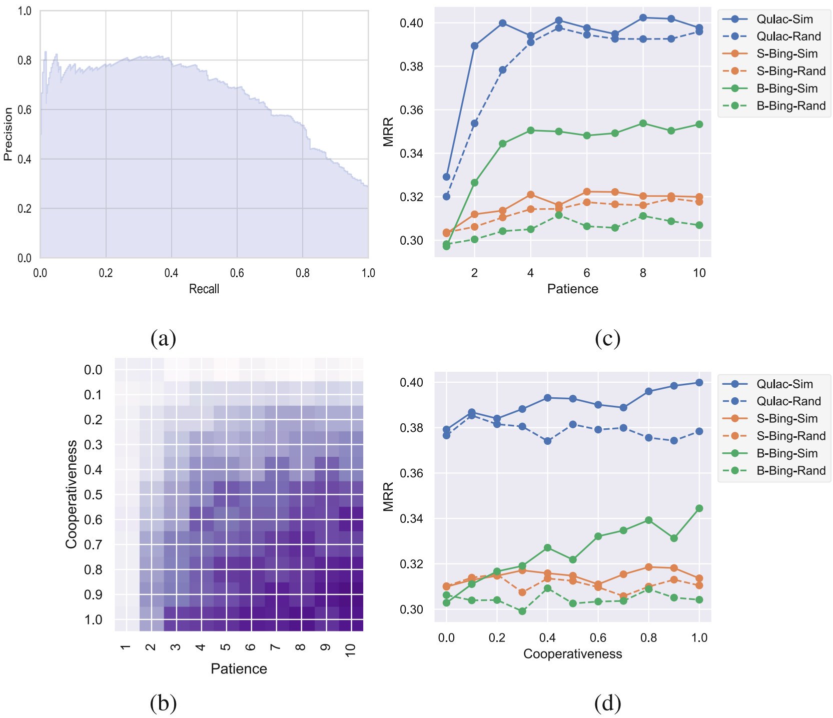


Fig. 2. (a) Precision-Recall curve of BERT Yes/No classifier on Qulac validation set. (b, c, d) The effect of varying patience/cooperativeness: (b) Heatmap of MRR for B-Bing using similarity facet ranker as patience/cooperativeness vary. (c) MRR for all facet providers using Sim and Rand facet rankers for cooperativeness=1 as patience varied. (d) Same as (c), but fixing patience at 3 and varying cooperativeness.

with complex entities, we observed an average success rate of 54%, compared to an overall average of 62%. These results indicates that entity extraction and disambiguation are key building blocks for successful conversational systems.

6.2 Qualitative Analysis: Case Studies

We complement our analysis above by offering case studies to provide intuition on why conversational search succeeds and fails in different situations under various user "personas" with varying degrees of cooperativeness. First, we consider an example of a cooperative user interacting with a system using the Qulac (static) topic facets, shown in Fig. 1b. Recall that for high value of cooperativeness, the user (and the simulator) often volunteer information to the search system, even if the initial response or guess was not correct, i.e., provide "informative no" responses. As a result, we observe the search system quickly converging on the true searcher intent. Another successful example using the Bing query suggestion facets is shown in Fig. 3a. Given the large number of relevant facets available via

USER: Quit Smoking *(the topic)*

BOT: **Are you looking for quit smoking withdrawal symptoms?**

COSEARCHER: no i am interested if it is possible to quit smoking cold turkey

BOT: **Are you looking for quit smoking cold turkey?**

COSEARCHER: yes specifically the cold turkey method

True intent: "Can you quit smoking using the cold turkey method?"

(a)

USER: iron *(the topic)*

BOT: **Are you looking for iron throne?**

COSEARCHER: no i want to know about iron deficiencies

BOT: **Are you looking for iron rich foods?**

COSEARCHER: no i would like information about iron deficiencies

BOT: **Are you looking for iron normal range?**

COSEARCHER: yes

True user intent: "Find info about iron deficiencies"

(b)

Fig. 3. (a) an example of a successful conversation (cooperativeness=1, Bing facets); (b) an example of a matching error (cooperativeness=1, Bing facets). The user incorrectly accepts a facet that is very closely related to the true intent.

the external search provider, the system is able to match the Qulac facet within 2 turns.

The example in Fig. 3a highlights the importance of realistically modeling "informative rejection" via our proposed cooperativeness parameter. In this example, a *cooperative* user volunteers her intent immediately, as soon as the system asks a clarification question. This is a known limitation of the Qulac dataset (which is crowdsourced with highly cooperative "users"), but may not be realistic. A more common scenario is that a user may not be able to fully specify her intent (hence the vague original query), but can easily recognize the topic facets she is, or is *not* interested in when prompted. The CoSearcher framework explicitly models and allows to automatically identify such cases. Consider a failed conversation (Fig. 3b), also with a cooperative user, using the Bing query suggestions (dynamic facets) as candidate facets. In the simulated conversation example below, the search system continues to ignore the search intent refinements volunteered by the cooperative CoSearcher user model, until the user simulator finally accepts the (incorrect) intent suggestion, likely resulting in non-relevant results.

We can see that in the above examples the system uses the information from the user to identify the true intent within a few turns. These examples provide additional intuition about the challenges in conversational search refinement, and illustrate the range of conversations and interactions that CoSearcher can support to simulate different types of users and search tasks.

7 Conclusions and Future Work

We investigated the effectiveness of conversational search refinement, a key task for conversational search systems. We hypothesized that the success of conversational search depends significantly on the users' behavior and the search task characteristics. To accomplish this, we introduced a parameterized conversational search user simulator, CoSearcher, to systematically probe the boundaries of conversational search intent refinement. CoSearcher was used to evaluate the effectiveness of query facet identification algorithm under a variety of conditions corresponding to different types of users. Our experiments on an existing benchmark (Qulac) and a new, dynamically generated dataset of search intent facets, demonstrate the power and generality of CoSearcher, exhibiting a new state of the art performance.

We also systematically explored the space of conversational search refinement outcomes for different types of search tasks and users. Specifically, we characterized the semantic differences between search topics and intents which are more (or less) amenable to conversational search refinement; We also empirically showed that (1) For the interesting real-world scenario where set of facets is large and a non-random facet ranker is used (B-Bing-Sim), cooperation on the user's part is fundamental for the success of conversational search refinement (in Fig. 2d, a uncooperative user's MRR in 3-turn-or-less dialogue is nearly identical to the .2938 topic-only baseline, improving up to .3444 as cooperativeness increases); and as illustrated in Fig. 2b), the effort (characterized by patience and cooperativeness) vs. benefit (MRR) tradeoff can be quantified: linear regression gives $MRR = .0038 \times \mathit{patience} + .034 \times \mathit{cooperativeness} + .29$ with $R^2 = 0.77$. (2) A simple semantic policy is effective for identifying searcher intent: in all experiments, it outperforms Random facet selection; in particular for B-Bing-Sim in Fig. 2c, MRR plateauing at 4 turns indicates that the best matching facet of the 216 candidates facets has been identified; (3) Dynamic search intent facet generation is feasible: MRR of .3444 for B-Bing-Sim is much higher than the topic-only baseline of .2938, suggesting a promising direction for future extensions by considering other sources of search intent facets.

We emphasize that the described results and analysis required simulating hundreds of thousands of conversational search refinement experiments, enabled by the presented CoSearcher simulator. In the future, we plan to expand CoSearcher to support more sophisticated behavior dynamics, which could be conditioned on the conversation length, search result quality, task characteristics, or other contextual factors. Additionally, CoSearcher is naturally suited for scenarios where the user intent is in natural language, but the system represents facets as database queries (e.g., over an e-commerce catalogue) and must select or generate these queries through dialogue.

The combination of the new state of the art results, our empirical insights, and the newly introduced flexible CoSearcher framework – complemented by the new dynamic search intent dataset to be released, provide significant progress towards more intelligent and effective conversational search systems.

References

1. Agirre, E., et al.: SemEval-2015 task 2: semantic textual similarity, English, Spanish and pilot on interpretability. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pp. 252–263. Association for Computational Linguistics, Denver, Colorado (June 2015). https://doi.org/10.18653/v1/S15-2045, https://www.aclweb.org/anthology/S15-2045
2. Agirre, E., et al.: SemEval-2014 task 10: Multilingual semantic textual similarity. In: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pp. 81–91. Association for Computational Linguistics, Dublin, Ireland (August 2014). https://doi.org/10.3115/v1/S14-2010, https://www.aclweb.org/anthology/S14-2010
3. Agirre, E., et al.: SemEval-2016 task 1: semantic textual similarity, monolingual and cross-lingual evaluation. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pp. 497–511. Association for Computational Linguistics, San Diego, California (June 2016). https://doi.org/10.18653/v1/S16-1081, https://www.aclweb.org/anthology/S16-1081
4. Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., Guo, W.: *SEM 2013 shared task: semantic textual similarity. In: Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity. pp. 32–43. Association for Computational Linguistics, Atlanta, Georgia, USA (June 2013), https://www.aclweb.org/anthology/S13-1004
5. Agirre, E., Diab, M., Cer, D., Gonzalez-Agirre, A.: Semeval-2012 task 6: a pilot on semantic textual similarity. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation. pp. 385–393. Association for Computational Linguistics (2012)
6. Aliannejadi, M., Kiseleva, J., Chuklin, A., Dalton, J., Burtsev, M.: ConvAI3: Generating Clarifying Questions for Open-Domain Dialogue Systems (ClariQ) (2020)
7. Aliannejadi, M., Zamani, H., Crestani, F., Croft, W.B.: Asking clarifying questions in open-domain information-seeking conversations. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 475–484. ACM (2019)
8. Belkin, N.J., Cool, C., Stein, A., Thiel, U.: Cases, scripts, and information-seeking strategies: on the design of interactive information retrieval systems. Expert Syst. Appl. **9**(3), 379–395 (1995)
9. Bordes, A., Boureau, Y.L., Weston, J.: Learning end-to-end goal-oriented dialog. arXiv preprint arXiv:1605.07683 (2016)
10. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp. 1–14. Association for Computational Linguistics, Vancouver, Canada (August 2017). https://doi.org/10.18653/v1/S17-2001, https://www.aclweb.org/anthology/S17-2001
11. Chandramohan, S., Geist, M., Lefèvre, F., Pietquin, O.: User simulation in dialogue systems using inverse reinforcement learning. In: Twelfth Annual Conference of the International Speech Communication Association (2011)

12. Clarke, C.L., Craswell, N., Soboroff, I.: Overview of the trec 2009 web track. WATERLOO UNIV (ONTARIO), Technical Report (2009)
13. Clarke, C.L., Craswell, N., Voorhees, E.M.: Overview of the TREC 2012 web track. Technical Report National Inst of Standards and Technology Gaithersburg MD (2012)
14. Croft, W.B., Thompson, R.H.: I3r: a new approach to the design of document retrieval systems. J. Am. Soc. Inf. Sci. **38**(6), 389–404 (1987)
15. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers), pp. 4171–4186 (2019)
16. El Asri, L., He, J., Suleman, K.: A sequence-to-sequence model for user simulation in spoken dialogue systems. Interspeech **2016**, 1151–1155 (2016)
17. Fagan, J.C.: Usability studies of faceted browsing: a literature review. Inf. Technol. Libr. **29**(2), 58–66 (2010)
18. Hearst, M.: Design recommendations for hierarchical faceted search interfaces. In: ACM SIGIR Workshop on Faceted Search, pp. 1–5. Seattle, WA (2006)
19. Hearst, M., Elliott, A., English, J., Sinha, R., Swearingen, K., Yee, K.P.: Finding the flow in web site search. Commun. ACM **45**(9), 42–49 (2002)
20. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. (TOIS) **20**(4), 422–446 (2002)
21. Kiesel, J., Bahrami, A., Stein, B., Anand, A., Hagen, M.: Toward voice query clarification. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 1257–1260. ACM (2018)
22. Kotov, A., Zhai, C.: Towards natural question guided search. In: Proceedings of the 19th International Conference on World Wide Web, pp. 541–550 (2010)
23. Kules, B., Capra, R., Banta, M., Sierra, T.: What do exploratory searchers look at in a faceted search interface? In: Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 313–322 (2009)
24. Li, X., Lipton, Z.C., Dhingra, B., Li, L., Gao, J., Chen, Y.N.: A user simulator for task-completion dialogues. arXiv preprint arXiv:1612.05688 (2016)
25. Mostafazadeh, N., Misra, I., Devlin, J., Mitchell, M., He, X., Vanderwende, L.: Generating natural questions about an image. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), pp. 1802–1813 (2016)
26. Papangelis, A., Papadakos, P., Kotti, M., Stylianou, Y., Tzitzikas, Y., Plexousakis, D.: Ld-sds: Towards an expressive spoken dialogue system based on linked-data (2017)
27. Rao, S., Daumé III, H.: Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), pp. 2737–2746 (2018)
28. Rao, S., Daumé III, H.: Answer-based adversarial training for generating clarification questions. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers), pp. 143–155 (2019)
29. Salle, A., Villavicencio, A.: Incorporating subword information into matrix factorization word embeddings. In: Proceedings of the Second Workshop on Subword/Character LEvel Models, pp. 66–71. Association for Computational Linguistics, New Orleans (June 2018). https://doi.org/10.18653/v1/W18-1209

30. Sun, Y., Zhang, Y.: Conversational recommender system. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 235–244. ACM (2018)
31. Trippas, J.R., Spina, D., Cavedon, L., Joho, H., Sanderson, M.: Informing the design of spoken conversational search: perspective paper. In: Proceedings of the 2018 Conference on Human Information Interaction & Retrieval, pp. 32–41. ACM (2018)
32. Tunkelang, D.: Faceted search. Synth. Lectures Inf. Concepts Retrieval Serv. **1**(1), 1–80 (2009)
33. Vandic, D., Aanen, S., Frasincar, F., Kaymak, U.: Dynamic facet ordering for faceted product search engines. IEEE Trans. Knowl. Data Eng. **29**(5), 1004–1016 (2017)
34. Weizenbaum, J., et al.: Eliza–a computer program for the study of natural language communication between man and machine. Commun. ACM **9**(1), 36–45 (1966)
35. Wen, T., et al.: A network-based end-to-end trainable task-oriented dialogue system. In: 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017-Proceedings of Conference. vol. 1, pp. 438–449 (2017)
36. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 401–408 (2003)
37. Young, S.J.: Probabilistic methods in spoken-dialogue systems. Philos. Trans. Royal Soc. London. Series A: Math. Phys. Eng. Sci. **358**(1769), 1389–1402 (2000)
38. Zamani, H., Lueck, G., Chen, E., Quispe, R., Luu, F., Craswell, N.: Mimics: a large-scale data collection for search clarification (2020)
39. Zhang, S., Balog, K.: Evaluating conversational recommender systems via user simulation. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1512–1520 (2020)

Causality-Aware Neighborhood Methods for Recommender Systems

Masahiro Sato(✉), Janmajay Singh, Sho Takemori, and Qian Zhang

Fuji Xerox, Yokohama, Japan
{sato.masahiro,janmajay.singh,takemori.sho,qian.zhang}@fujixerox.co.jp

Abstract. The business objectives of recommenders, such as increasing sales, are aligned with the causal effect of recommendations. Previous recommenders targeting for the causal effect employ the inverse propensity scoring (IPS) in causal inference. However, IPS is prone to suffer from high variance. The matching estimator is another representative method in causal inference field. It does not use propensity and hence free from the above variance problem. In this work, we unify traditional neighborhood recommendation methods with the matching estimator, and develop robust ranking methods for the causal effect of recommendations. Our experiments demonstrate that the proposed methods outperform various baselines in ranking metrics for the causal effect. The results suggest that the proposed methods can achieve more sales and user engagement than previous recommenders.

Keywords: Recommendation · Causal inference · Matching estimator

1 Introduction

Recommender systems have been used in various services to improve sales and user engagement [15]. For these purposes, it is essential to increase users' positive interactions, such as purchases and views. If recommended items are purchased or viewed, the recommendations are typically considered to be successful. However, the recommended items might have been interacted even without the recommendations. In this case, the user interactions are not *caused* by the recommendations. For example, if a user is an enthusiastic fan of a movie director, the user would watch a new movie of the director whether it is recommended or not. Sharma et al. [43] analyzed the browsing logs of an e-commerce site and revealed that at least 75% of recommended visits would likely occur in the absence of the recommendations. To improve sales and user engagement, it is important to generate recommendations that truly increase user interactions.

Such an increase produced purely by recommendation is called *causal effect*. Figure 1 illustrates the causal effect of recommendation. It is the difference of user interactions in two cases: if recommended and if not recommended. The challenge for ranking items by the causal effect is that we can not directly measure the causal effect since an item is either recommended or not for a specific user.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 603–618, 2021.
https://doi.org/10.1007/978-3-030-72113-8_40

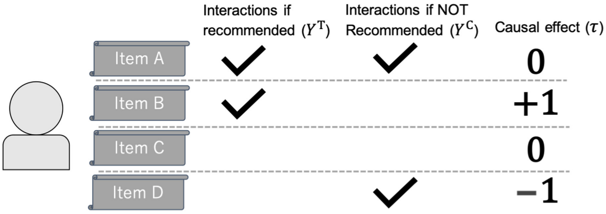

Interactions if recommended (Y^T)
Interactions if NOT Recommended (Y^C)
Causal effect (τ)
Item A
Item B
Item C
Item D
✓ ✓ 0
✓ +1
0
✓ −1

Fig. 1. A figure to illustrate the causal effect of recommendations. Recommending Item B results in increase of user interactions than without recommending, hence it has positive causal effect.

Such unobservable nature is a fundamental problem of causal inference [12] and various methods have been developed to address the problem [11,14].

Few works targeting recommendation causal effect exist [4,38–40], and it is largely an unexplored area of research. Among them, a recent work [40] employed IPS method [27] in causal inference field, and developed unbiased learning-to-rank methods. However, the IPS has been known to suffer from high variance due to small propensities [35,45,50]. Although the previous work [40] mitigates the variance by propensity capping, it incurs bias and affects the recommendation performance. The matching estimator [44] is another representative method in causal inference. It does not rely on propensities and enables a stable estimate of causal effect under various conditions of propensities. Despite the potential advantage, there have been no attempts to apply the matching estimator for the causal effect of recommendations.

In this work, we explore the matching estimator approach to rank items by the causal effect of recommendations. Matching estimators estimate causal effect by comparing observed outcomes for treated/untreated persons to those of similar persons in untreated/treated group. Leveraging person similarity is analogous to traditional neighborhood recommendation methods. We unify neighborhood recommendation methods with the matching estimator, and construct estimators of the causal effect for each user-item pair. To obtain item rankings robust to randomness of user behaviors, we further improve the estimators by 1) mixing own and neighbor observations and 2) introducing a shrinkage hyper-parameter to adjust outcome estimates depending on computed neighborhood size. We experimentally compare our methods with various baselines including recent IPS-based methods. The results demonstrate the effectiveness of our methods for ranking items by the causal effect. Such ranking can lead to increase of sales and user engagement, and have a practical benefit for businesses.

2 Related Work

Collaborative filtering is a widely used technique in recommender systems. It can be grouped into the two general classes: neighborhood and model-based methods [23,29]. Among model-based methods, matrix factorization models have been most popular [20,24] and recently neural network models are gaining popularity [53]. Neighborhood methods have been used since the dawn of recommender systems [37,42]. They are still competitive to recent neural model-based methods [7], especially in session-based recommendation [26]. In this work, we extend neighborhood methods for the causal effect of recommendations.

Early work of recommendation for the causal effect proposed a two-stage purchase prediction model comprising awareness and satisfaction [4], similar to recent exposure modeling [25]. It assumed that recommendations make users aware of the items. Later work [38] incorporated user- and item-dependent responsiveness to recommendations. Both methods predict purchase probabilities with and without recommendations and rank items by the difference of these probabilities. Another strategy is to directly optimize ranking models for the causal effect [39,40]. ULRMF and ULBPR [39] are heuristic pointwise and pairwise learning methods inspired by the label transformation [16,22] in uplift modeling [8,32]. Very recent work [40] proposed DLCE, an IPS-based unbiased learning-to-rank method for the causal effect.

The IPS has been gaining popularity in counterfactual learning [17,31]. It has been applied to address missing not at random recommender feedback [36,41], position bias in information retrieval [2,18,51], and selection bias in bandit feedback [6,54]. Domain adaptation is another counterfactual learning method [19] that are applied for recommenders [5]. To the best of our knowledge, matching estimator has not been applied for recommenders or information retrieval.

3 Preliminaries

3.1 Matching Estimator for Causal Inference

Let Y_n^{T} and Y_n^{C} be the *potential outcomes* [34] of subject n that would occur under treatment and control conditions, respectively. A potential outcome is one of possible two outcomes: one under treatment and another under control conditions. In medicine, for example, a subject is a patient, an outcome is recovery from disease, and treatment is to take a specific drug. Let Z_n be the indicator of treatment ($Z_n = 1$ if treated and $Z_n = 0$ if not treated). Observed outcome is expressed as: $Y_n = Z_n Y_n^{\mathrm{T}} + (1 - Z_n) Y_n^{\mathrm{C}}$. Note that $Y_n = Y_n^{\mathrm{T}}$ if $Z_n = 1$ and $Y_n = Y_n^{\mathrm{C}}$ if $Z_n = 0$. The causal effect τ_n is defined as the difference between the potential outcomes: $\tau_n = Y_n^{\mathrm{T}} - Y_n^{\mathrm{C}}$. However, τ_n can not be obtained since either Y_n^{T} or Y_n^{C} is observed for each subject. Matching estimator [44] estimates unobserved potential outcomes from the observed outcomes of the closest subjects.

$$\hat{Y}_n^{\mathrm{T}} = \frac{1}{|\mathcal{M}^{\mathrm{T}}(n)|} \sum_{m \in \mathcal{M}^{\mathrm{T}}(n)} Z_m Y_m, \quad \hat{Y}_n^{\mathrm{C}} = \frac{1}{|\mathcal{M}^{\mathrm{C}}(n)|} \sum_{m \in \mathcal{M}^{\mathrm{C}}(n)} (1 - Z_m) Y_m, \quad (1)$$

where $\mathcal{M}^{\mathrm{T}}(n)$ and $\mathcal{M}^{\mathrm{C}}(n)$ are sets of matched subjects under treatment and control conditions. Matched samples are typically chosen by similarity of subjects' covariates, e.g., demographics or previous medical histories. The causal effect τ_n is estimated as, $\hat{\tau}_n = Z_n \left(Y_n - \hat{Y}_n^{\mathrm{C}} \right) + (1 - Z_n) \left(\hat{Y}_n^{\mathrm{T}} - Y_n \right)$.

In the field of causal inference, we are mostly interested in the average treatment effect (ATE) or the average treatment effect on the treated (ATT), hence we take average of the above estimate over the set of subjects $\mathcal{S}$ or the set of treated subjects $\mathcal{S}^{\mathrm{T}}$: $\bar{\tau}_{\mathrm{ATE}} = \frac{1}{|\mathcal{S}|}\sum_{n\in\mathcal{S}}\hat{\tau}_n, \bar{\tau}_{\mathrm{ATT}} = \frac{1}{|\mathcal{S}^{\mathrm{T}}|}\sum_{n\in\mathcal{S}^{\mathrm{T}}}\hat{\tau}_n$. The higher the value, the treatment is considered to be more effective.

3.2 Neighborhood Method for Recommender System

Neighborhood methods are divided into user-based neighborhood (UBN) and item-based neighborhood (IBN) methods. Let $\mathcal{U}$ and $\mathcal{I}$ be sets of users and items, respectively, and $u, v \in \mathcal{U}$ and $i, j \in \mathcal{I}$. The predictions of UBN and IBN are expressed as follows.

$$\hat{Y}_{ui}^{\mathrm{UBN}} = \frac{\sum_{v\in\mathcal{N}(u)} w_{uv} Y_{vi}}{\sum_{v\in\mathcal{N}(u)} w_{uv}}, \quad \hat{Y}_{ui}^{\mathrm{IBN}} = \frac{\sum_{j\in\mathcal{N}(i)} w_{ij} Y_{uj}}{\sum_{j\in\mathcal{N}(i)} w_{ij}}, \tag{2}$$

where $\mathcal{N}(u)$ and $\mathcal{N}(i)$ are the sets of neighborhood users for u and neighborhood items for i, respectively. The weights w_{uv} and w_{ij} depend on the similarity between user pairs u and v, and between item pairs i and j, respectively.

The similarities are calculated based on previous interactions. In UBN, if user u and user v have positive interactions for same items, they are regarded to be similar. Popular choices for the similarity measure include cosine similarity, Pearson correlation, and Jaccard index among others [29]. The cosine similarity between users is expressed as, $\cos(u, v) = \boldsymbol{Y}_{u*} \cdot \boldsymbol{Y}_{v*} / ||\boldsymbol{Y}_{u*}||||\boldsymbol{Y}_{v*}||$, where $\boldsymbol{Y}_{u*} \equiv [Y_{u1}, Y_{u2}, ...Y_{u|\mathcal{I}|}]$ and $\boldsymbol{Y}_{v*} \equiv [Y_{v1}, Y_{v2}, ...Y_{v|\mathcal{I}|}]$ are vectors representing previous interactions for u and v, respectively. Top k users by the similarity measure are chosen as neighborhood $\mathcal{N}(u)$. The weight w_{uv} becomes $w_{uv} = (\cos(u, v))^{\alpha}$, where α is a scaling factor. $\mathcal{N}(i)$ and w_{ij} for IBN are derived analogously.

4 Causality-Aware Neighborhood Method

Using notations similar to Subsect. 3.1, the causal effect of recommending item i to user u is expressed as $\tau_{ui} = Y_{ui}^{\mathrm{T}} - Y_{ui}^{\mathrm{C}}$. In this setting, treatments are recommendations ($Z_{ui} = 1$ if recommended) and outcomes are users' interactions ($Y_{ui} = 1$ means positive interactions, such as purchases). Total interactions from recommendations is the sum of τ_{ui} in recommendation lists. Hence, we want to estimate τ_{ui} and rank items by the estimates. In this section, we unify the matching estimator in causal inference and the neighborhood methods for recommender systems, and propose causality-aware neighborhood methods to rank items for the causal effect of recommendations.

Estimating the unobserved potential outcomes is a key component for estimating the causal effect. We can apply UBN or IBN for the estimates.

$$\text{UBN: } \hat{Y}_{ui}^{\mathrm{T}} = \frac{\sum_{v\in\mathcal{N}(u)} w_{uv} Z_{vi} Y_{vi}}{\sum_{v\in\mathcal{N}(u)} w_{uv} Z_{vi}}, \ \hat{Y}_{ui}^{\mathrm{C}} = \frac{\sum_{v\in\mathcal{N}(u)} w_{uv}(1 - Z_{vi}) Y_{vi}}{\sum_{v\in\mathcal{N}(u)} w_{uv}(1 - Z_{vi})}, \tag{3}$$

$$\text{IBN: } \hat{Y}^{\mathrm{T}}_{ui} = \frac{\sum_{j \in \mathcal{N}(i)} w_{ij} Z_{uj} Y_{uj}}{\sum_{j \in \mathcal{N}(i)} w_{ij} Z_{uj}}, \; \hat{Y}^{\mathrm{C}}_{ui} = \frac{\sum_{j \in \mathcal{N}(i)} w_{ij} (1 - Z_{uj}) Y_{uj}}{\sum_{j \in \mathcal{N}(i)} w_{ij} (1 - Z_{uj})}. \tag{4}$$

Note that these estimates require only observed variables. Direct application of the matching estimator to our setting yields the formula below,

$$\hat{\tau}_{ui} = Z_{ui} \left(Y_{ui} - \hat{Y}^{\mathrm{C}}_{ui} \right) + (1 - Z_{ui}) \left(\hat{Y}^{\mathrm{T}}_{ui} - Y_{ui} \right). \tag{5}$$

The observed outcome Y_{ui} is used either as Y^{T}_{ui} or Y^{C}_{ui}.

However, user behavior is not deterministic and the observed outcome has a random noise.[1] Hence we mix the own interaction Y_{ui} and the neighbor interactions Y_{vi} or Y_{uj} to reduce random noises. More specifically, we include u and i in $\mathcal{N}(u)$ and $\mathcal{N}(i)$, respectively, and we set $w_{uu} = 1$ and $w_{ii} = 1$.

To further reduce the variance, we force the estimates to shrink to zero if they rely on a few neighbors with low similarity. We introduce shrinkage parameters β^{T} and β^{C} for the estimates of $\hat{Y}^{\mathrm{T}}_{ui}$ and $\hat{Y}^{\mathrm{C}}_{ui}$, respectively, and add them in the denominator.

$$\text{UBN: } \hat{Y}^{\mathrm{T}}_{ui} = \frac{\sum_{v \in \mathcal{N}'(u)} w_{uv} Z_{vi} Y_{vi}}{\beta^{\mathrm{T}} + \sum_{v \in \mathcal{N}'(u)} w_{uv} Z_{vi}}, \; \hat{Y}^{\mathrm{C}}_{ui} = \frac{\sum_{v \in \mathcal{N}'(u)} w_{uv} (1 - Z_{vi}) Y_{vi}}{\beta^{\mathrm{C}} + \sum_{v \in \mathcal{N}'(u)} w_{uv} (1 - Z_{vi})}, \tag{6}$$

$$\text{IBN: } \hat{Y}^{\mathrm{T}}_{ui} = \frac{\sum_{j \in \mathcal{N}'(i)} w_{ij} Z_{uj} Y_{uj}}{\beta^{\mathrm{T}} + \sum_{j \in \mathcal{N}'(i)} w_{ij} Z_{uj}}, \; \hat{Y}^{\mathrm{C}}_{ui} = \frac{\sum_{j \in \mathcal{N}'(i)} w_{ij} (1 - Z_{uj}) Y_{uj}}{\beta^{\mathrm{C}} + \sum_{j \in \mathcal{N}'(i)} w_{ij} (1 - Z_{uj})}. \tag{7}$$

Here the sets of neighbors $\mathcal{N}'(u)$ and $\mathcal{N}'(i)$ include u and i themselves. With Eqs. (6) and (7), we estimate the causal effect as,

$$\hat{\tau}_{ui} = \hat{Y}^{\mathrm{T}}_{ui} - \hat{Y}^{\mathrm{C}}_{ui}, \tag{8}$$

where the own interaction Y_{ui} is included in either $\hat{Y}^{\mathrm{T}}_{ui}$ or $\hat{Y}^{\mathrm{C}}_{ui}$ depending on Z_{ui}. Finally, to generate recommendation lists, items are ranked by the descending order of $\hat{\tau}_{ui}$ for each user.

We call our causality-aware user-based and item-based neighborhood methods as CUBN and CIBN, respectively. To calculate similarity of users or items, we can use previous interactions, similar to original UBN and IBN. We can also use the similarity based on previous treatment assignments $\boldsymbol{Z}_{u*} \equiv [Z_{u1}, Z_{u2}, ... Z_{u|\mathcal{I}|}]$ since we can expect that similar users receive similar recommendations if recommendations are properly personalized. We suffix -O or -T in the names of our methods to clarify whether outcomes or treatment assignments are used. The pseudo code of CUBN-O is shown in Algorithm 1. Here $\text{rank}_u(\hat{\tau}_{ui})$ is the ranking position of item i for user u when items are sorted by $\hat{\tau}_{ui}$ in descending order. Cosine similarity is used in this work. To obtain the algorithm for CUBN-T, line 4 is substituted with $w_{uv} \leftarrow (\boldsymbol{Z}_{u*} \cdot \boldsymbol{Z}_{v*} / ||\boldsymbol{Z}_{u*}|| ||\boldsymbol{Z}_{v*}||)^{\alpha}$.

[1] If we focus on ATE or ATT, as often the case in causal inference, the random noise is not a severe problem since it disappears by taking average of large samples. It becomes a problem when we want to rank items by the estimates for each item.

Algorithm 1: Causality-aware User-Based Neighborhood method by Outcome similarity (*CUBN-O*).

```
Input: $k$, $\alpha$, $\beta^{\mathrm{T}}$, $\beta^{\mathrm{C}}$, $\{Y_{ui}\}$, $\{Z_{ui}\}$
Output: $\{L_u | u \in \mathcal{U}\}$
// Phase1: neighborhood preparation
for $u \in \mathcal{U}$ do
    for $v \in \mathcal{U}$ do
        $w_{uv} \leftarrow \left(\frac{\boldsymbol{Y}_{u*} \cdot \boldsymbol{Y}_{v*}}{||\boldsymbol{Y}_{u*}||||\boldsymbol{Y}_{v*}||}\right)^{\alpha}$    // cosine similarity with scaling
    $\mathcal{N}'(u) \leftarrow \arg\max_{\mathcal{G}(u)\subset\mathcal{U},|\mathcal{G}(u)|=k} \sum_{v\in\mathcal{G}(u)} w_{uv}$    // top-$k$ neighbors
// Phase2: item ranking
for $u \in \mathcal{U}$ do
    for $i \in \mathcal{I}$ do
        $\hat{\tau}_{ui} \leftarrow \frac{\sum_{v\in\mathcal{N}'(u)} w_{uv} Z_{vi} Y_{vi}}{\beta^{\mathrm{T}} + \sum_{v\in\mathcal{N}'(u)} w_{uv} Z_{vi}} - \frac{\sum_{v\in\mathcal{N}'(u)} w_{uv}(1-Z_{vi}) Y_{vi}}{\beta^{\mathrm{C}} + \sum_{v\in\mathcal{N}'(u)} w_{uv}(1-Z_{vi})}$
    $L_u \leftarrow \{\mathrm{rank}_u(\hat{\tau}_{ui}) | i \in \mathcal{I}\}$    // ranking list by descending order of $\hat{\tau}_{ui}$
return $\{L_u | u \in \mathcal{U}\}$
```

Standard collaborative filtering methods use only interaction logs $\{Y_{ui}\}$. Our methods require previous recommendation logs $\{Z_{ui}\}$ in addition. We assume that a certain recommender is already deployed in the service and we have the logs of the recommender.[2] Recommendation logs are commonly needed for previous methods targeting the causal effect [4,38–40]. The previous IPS-based method [40] further requires propensity, i.e., the probability of recommendations. Our methods do not use propensity, hence we believe they are easier to deploy.

Our methods are based on standard assumptions of causal inference: ignorability, no interference, and no multiple versions [11,14].[3] The *ignorability* assumption implies that treatment assignment (Z_{ui}) is independent of the potential outcomes ($Y_{ui}^{\mathrm{T}}, Y_{ui}^{\mathrm{C}}$) given the covariates ($X_u, X_i$): $Y_{ui}^{\mathrm{T}}, Y_{ui}^{\mathrm{C}} \perp Z_{ui} | X_u, X_i$ (see also causal graph of Fig. 1(b) in [40]). Here X_u and X_i are features of user u and item i, respectively. We assume that user neighbors $\mathcal{N}(u)$ and item neighbors $\mathcal{N}(i)$ have features similar to user u and item i, respectively. The *no interference* assumption means that a recommendation (Z_{ui}) does not affect other users' or items' outcomes (Y_{vi} or Y_{uj}). As a result of this assumption, there is no influence by item sequences in recommendation lists. The *no multiple versions* assumption states that there is only a single version of recommendation. There could be several ways to recommend items, such as browser pop-ups and sending e-mails, but we assume that only one way is chosen for each dataset. Relaxing these

[2] Note that the deployed recommender is different from recommenders that we train and evaluate from $\{Y_{ui}\}$ and $\{Z_{ui}\}$, hence we might not have control over previous recommendation logs. In experiment section, we also investigate how different conditions of previous recommendations affect the proposed recommenders.

[3] The latter two taken together are called the *stable unit treatment value assumption* (SUTVA).

assumptions is an active area of research in causal inference [13,49,52] and is also interesting future direction of this study.

5 Experiments

5.1 Experimental Settings[4]

Datasets. We used the MovieLens (ML)[5] 100K and 1M datasets, and the Dunnhumby (DH)[6] dataset. The ML datasets [10] contains five-star movie ratings. The DH dataset contains purchase and promotion logs from grocery stores. For DH, we followed procedure described in [40] to generate a semi-synthetic dataset in *Original* (DH-Ori) and *Personalized* (DH-Per) settings. For ML, we generated semi-synthetic datasets as follows,

1. The ratings of all user-item pairs $\{\hat{R}_{ui}\}$ were predicted using rating matrix factorization [24].
2. The probabilities of observing the ratings $\{\hat{O}_{ui}\}$ were predicted using logistic matrix factorization [20].
3. The probabilities of positive outcomes with and without recommendations were formulated as follows.

$$\mu_{ui}^{\mathrm{T}} = \sigma(\hat{R}_{ui} - \epsilon), \quad \mu_{ui}^{\mathrm{C}} = \hat{O}_{ui}. \tag{9}$$

   Here σ is a sigmoid function that converts predicted ratings $\hat{R}_{ui} \in [1,5]$ to probabilities $\mu_{ui}^{\mathrm{T}} \in [0,1]$. We set $\epsilon = 5.0$ the same as [36].
4. The propensities were determined by users' preferences to items.

$$P_{ui} = \min\left(1, a\,(1/\mathrm{rank}_u)^b\right). \tag{10}$$

   Here rank_u is item rankings by $\mu_{ui}^{\mathrm{T}} + \mu_{ui}^{\mathrm{C}}$. The parameters a and b control the average and the unevenness of propensities, respectively. We set $b = 1.0$ for the default condition. The average number of recommendations for users was set to 100 by adjusting a.
5. The potential outcomes under treatment and control conditions, and recommendation assignments were sampled as follows.

$$Y_{ui}^{\mathrm{T}} \sim \mathrm{Bernoulli}(\mu_{ui}^{\mathrm{T}}), \quad Y_{ui}^{\mathrm{C}} \sim \mathrm{Bernoulli}(\mu_{ui}^{\mathrm{C}}), \quad Z_{ui} \sim \mathrm{Bernoulli}(P_{ui}). \tag{11}$$

   Then, causal effect τ_{ui} and observed outcome Y_{ui} were obtained as,

$$\tau_{ui} = Y_{ui}^{\mathrm{T}} - Y_{ui}^{\mathrm{C}}, \quad Y_{ui} = Z_{ui}Y_{ui}^{\mathrm{T}} + (1 - Z_{ui})Y_{ui}^{\mathrm{C}}. \tag{12}$$

   Note that τ_{ui} was provided only for evaluation. This sampling can be repeated n times for each user-item pair. We independently sampled training, validation, and test data, and used for the purposes.

[4] The codes and chosen hyper parameters for each method are available as ancillary files at http://arxiv.org/abs/2012.09442.

[5] https://grouplens.org/datasets/movielens.

[6] https://www.dunnhumby.com/careers/engineering/sourcefiles.

The steps 1, 2 and 3 are similar to that of [36]. The steps 4 and 5 are similar to steps 3 and 4 of [40]. Unlike [40], we generated only one observation for each user-item pair for training data (i.e., we set $n_{train} = 1$ as opposed to $n_{train} = 10$ in [40]) since this setting more directly reflects the unobservable nature of the causal effect. The reasoning of Eq. (9) in step 3 is as follows. A choice of a movie to watch (O_{ui}) may be said to depend on expected entertainment from watching it. A rating (R_{ui}) reflects the experienced entertainment value after watching the movie. If a user knew the entertainment value before consumption, the user would choose movies based on this. Recommendations are often provided with explanations [47] and the explanations help users predict entertainment values of items [3,46]. Hence we related the watching probability with recommendation μ_{ui}^{T} to experienced entertainment value R_{ui}, and the watching probability without recommendation μ_{ui}^{C} to users' natural watching behavior O_{ui}.

The statistics of generated datasets are summarized in Table 1. ATE over whole user-item pairs are positive, meaning that recommendations generally tend to promote user interactions. We also confirmed that $\mu_{ui}^{\mathrm{T}} > \mu_{ui}^{\mathrm{C}}$ for about 90% of user-item pairs in the ML datasets and about 80% of user-item pairs in the DH datasets. However, $\mu_{ui}^{\mathrm{T}} < \mu_{ui}^{\mathrm{C}}$ for the remaining pairs and thus τ_{ui} tend to be negative for those pairs. Recommendations can have negative impact when they create bad feelings for users, e.g., creepiness [48]. Note that τ_{ui} can become negative by the randomness of user behaviors when $\mu_{ui}^{\mathrm{T}} \approx \mu_{ui}^{\mathrm{C}}$.

Compared Methods. The following methods were compared.

- **Random**: Items are ranked randomly.
- **Pop**: Items are ranked by popularity, i.e., number of positive outcomes.
- **UBN/IBN**: Traditional user-based and item-based neighborhood methods.
- **BPR** [33]: A commonly used pairwise learning method.
- **CausE** [5]: A joint training of prediction models for Y_{ui}^{T} and Y_{ui}^{C}.
- **ULRMF/ULBPR** [39]: Pointwise and pairwise learning methods for τ_{ui}.
- **DLTO/DLCE** [40]: IPS-based unbiased learning methods for Y_{ui}^{T} and τ_{ui}.
- **CUBN/CIBN**: Our causality-aware user-based and item-based neighborhood methods for τ_{ui}.

By comparing CUBN/CIBN and UBN/IBN, we verify whether our methods successfully extend UBN/IBN for the causal effect. We also compare our neighborhood methods with previous model-based methods targeting the causal effect:

Table 1. Statistics of generated datasets.

Dataset	#User	#Item	$\{Y_{ui} = 1\}$	$\{Z_{ui} = 1\}$	ATE
DH-Original	2,309	1,372	35,010	483,660	0.0044
DH-Personalized	2,309	1,372	37,731	483,727	0.0045
ML-100K	943	1,682	92,523	94,054	0.0735
ML-1M	6,040	3,952	985,994	603,108	0.0981

ULBPR, ULRMF, and DLCE. Previous research [39,40] shows that CausE and DLTO are also strong baselines, hence we included them. Our methods can use treatment assignments or positive outcomes for calculating user/item similarities. We suffix -T or -O to clarify which one is used. To investigate the effectiveness of mixing own and neighbor interactions, we also experimented on our methods without the mixture (-woM), i.e., Eqs. (3)–(5) are used instead of Eqs. (6)–(8).

Evaluation Protocols. Commonly used accuracy metrics, such as precision, reward positive interactions even if that would occur in the absence of recommendation (e.g., item A in Fig. 1.) We want to reward positive interactions purely caused by recommendation (e.g., item B in Fig. 1), and the accuracy metrics is not suitable (see also Sect. 2.1 in [39]). Hence, we used the causal variants of precision@n (CP@n), discounted cumulative gain (CDCG), and average rank (CAR) [40]. They are expressed respectively as,

$$\sum_i \frac{\mathbf{1}(\mathrm{rank}_u(\hat{s}_{ui}) \le n)\tau_{ui}}{n}, \quad \sum_i \frac{\tau_{ui}}{\log_2(1+\mathrm{rank}_u(\hat{s}_{ui}))}, \quad \frac{1}{I}\sum_i \mathrm{rank}_u(\hat{s}_{ui})\tau_{ui}$$

where $\hat{s}_{ui}$ is the predicted score of item i for user u and $\mathrm{rank}_u(\hat{s}_{ui})$ is the ranking position of the item. Items are ranked by the descending order of $\hat{s}_{ui}$. In our methods, items are ranked by the causal effect estimates $\hat{\tau}_{ui}$, i.e., $\hat{s}_{ui} = \hat{\tau}_{ui}$. We calculated the above metrics for each user and took average over all users. Note that τ_{ui} is a ternary variable ($\tau_{ui} \in \{1, 0, -1\}$) and the metrics can be negative.

The hyper parameters of each method were tuned with validation data to optimize each metric, i.e., chosen parameters were different for each metric. We used the same shrinkage parameters for treatment and control ($\beta = \beta^{\mathrm{T}} = \beta^{\mathrm{C}}$). The exploration ranges for the proposed methods were as follows: the maximum number of neighbors $\in \{10, 30, 100, 300, 1000, 3000, 10000\}$, the scaling factor $\alpha \in \{0.33, 0.5, 1.0, 2.0, 3.0, 5.0\}$, and the shrinkage parameter $\beta \in \{0, 0.3, 1, 3, 10, 30, 100\}$. The exploration ranges for other baselines were same with [40].

5.2 Results and Discussions

Performance Comparison. Tables 2 and 3 show the performance comparison. The best among previous methods differ for datasets. Our CUBNs constantly outperform them in all datasets. CIBNs perform worse but are still competitive to other baselines. CUBN-O and CUBN-T tend to perform similarly, and any differences depend on datasets and metrics. CUBN-O uses previous outcomes for user similarities same as traditional UBN. On the other hand, CUBN-T uses previous treatment assignments for user similarities that is original to our work. The result indicates that similarity of previous treatment assignments can provide good measure of user similarities. Furthermore, CUBN and CIBN counterparts not using own and neighborhood interaction mixtures (-woM) are often outperformed by methods which do, showing its importance.

Table 2. Performance comparison in the Dunnhumby (DH) dataset. The best results are highlighted in bold. Note that the smaller is better in CAR.

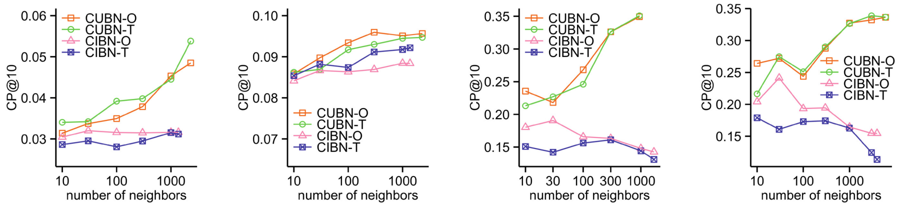
	DH-Original				DH-Personalized			
	CP@10	CP@100	CDCG	CAR	CP@10	CP@100	CDCG	CAR
Random	0.0046	0.0049	0.726	3.01	0.0048	0.0044	0.672	2.84
Pop	0.0293	0.0157	0.925	1.86	0.0275	0.0131	0.858	1.64
BPR	0.0331	0.0153	0.923	1.86	0.0564	0.0187	0.858	1.54
UBN	0.0294	0.0153	0.926	1.87	0.0419	0.0190	0.922	1.36
IBN	0.0301	0.0138	0.903	1.94	0.0438	0.0179	0.928	1.49
CausE	0.0337	0.0204	1.009	1.95	0.0857	0.0186	1.110	1.39
ULRMF	0.0359	0.0168	0.937	**1.78**	0.0802	0.0203	1.005	1.39
ULBPR	0.0343	0.0143	0.918	1.80	0.0806	0.0209	1.038	1.32
DLTO	0.0358	0.0151	0.955	1.82	0.0813	0.0198	1.063	1.41
DLCE	0.0354	0.0116	0.882	2.70	0.0839	0.0209	1.036	1.38
CUBN-O	0.0424	0.0193	0.986	1.98	0.0877	0.0240	1.124	1.24
CUBN-T	**0.0513**	**0.0216**	**1.030**	**1.78**	0.0890	**0.0257**	1.112	**1.13**
CIBN-O	0.0328	0.0110	0.892	2.43	0.0871	0.0190	1.112	1.36
CIBN-T	0.0301	0.0095	0.872	2.61	0.0889	0.0181	**1.135**	1.61
CUBN-O-woM	0.0437	0.0186	0.979	2.20	**0.0902**	0.0199	1.107	1.30
CUBN-T-woM	0.0436	0.0198	0.991	2.10	0.0901	0.0124	1.005	2.40
CIBN-O-woM	0.0382	0.0140	0.909	2.38	0.0738	0.0175	1.008	1.39
CIBN-T-woM	0.0333	0.0098	0.890	2.69	0.0881	0.0168	1.098	2.03

Table 3. Performance comparison in the MovieLens (ML) 100K and 1M datasets. The best results are highlighted in bold. Note that the smaller is better in CAR.

	ML-100K				ML-1M			
	CP@10	CP@100	CDCG	CAR	CP@10	CP@100	CDCG	CAR
Random	0.076	0.075	13.9	61.8	0.097	0.098	38.0	194
Pop	-0.215	-0.085	11.3	73.7	-0.135	-0.042	35.5	196
BPR	0.092	0.088	14.0	61.7	0.102	0.103	38.1	194
UBN	-0.217	-0.102	11.1	66.6	-0.175	-0.058	35.2	165
IBN	0.098	0.099	14.0	63.2	0.052	0.055	36.8	177
CausE	0.310	0.214	16.4	34.4	0.309	0.246	42.4	122
ULRMF	0.302	0.148	15.8	39.0	0.160	0.152	39.9	152
ULBPR	0.333	0.163	15.6	43.9	0.245	0.187	40.4	143
DLTO	0.330	0.155	15.3	53.2	0.289	0.202	40.5	152
DLCE	0.330	0.215	16.6	28.8	0.319	**0.258**	42.4	119
CUBN-O	0.349	**0.218**	**16.9**	27.2	0.334	**0.258**	**42.7**	116
CUBN-T	**0.350**	**0.218**	16.8	**25.9**	**0.336**	0.256	42.6	127
CIBN-O	0.184	0.145	15.5	30.0	0.236	0.186	41.1	120
CIBN-T	0.160	0.149	15.6	31.8	0.188	0.173	40.9	122
CUBN-O-woM	0.310	0.194	16.6	29.0	0.291	0.233	42.4	115
CUBN-T-woM	0.311	0.194	16.6	29.0	0.294	0.237	42.4	**114**
CIBN-O-woM	0.147	0.123	15.1	34.6	0.216	0.183	40.6	117
CIBN-T-woM	0.118	0.126	15.2	34.5	0.160	0.168	40.8	123

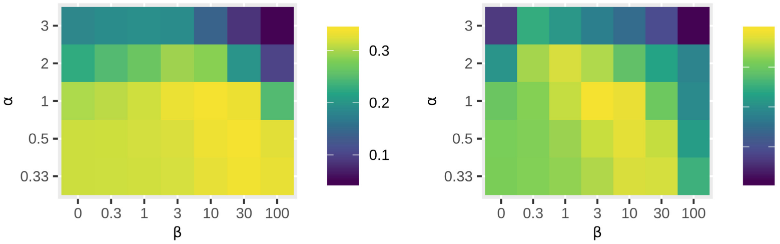
Dependence on Hyper Parameters. As our methods are neighborhood methods, the dependence on the number of neighbors is important. Figure 2 shows the results. General trends show that performance improves with increasing numbers of neighbors. In ML-100K and ML-1M datasets, CIBNs reach maximum performance with relatively smaller numbers of neighbors.

Our methods have other two key hyper-parameters: the scaling factor α and the shrinkage parameter β. We investigated the dependence on these parameters (Fig. 3). The best performances were obtained at $\beta > 0$, showing the effectiveness of introducing the shrinkage. Optimal β for CP@10 is larger than that for CP@100. This trend was similarly observed in other datasets. We suppose that inappropriate item selection by random noise of causal effect estimates affects CP more severely when recommendation list is small, thus the shrinkage β should be larger for CP@10.



(a) DH-Ori. (b) DH-Per. (c) ML-100K. (d) ML-1M.

Fig. 2. Dependence on the number of neighbors in validation datasets. The scaling factor α and the shrinkage parameter β are set to the optimal values for each number of neighbors. Note that the possible number of neighbors are restricted by either the number of users or that of items.

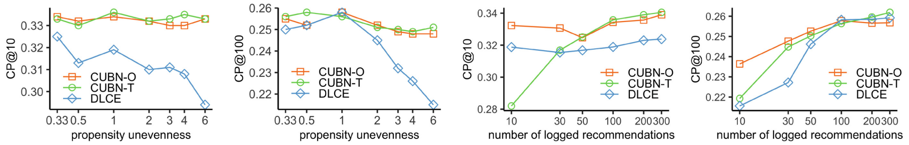


(a) ML-1M (CP@10). (b) ML-1M (CP@100).

Fig. 3. Dependence on the scaling factor α and the shrinkage parameter β for CUBN-O in the ML-1M dataset. The number of neighbors are set to 6,040.

Influence of Difference in Previous Recommendation Logs. IPS are known to suffer from variance by very small propensities. This happens when recommendation assignments shift toward deterministic assignments, i.e., propensities are close to 0.0 or 1.0. In our semi-synthetic data generation, increasing unevenness parameter b in Eq. (10) makes recommendations more deterministic. Hence we investigated how it affects our methods and IPS-based previous method (DLCE). As seen from Fig. 4(a, b), DLCE degrades with increasing unevenness. On the other hand, our methods are more robust to this unevenness.

Recommendation methods targeting the causal effect commonly require recommendation logs. Here we investigated how the number of logged recommendations for each user affects the performance. For CP@10 (Fig. 4(c)), the performances of CUBN-O and DLCE are mostly stable, while CUBN-T degrades with less number of logged recommendations. This is reasonable considering that CUBN-T obtains neighbors by the similarity of recommendation assignments. For CP@100 (Fig. 4(d)), all methods are affected by the number of logged recommendations, but CUBN-O is relatively robust.



(a) CP@10. (b) CP@100. (c) CP@10. (d) CP@100.

Fig. 4. Performances under the varied unevenness of propensity (a, b) and the varied number of logged recommendations per user (c, d).

6 Conclusions

We proposed causality-aware neighborhood methods to generate item ranking by the causal effect of recommendations. We unified traditional neighborhood-based recommendation methods with matching estimator, and further enhanced them by mixing the own and neighbor observations and introducing the shrinkage for potential outcome estimates. Models proposed in this paper outperformed baselines on causal effect versions of commonly used ranking metrics. This was particularly true for models augmenting user-based neighborhood methods for causal effect. The results suggest that these models can lead to improved sales and user engagement and are thus highly beneficial for businesses employing recommender systems. In the future work, our methods can be enhanced by applying graph-based neighborhood similarities [9,28] or by learning neighborhood similarities [21,30]. Another direction of future work is to leverage contextual information [1]. Since neighborhood methods are known to be effective in session-based recommendations [26], it would be also interesting to extend our methods for session-based recommendations.

References

1. Adomavicius, G., Sankaranarayanan, R., Sen, S., Tuzhilin, A.: Incorporating contextual information in recommender systems using a multidimensional approach. ACM Trans. Inf. Syst. **23**(1), 103–145 (2005). https://doi.org/10.1145/1055709.1055714
2. Agarwal, A., Takatsu, K., Zaitsev, I., Joachims, T.: A general framework for counterfactual learning-to-rank. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, pp. 5–14. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3331184.3331202
3. Bilgic, M., Mooney, R.J.: Explaining recommendations: satisfaction vs. promotion. In: Beyond Personalization Workshop, IUI, vol. 5, p. 153 (2005)
4. Bodapati, A.V.: Recommendation systems with purchase data. J. Mark. Res. **45**(1), 77–93 (2008)
5. Bonner, S., Vasile, F.: Causal embeddings for recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, pp. 104–112. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3240323.3240360
6. Bottou, L., et al.: Counterfactual reasoning and learning systems: the example of computational advertising. J. Mach. Learn. Res. **14**(1), 3207–3260 (2013)
7. Dacrema, M.F., Cremonesi, P., Jannach, D.: Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, pp. 101–109. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3298689.3347058
8. Devriendt, F., Moldovan, D., Verbeke, W.: A literature survey and experimental evaluation of the state-of-the-art in uplift modeling: a stepping stone toward the development of prescriptive analytics. Big Data **6**(1), 13–41 (2018)
9. Fouss, F., Pirotte, A., Renders, J.M., Saerens, M.: Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Trans. Knowl. Data Eng. **19**(3), 355–369 (2007)
10. Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. ACM Trans. Interact. Intell. Syst. 5(4) (2015). https://doi.org/10.1145/2827872
11. Hernán, M., Robins, J.: Causal Inference: What If. Chapman & Hill/CRC, Boca Raton (2020)
12. Holland, P.W.: Statistics and causal inference. J. Am. Stat. Assoc. **81**(396), 945–960 (1986)
13. Hudgens, M.G., Halloran, M.E.: Toward causal inference with interference. J. Am. Stat. Assoc. **103**(482), 832–842 (2008)
14. Imbens, G.W., Rubin, D.B.: Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction. Cambridge University Press, New York (2015)
15. Jannach, D., Jugovac, M.: Measuring the business value of recommender systems. ACM Trans. Manage. Inf. Syst. **10**(4) (2019). https://doi.org/10.1145/3370082
16. Jaskowski, M., Jaroszewicz, S.: Uplift modeling for clinical trial data. In: ICML Workshop on Clinical Data Analysis (2012)
17. Joachims, T., Swaminathan, A.: Counterfactual evaluation and learning for search, recommendation and ad placement, SIGIR 2016, pp. 1199–1201. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2911451.2914803

18. Joachims, T., Swaminathan, A., Schnabel, T.: Unbiased learning-to-rank with biased feedback. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, pp. 781–789. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3018661.3018699
19. Johansson, F.D., Shalit, U., Sontag, D.: Learning representations for counterfactual inference. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning, ICML 2016, vol. 48, pp. 3020–3029. JMLR.org (2016)
20. Johnson, C.C.: Logistic matrix factorization for implicit feedback data. Adv. Neural. Inf. Process. Syst. **27**, 1–9 (2014)
21. Kabbur, S., Ning, X., Karypis, G.: FISM: factored item similarity models for top-N recommender systems. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, pp. 659–667. Association for Computing Machinery, New York (2013). https://doi.org/10.1145/2487575.2487589
22. Kane, K., Lo, V.S., Zheng, J.: Mining for the truly responsive customers and prospects using true-lift modeling: comparison of new and existing methods. J. Mark. Anal. **2**(4), 218–238 (2014)
23. Koren, Y., Bell, R.: Advances in collaborative filtering. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 77–118. Springer, Boston, MA (2015). https://doi.org/10.1007/978-1-4899-7637-6_3
24. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **8**, 30–37 (2009)
25. Liang, D., Charlin, L., McInerney, J., Blei, D.M.: Modeling user exposure in recommendation. In: Proceedings of the 25th International Conference on World Wide Web, WWW 2016, pp. 951–961 (2016)
26. Ludewig, M., Mauro, N., Latifi, S., Jannach, D.: Performance comparison of neural and non-neural approaches to session-based recommendation. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, pp. 462–466. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3298689.3347041
27. Lunceford, J.K., Davidian, M.: Stratification and weighting via the propensity score in estimation of causal treatment effects: a comparative study. Stat. Med. **23**(19), 2937–2960 (2004)
28. Luo, H., Niu, C., Shen, R., Ullrich, C.: A collaborative filtering framework based on both local user similarity and global user similarity. Mach. Learn. **72**(3), 231–245 (2008)
29. Ning, X., Desrosiers, C., Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 37–76. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6_2
30. Ning, X., Karypis, G.: SLIM: sparse linear methods for top-N recommender systems. In: 2011 IEEE 11th International Conference on Data Mining, pp. 497–506. IEEE (2011)
31. Oosterhuis, H., Jagerman, R., de Rijke, M.: Unbiased learning to rank: counterfactual and online approaches. In: Companion Proceedings of the Web Conference 2020, WWW 2020, pp. 299–300. Association for Computing Machinery, New York (2020). https://doi.org/10.1145/3366424.3383107
32. Radcliffe, N.J., Surry, P.D.: Real-world uplift modelling with significance-based uplift trees. White Paper TR-2011-1, Stochastic Solutions (2011)

33. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2009, pp. 452–461. AUAI Press, Arlington (2009)
34. Rubin, D.B.: Estimating causal effects of treatments in randomized and nonrandomized studies. J. Educ. Psychol. **66**(5), 688 (1974)
35. Saito, Y.: Doubly robust estimator for ranking metrics with post-click conversions. In: Fourteenth ACM Conference on Recommender Systems, RecSys 2020, pp. 92–100. Association for Computing Machinery, New York (2020). https://doi.org/10.1145/3383313.3412262
36. Saito, Y., Yaginuma, S., Nishino, Y., Sakata, H., Nakata, K.: Unbiased recommender learning from missing-not-at-random implicit feedback. In: Proceedings of the 13th International Conference on Web Search and Data Mining, WSDM 2020, pp. 501–509. Association for Computing Machinery, New York (2020). https://doi.org/10.1145/3336191.3371783
37. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, WWW 2001, pp. 285–295. Association for Computing Machinery, New York (2001). https://doi.org/10.1145/371920.372071
38. Sato, M., Izumo, H., Sonoda, T.: Modeling individual users' responsiveness to maximize recommendation impact. In: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, UMAP 2016, pp. 259–267. ACM, New York (2016). https://doi.org/10.1145/2930238.2930259
39. Sato, M., Singh, J., Takemori, S., Sonoda, T., Zhang, Q., Ohkuma, T.: Uplift-based evaluation and optimization of recommenders. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, pp. 296–304. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3298689.3347018
40. Sato, M., Takemori, S., Singh, J., Ohkuma, T.: Unbiased learning for the causal effect of recommendation. In: Fourteenth ACM Conference on Recommender Systems, RecSys 2020, pp. 378–387. Association for Computing Machinery, New York (2020). https://doi.org/10.1145/3383313.3412261
41. Schnabel, T., Swaminathan, A., Singh, A., Chandak, N., Joachims, T.: Recommendations as treatments: debiasing learning and evaluation. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning, ICML 2016, vol. 48. pp. 1670–1679. JMLR.org (2016)
42. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating "word of mouth". In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 1995, pp. 210–217. ACM Press/Addison-Wesley Publishing Co., New York (1995). https://doi.org/10.1145/223904.223931
43. Sharma, A., Hofman, J.M., Watts, D.J.: Estimating the causal impact of recommendation systems from observational data. In: Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC 2015, pp. 453–470. ACM, New York (2015). https://doi.org/10.1145/2764468.2764488
44. Stuart, E.A.: Matching methods for causal inference: a review and a look forward. Stat. Sci. **25**(1), 1–21 (2010). https://doi.org/10.1214/09-STS313
45. Swaminathan, A., Joachims, T.: The self-normalized estimator for counterfactual learning. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Quebec, Canada, 7–12 December 2015, pp. 3231–3239 (2015). http://papers.nips.cc/paper/5748-the-self-normalized-estimator-for-counterfactual-learning

46. Tintarev, N., Masthoff, J.: Over- and underestimation in different product domains. In: Ghallab, M., Spyropoulos, C., Fakotakis, N., Avouris, N. (eds.) Workshop on Recommender Systems, 18th European Conference on Artificial Intelligence (ECAI 2008), 21–25 July 2008. IOS Press (2008)
47. Tintarev, N., Masthoff, J.: Explaining recommendations: design and evaluation. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 353–382. Springer, Boston, MA (2015). https://doi.org/10.1007/978-1-4899-7637-6_10
48. Torkamaan, H., Barbu, C.M., Ziegler, J.: How can they know that? a study of factors affecting the creepiness of recommendations. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, pp. 423–427. Association for Computing Machinery, New York (2019). https://doi.org/10.1145/3298689.3346982
49. Tyler, J.V., Miguel, A.H., et al.: Causal inference under multiple versions of treatment. J. Causal Inference **1**(1), 1–20 (2013)
50. Wang, X., Zhang, R., Sun, Y., Qi, J.: Doubly robust joint learning for recommendation on data missing not at random. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 6638–6647. PMLR, Long Beach, 09–15 June 2019. http://proceedings.mlr.press/v97/wang19n.html
51. Wang, X., Bendersky, M., Metzler, D., Najork, M.: Learning to rank with selection bias in personal search. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2016, pp. 115–124. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2911451.2911537
52. Wang, Y., Blei, D.M.: The blessings of multiple causes. J. Am. Stat. Assoc. **114**(528), 1574–1596 (2019)
53. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**(1) (2019). https://doi.org/10.1145/3285029
54. Zhuang, S., Zuccon, G.: Counterfactual online learning to rank. In: Jose, J.M., et al. (eds.) ECIR 2020. LNCS, vol. 12035, pp. 415–430. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45439-5_28

User Engagement Prediction for Clarification in Search

Ivan Sekulić[1](✉), Mohammad Aliannejadi[2], and Fabio Crestani[1]

[1] Faculty of Informatics, Università della Svizzera italiana, Lugano, Switzerland
{ivan.sekulic,fabio.crestani}@usi.ch
[2] University of Amsterdam, Amsterdam, The Netherlands
m.aliannejadi@uva.nl

Abstract. Clarification is increasingly becoming a vital factor in various topics of information retrieval, such as conversational search and modern Web search engines. Prompting the user for clarification in a search session can be very beneficial to the system as the user's explicit feedback helps the system improve retrieval massively. However, it comes with a very high risk of frustrating the user in case the system fails in asking decent clarifying questions. Therefore, it is of great importance to determine *when* and *how* to ask for clarification.

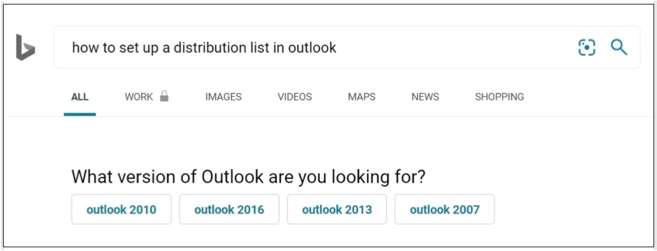
To this aim, in this work, we model search clarification prediction as user engagement problem. We assume that the better a clarification is, the higher user engagement with it would be. We propose a Transformer-based model to tackle the task. The comparison with competitive baselines on large-scale real-life clarification engagement data proves the effectiveness of our model. Also, we analyse the effect of all result page elements on the performance and find that, among others, the ranked list of the search engine leads to considerable improvements. Our extensive analysis of task-specific features guides future research.

Keywords: Search clarification · Mixed-initiative conversations · User engagement prediction

1 Introduction

The primary goal of an information retrieval (IR) system is satisfying the user information need, which can often be ambiguous when expressed as short queries. Incorporating users' implicit feedback has long been studied for improved retrieval [17]. However, the recent rise of interest in conversational systems and mixed-initiative interactions have enabled IR systems to collect users' explicit feedback. Current research focuses on prompting users for feedback by asking for clarification [2,32,43]. For example, search clarification has recently been utilised in search engines, leading to an improved user experience [43]. Another prominent area studying clarification is conversational search, as the system can usually output only one response, thus requiring to clarify the user's intent [2,32].

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 619–633, 2021.
https://doi.org/10.1007/978-3-030-72113-8_41



how to set up a distribution list in outlook

What version of Outlook are you looking for?

Fig. 1. An example of Bing clarification pane taken from [44].

The importance of clarification further increases in a mixed-initiative conversational setting [39], where control of the conversation goes back and forth between user and system through assertions, prompts, and questions [32].

However, clarification in search proved to be a cumbersome task [45], posing higher risk of user dissatisfaction. The challenge arises from two main aspects: deciding whether or not it is necessary to ask for clarification, and selecting or generating the appropriate clarifying question. Clarification selection can in fact be formalised as a user engagement prediction problem. User engagement refers to the quality of user experience characterised by, among others, attributes of positive affect, attention, interactivity, and perceived user control [26]. Persistent users' interactions with the clarification mechanism are an indication of a well-designed system. Furthermore, through these interactions users provide implicit feedback about the *necessity* and the *quality* of prompted clarifications.

Recently, modern search engines include various types of clarification components into their systems. An example of such a component in Bing, namely a clarification pane, can be seen on Fig. 1. Given a user query, a number of Microsoft's internal algorithms propose a clarifying question and offer clickable answers that would filter the retrieved results according to the user's need. The research on the quality of asked clarifying questions and potential answers is still in its early stages [43]; however, Zamani et al. [44] argued that engagement level could be an indicator of the clarification system quality. User engagement prediction has been studied in various domains of IR [25]. However, studying and modelling user engagement for web search clarification is relatively unstudied.

In this paper, we focus on the task of predicting user engagement level (ELP) on clarification panes. Given an initial query, search results, and clarification pane, ELP aims to estimate how engaged the user would be with the clarification pane. Previous work [45] studies how engagement levels correlate with the query attributes such as query type and aspects. However, the relationship between SERPs and engagement has not yet been explored. We stress the importance of utilising retrieved results, as they can contain cues as to how faceted or ambiguous the query is, suggesting how necessary the clarification is in the first place.

Moreover, users' engagement with the system implicitly discloses information about the *necessity* and the *quality* of the asked clarification. The *quality* aspect can be modelled under the assumption that the higher the engagement levels, the better the question and the provided answers are. We make this assumption inspired by a large body of work in the IR community on implicit feedback from aggregated click-through rates for document retrieval [42]. Also, we study clarification *necessity* prediction through ELP. Our clarification necessity prediction model takes as input the initial query and the retrieved results list and predicts the level of user engagement with a clarification pane. Although certain attributes of the initial query such as length and ambiguity could indicate the necessity of asking clarifying questions, we show that incorporating other SERP elements such as result titles and snippets play important roles in improved prediction accuracy.

We formulate the task as supervised regression and propose a deep learning-based model for the prediction of the engagement levels. We compare the performance of the model to various central tendency measures and a number of traditional machine learning algorithms, as well as popular neural models. Our model, based on a Transformer architecture, jointly encodes the user query, the clarification pane, and the SERP elements, outperforming competitive baselines. We evaluate the performance of our model on a large-scale dataset of search clarification engagements called MIMICS[1] [44], collected from millions of interaction records of Bing[2] users. Our extensive experiments establish a strong baseline for the task, while ablation studies and analysis of the model's inner mechanisms provide guidelines for future research. Our main contributions can be summarised as follows:

- We formally introduce the clarification pane ELP task as supervised regression and propose a transformer-based model to tackle it. We make the code publicly available for reproducibility purposes[3].
- We perform ablation studies with respect to the model input data. We find that utilising retrieved search results greatly benefits the model's performance.
- We perform detailed analysis of the performance of our model w.r.t. various characteristics of the SERP.

To the best of our knowledge, our work is the first to utilise SERP elements for clarification pane engagement prediction. More precisely, we find that utilising search results in certain ways is highly beneficial for the ELP task, as the performance of our model increases by up to 40% when provided with retrieved results, compared to the query and the clarification pane only.

2 Related Work

Our work is related to work done in conversational and web search clarification, engagement level prediction, and neural networks. In this section we briefly review some of the works in these areas.

[1] https://github.com/microsoft/MIMICS.

[2] http://www.bing.com.

[3] https://github.com/isekulic/mimics-EL-benchmark.

Clarification. Search clarification has recently been addressed as an important problem in the IR community. Recent research efforts study clarification in a wide range of areas, including web search engines [45], community question answering [6], voice queries [18], dialogue systems [38], entity disambiguation [9], and information-seeking conversations [2,20,36].

Radlinski and Craswell [32] discuss the need for clarification in their proposed theoretical framework for conversational search, highlighting the necessity of multi-turn interactions with users. Moreover, the report from the Dagstuhl Seminar on Conversational Search [4] summarises potential research topics in conversational search, and recognises clarification as an integral part of a conversational information seeking (CIS) system, which was also argued by Penha et al. [29] for information-need elucidation. Asking clarifying questions was studied by Aliannejadi et al. [2], who propose an offline evaluation setting of an open-domain CIS system, which was highlighted as a hard-to-evaluate setting [30]. They find that asking clarifying questions reduces the number of turns needed for identifying the underlying user information need. Adding the fact that users like to be prompted for clarification [18], we see a clear importance for clarification.

Clarification is further highlighted in mixed-initiative conversational search, where system in each turn needs to decide whether to ask for clarification or issue a response [32]. Hashemi et al. [15] propose a Guided Transformer model for document retrieval and next clarifying question selection in a conversational search setting. Zamani et al. [43] propose supervised and reinforcement learning models for generating clarifying questions and the corresponding candidate answers from weak supervision data. On the other hand, Ren et al. [34] introduce the task of conversations with search engines, where system generates a short, summarised response of the retrieved passages. Although generating and selecting clarifying questions for such purposes has recently been studied, the necessity of asking for clarification is still a relatively unexplored topic [1]. Whether or not it is necessary to ask for clarification depends mostly on the level of ambiguity of the query.

User Engagement. O'Brien and Toms [26] define user engagement as the quality of user experience in interaction with a system, characterised by various attributes, e.g., positive affect, aesthetic and sensory appeal, attention, novelty, perceived user control. In their recent study [25], they point user engagement as an important outcome measure in interactive IR research. User engagement has previously been studied in the context of commercial software, social media [13], online news [24], student engagement with online courses [12], and applications for monitoring health-related signals [3].

User engagement in the aforementioned studies has usually been measured by self-reported questionnaires, facial expression analysis or speech analysis, signal processing methods, or web analytics [21]. Recently, Zamani et al. [44] created a collection of datasets for studying clarification in search by aggregating user interactions with clarification pane in a major commercial search engine, thus falling into the category of measuring the user engagement by web analytics. In this paper however, instead of estimating the engagement levels with a goal of advancing search engine clarification feature, we analyse the implicit signals of the interactions that contain valuable information about the ambiguity of the query, diversity of retrieved results, and the quality of the clarifying question. Thus, motivated by work on implicit feedback of aggregated users' click-through logs for ad hoc retrieval [17], we view the engagement levels as implicit evaluation of clarifying questions with respect to the query and search results. Intuitively, the higher the engagement levels with the clarification system, the higher the quality of the prompted clarification, and higher the need for asking for clarification.

Zamani et al. [45] study the clarifying question selection with respect to user queries, prompted questions and candidate answers in clarification panes of a search engine. However, the retrieved search engine results for a query have not yet been studied. To bridge this gap, in this paper, we propose a model to predict the user engagement levels, not only from the information in clarification pane, but from the retrieved search results.

Transformers. The unprecedented success of the Transformer-based architectures in the large variety of the IR and natural language processing tasks motivated their application to the engagement level prediction task as well. One of the most prominent Transformer-based models is BERT [11]. BERT has reached state-of-the-art results in multiple language understanding benchmarks, such as GLUE [40] and SQuAD [33], as well as IR tasks, such as passage and document ranking [23,37]. In this work, we utilise ALBERT [22] – a lite BERT. ALBERT offers the performance of BERT, or even a higher one, while having fewer parameters, reducing the GPU/TPU memory requirements.

3 Engagement Level Prediction

In this section, we first describe the dataset used for engagement level prediction (ELP). Then, we formally introduce the task of ELP and propose a BERT-based model to tackle it.

3.1 Data

MIMICS [44] is a recently proposed large-scale collection of datasets for research on search clarification. It enables the IR community to study various aspects of search clarification, ranging from clarification generation and selection, over re-ranking of candidate answers, to user engagement prediction and click models for clarification. MIMICS consists of three datasets:

1. **MIMICS-Click**, including over 400k unique queries, their corresponding clarification panes, and the aggregated user interaction signals.
2. **MIMICS-ClickExplore**, consisting of over 60k unique queries, each with multiple clarification panes, and the aggregated interaction signals.

Table 1. Dataset statistics for MIMICS-Click.

	Mean	Std	Median	min-max
Query length	2.66	1.18	2	1 - 12
Question length	6.05	0.47	6	5 - 14
SERP Titles length	7.65	2.71	8	0 - 30
SERP Snippets length	43.47	14.76	45	0 - 149
Answers per query	2.81	1.06	2	2 - 5
Responses per query	9.07	1.19	9	0 - 10

3. **MIMICS-Manual**, containing 2k query-clarification pairs, manually labelled for the quality of clarifying questions, candidate answer sets, and landing result pages of each answer.

In this work, we mainly focus on MIMICS-Click, as the largest, most generic one. Each sample in MIMICS-Click consists of the initial query q, the clarification question c, and answers offered as options by the system $A = [a_1, ...a_5]$. The sample is associated with user interaction signals as labels. The *impression level* i, a categorical variable where $i \in \{low, medium, high\}$, represents the frequency of the clarification pane being presented to the user for the corresponding query. The *engagement level* $e \in [0, 10]$ shows the level of total engagement received by the users in terms of click-through rate. Each answer is also associated with its conditional click probability.

The authors also released search engine results pages (SERPs) for each query, as retrieved by Bing. In addition to the query meta-data, SERPs contain up to 10 retrieved instances with a title, an URL, and a short snippet of a web document. We denote retrieved results as $R = [r_1, r_2, \ldots r_n]$, where $n \in [0, 10]$. Each of the results r_i consists of a tuple $r_i = (t_i, s_i)$, where t_i and s_i are title and snippet of the i-th result. Table 1 shows the average lengths of queries[4], questions, retrieved titles and snippets, as well as the number of retrieved results in SERPs. We utilise all of the available text and information as input to our models to compose our experiments, as described in Sect. 3.3.

3.2 Task Formulation

We formulate the task of user engagement level prediction as a supervised regression. The goal of the regression is to predict the value of the target variable y, given a D-dimensional vector $\mathbf{x}$ of input variables [5]. Given the dataset of N observation pairs $(\mathbf{x}_n, y_n)$, where $n = 1, ..., N$, the goal is to find a function $f(\mathbf{x})$ whose outputs $\hat{y}$ for new inputs $\mathbf{x}$ produce the predictions for the corresponding values of y. The loss function of the predicted values $\hat{y}$ and the actual values y are model-dependent and described in Sect. 3.3.

[4] The length was computed by splitting the text on whitespaces.

The target variable y is given in the dataset in the range of 0 to 10, corresponding to the level of user engagement with the clarification pane. We approach ELP as a regression problem as it poses itself as a natural formulation of our task. Compared to classification, false predictions of different value are penalised differently. For example, classification would punish false predictions of $\hat{y} = 7$ and $\hat{y} = 1$ for a sample with $y = 8$ the same, while in reality, the predicted label of 7 is much closer to the actual engagement level. Therefore, even though still wrong, one would prefer a system to predict 7 instead of 1. Moreover, the task of user engagement prediction has been evaluated as regression in various applications such as [12,35].

3.3 Our Approach

We now define our model called **ELBERT** (**E**ngagement **L**evel prediction by A**LBERT**). As mentioned in the previous section, the goal is to predict the engagement level y based on the initial query q, clarification question c, list of candidate answers A, and retrieved results R. We predict the engagement level EL as follows:

$$EL(q, c, A, R) = \psi(\phi_q(q), \phi_c(c), \phi_A(A), \phi_R(R)) \tag{1}$$

where $\phi_{\{q,c,A,R\}}$ are high-dimensional representations of q, c, A, and R. The aggregation function ψ outputs the final engagement levels based on the input representations. All of these components can be modelled with numerous methods. In this work, we utilise ALBERT as our encoder for generating $\phi_{\{q,c,A,R\}}$ representations in a joint fashion. More specifically, as ALBERT has been shown to consistently help downstream tasks with multiple inputs [22], we essentially learn the joint representation of query, clarification question, answers, and results as:

$$\varPhi(q, c, A, R) = ALBERT(q, c, A, R) \tag{2}$$

reducing our Eq. 1 to:

$$EL(q, c, A, R) = \psi(\varPhi(q, c, A, R)). \tag{3}$$

Input to the ALBERT component is composed of tokenized query, question, answers, and results, separated by the separation token $[SEP]$, with classification token $[CLS]$ inserted in the beginning of a sequence. Answers a_i are aggregated before feeding them to the model. Similarly, we aggregate SERP information R, with a difference that we experiment with both, titles t_i and snippets s_i as inputs. In either case, texts of titles or of snippets are joined by whitespace prior to being fed to the model. We note that in ablation studies some of the components are left out by simply removing them from Eq. 2. We use a pretrained ALBERT-base [22] as a text encoder and truncate the total input sequence length to a maximum of 512 tokens. Our model has $11M$ training parameters, making it considerably smaller than other Transformer-based model (e.g., BERT has $110M$).

The regression component ψ, that outputs the engagement level, is constructed as follows: last layer hidden-state of the first token of the encoded sequence ($[CLS]$ token) is further processed by a linear layer and a non-linear activation function. We then add another linear layer, with dropout and a non-linear activation function in between, to produce the final 1-dimensional output that corresponds to EL. The model is trained using mean squared error as a loss function for 4 epochs, with a learning rate of 5×10^{-5}, Adam optimizer [19] and linear weight decay with warmup.

4 Experiments

In this section, we introduce our experimental setup and present main results for the engagement level prediction. Furthermore, we analyse the effect of SERP elements on model's performance and perform detailed analysis w.r.t. various characteristics of the data.

4.1 Baselines

We use central tendency measures as our first baselines for predicting the engagement level. More specifically, we have three different static baselines: (i) *mean* of the data (MeanEngagement); (ii) *median* of the data (MedianEngagement); (iii) *sampling* from a normal distribution $\mathcal{N}(\mu, \sigma^2)$, where μ and σ are the mean and the standard deviation of the engagement levels in the training data, respectively (NormalEngagement).

To tackle the task of ELP, we experiment with a number of models from traditional machine learning and deep learning. Namely:

Linear Regression. First baseline is a linear regression model, fitted using ordinary least squares approach.

SVR. We employ support vector regression machines [14], a version of support vector machines [10] for regression. We experiment with the linear, as well as the radial basis function (RBF) kernel.

Random Forests. An ensemble meta-algorithm that uses bootstrap aggregating (bagging) technique to improve the stability of decision trees [7].

LSTM. Long-short term memory [16] are a well-established method for sequence modelling, especially on text data. We experiment with multi-layer bidirectional networks.

The input to traditional ML models are tf-idf weighted bag-of-word features extracted from the input text. LSTM is fed with pretrained GloVe word embeddings [31] of tokenized input text. We use Scikit-learn [28], HuggingFace [41], and Pytorch [27] for the implementation of the aforementioned models.

4.2 Evaluation Metrics

We evaluate the effectiveness of our models using standard evaluation metrics for the task of supervised regression. The first two are Mean Absolute Error (MAE) and Mean Squared Error (MSE). We also evaluate our regression models with

Table 2. Performance on the full MIMICS-Click dataset (400k+ samples) and a subset where engagement levels are higher than zero (71k samples). Bold values denote the best results for each metric. Symbols † and ‡ mark statistically significant improvement over central tendency measures and traditional ML models, respectively ($p < 0.01$).

Model	Full MIMICS-Click			EL-only MIMICS-Click		
	MAE	MSE	R^2	MAE	MSE	R^2
Mean	0.1531	0.0546	0.0	0.2426	0.0790	0.0
Median	**0.0921**†	0.0531	0.0	0.2412	0.0805	0.0
Normal	0.1896	0.0823	0.0	0.4316	0.2637	0.0
Linear Regression	0.1463	0.0530	0.0359	0.2364	0.0783	0.0083
SVR	0.1462	0.0522	0.0490	0.2318†	0.0736†	0.0676†
RandomForest	0.1477	0.0526	0.0423	0.2301†	0.0729†	0.0775†
BiLSTM	0.1452†‡	0.0511†‡	0.0606†‡	0.2299†	0.0720†	0.0789†
ELBERT	0.1439†‡	**0.0505**†‡	**0.0762**†‡	**0.2224**†‡	**0.0692**†‡	**0.1124**†‡

Coefficient of Determination or R^2. It is a statistical measurement that examines the proportion of the variance in one variable that is predictable from the second variable, estimating the "goodness of a fit". It is defined as: $R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y}_i)^2}$, where N is the number of samples, y_i is the actual value in the dataset for the i-th sample, $\hat{y}$ is the predicted value, and $\overline{y}$ is the mean of the actual values.

4.3 Experimental Setup

We evaluate our models using a hold-out method, i.e., reserving 20% of the dataset for the test set. We train, and tune traditional ML models in a cross-validation manner [8]. We use 5-fold split of the training set into training and development set, which is used for grid-searching of the best parameters. The extensive grids of parameters include regularisation parameter C, the choice of *kernel*, *gamma*, and *epsilon* for SVR, number of estimators and depth of random forest regressor, as well as feature selection process. All of the parameters can be found on our GitHub repository.

For tuning the hyper-parameters of our neural models, we split the training set into training and development sets. Notice that models are retrained on the full training set with the best parameters before being evaluated on the hold-out test set.

We evaluate the models on the full MIMICS-Click dataset, consisting of more than 400k query-clarification-SERP tuples, and on the subset of that dataset, in which only samples with the engagement level larger than zero are selected. The models in this setting were fed all the available data, i.e., the queries, clarification panes, and the SERPs, while the ablation studies in Sect. 4.4 go into the analysis of input data.

Table 3. Impact of SERP elements available on the model performance. Bold values denote the best performance of each metric. Statistically significant results (with $p < 0.05$) over *query* setting and *query+pane* setting are marked with † and ‡, respectively.

#	Setting	Full MIMICS-Click			EL-only MIMICS-Click		
		MAE	MSE	R^2	MAE	MSE	R^2
1	query	0.1500	0.0519	0.0485	0.2275	0.0719	0.0776
2	query+pane	$0.1354^{\dagger}$	0.0512	$0.0626^{\dagger}$	$0.2257^{\dagger}$	0.0714	$0.0839^{\dagger}$
3	query+titles	**0.1335**†‡	**0.0436**†‡	**0.0814**†‡	$0.2229^{\dagger\ddagger}$	**0.0692**†‡	**0.1124**†‡
4	query+snippets	$0.1459^{\dagger}$	0.0513	$0.0606^{\dagger}$	$0.2255^{\dagger\ddagger}$	$0.0706^{\dagger\ddagger}$	$0.0944^{\dagger\ddagger}$
5	query+pane+titles	$0.1450^{\dagger}$	$0.0505^{\dagger}$	$0.0745^{\dagger\ddagger}$	**0.2224**†‡	**0.0692**†‡	**0.1124**†‡
6	query+pane+snippets	$0.1439^{\dagger}$	$0.0505^{\dagger}$	$0.0762^{\dagger\ddagger}$	$0.2240^{\dagger\ddagger}$	$0.0704^{\dagger\ddagger}$	$0.0969^{\dagger\ddagger}$

4.4 Results and Discussion

Performance Comparison. Here, we compare the performance of our ELBERT model against the baselines on the complete dataset, as well as the subset of data with EL > 0. Table 2 lists the results in terms of all our evaluation metrics. We can notice that heuristic baselines (i.e., MeanEngagement, MedianEngagement and NormalEngagement) are consistently outperformed by both, the traditional ML models, and the neural models. However, one exception is MedianEngagement, a baseline that always outputs the median of the training set, i.e., EL of 0.0, when evaluated on the full MIMICS-Click by mean absolute error. Since more than 80% of the dataset have EL of 0.0, and MAE does not penalise large errors as hard as MSE or R^2, this is expected. The tide turns swiftly when evaluating on the subset of the data with EL larger that 0.0, where all of the static baselines, including MedianEngagement, are outperformed by all of our models.

Moreover, we see a clear disparency in the performance of traditional ML models and neural networks. This is consistent with recent research in various tasks in IR and NLP fields. Moreover, we see that ELBERT significantly outperforms BiLSTM model. Through its powerful encoder, ELBERT is able to capture deeper semantic relations, as it is pretrained on a large body of text. This is also consistent with recent research on deep learning-based models for natural language understanding.

Effect of SERP Elements on ELP. In this experiment, we aim to analyse the effect of clarification panes and every SERP element on the performance of our model. Our hypothesis is that each SERP element (e.g., result titles and snippets) provides a complementary set of features that aids the model towards more effective prediction. Therefore, we train our ELBERT model with different combination of SERP elements and clarification panes, and compare the performance of the different models. We report the results in Table 3. We see that the relative improvement when utilising titles from SERPs is up to 35%

compared to using query and clarification pane, and more than 45% over query-only setting. The results strongly suggest the advantage of making use of SERP elements for ELP.

An interesting finding is that even though snippets contain more text than titles and thus arguably more information as well, the model does not consistently perform better with snippets as input. In fact, even though results with titles seem better than ones with snippets, we observe no statistically significant difference between the performance of *query+titles* and *query+snippets* on full MIMICS-Click, nor EL-only MIMICS-Click. There are several reasons why snippets do not exceed the performance of titles. First, it might be the quality and type of text shown in snippets. Snippets often show only short excerpts, or even multiple excerpts which are not clearly divided, from a longer document, focusing on query words in the retrieved document. Thus, they might not contain all the semantics of the document, while titles usually do. Second, it might be the maximum input length of our encoder, which is 512 sub-word tokens. As mentioned in Table 1, a median length of a title is 8 tokens, while median snippet length is 45. Considering that most of the samples have 9 or more title-snippet pairs in their SERPs, it is evident that some portion of concatenated snippets get left out. The potential limitation of truncating input length in most of BERT-based models is a research direction on its own.

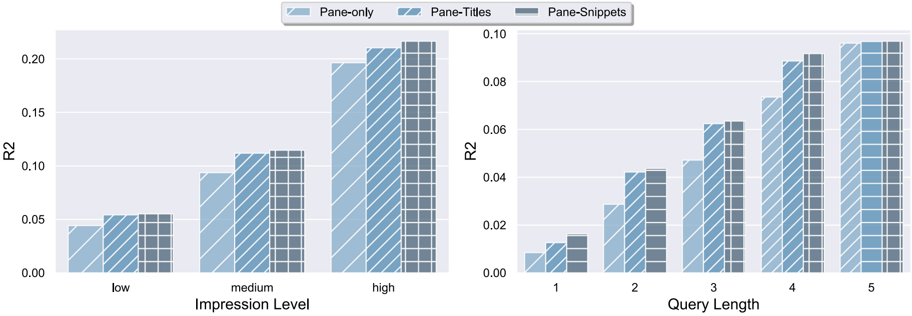


Fig. 2. Performance by impression levels (left) and query lengths (right) with different input configurations.

We point out that the necessity of asking the clarification can be estimated from the initial query and retrieved search results, i.e., rows 1, 3, and 4 in Table 3, The success of the model to predict EL based on SERPs and the query alone, suggests that this framework can be used for determining whether or not to ask a clarifying question. However, we leave this aspect for future work. Instead, in the next subsection we evaluate our model trained on ELP task for clarification pane selection, addressing the pane quality aspect.

4.5 Additional Experiments

Here we show ELBERT performance, as measured by R^2, with respect to various characteristics of the dataset and the input components.

Impression Level. Figure 2 (left) shows the performance of our model w.r.t. impression levels. We notice that our model performs significantly better on queries with high impression rate, i.e., those whose clarification panes have been shown to users more frequently. The differences between models at each impression level are not statistically significant, while differences between levels are, with $p < 0.01$. As the engagement level labels have been computed by aggregating user click information, this suggest that query-clarification pairs that have been implicitly evaluated by a small number of users, i.e., have low impression level, contain noise.

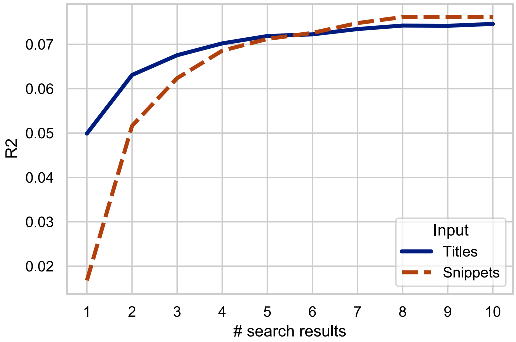


Fig. 3. Performance by number of search results made available to the model.

Query Length. Figure 2 (right) presents the performance of our model w.r.t. query length. The difference in performance between all query lengths is statistically significant. We notice that longer queries generally lead to better performance. This can be attributed to them being more descriptive, thus allowing the search engine to retrieve more relevant results. Consequently, our model would utilise SERPs of higher quality, improving the ELP. Highest improvement is seen for a query and pane-only setting. Since the model in that setting does not see any SERP content, it benefits the most out of longer, more descriptive queries.

Number of Search Results. Since user behaviour is mainly biased by the results they see, and they mostly look at top results only, we perform experiments to see how our models behave in a setting with limited number of retrieved results. As mentioned before, MIMICS dataset contains up to 10 retrieved results for each query. We evaluate our model with $1, 2, \ldots 10$ SERP elements made available to it. Results for both, titles setting and snippets setting, are presented in Fig. 3. We see a clear improvement in the performance as the number of search results fed to the model rises. This suggests that our model highly utilises SERP elements for ELP. We notice a saturation after 7 elements, especially in the setting with snippets. This might be due to snippets exceeding the maximum length of input to transformer-based models, which is 512 subword tokens.

5 Conclusions

In this study, we conducted various experiments on engagement level prediction task for clarification in search. We showed that semantic-rich models, like ALBERT, are much more successful in the task than traditional ML models. Furthermore, we demonstrated the benefit of utilising information from search engine result pages, such as titles and text snippets of retrieved documents, in the ELP task. Modelling of engagement levels can help guide the system on when and which clarifications to prompt, thus improving the overall user experience. Future work involves deeper analysis of topical changes in the retrieved pages, that could lead to more accurate prediction of engagement levels, and estimating the necessity of asking for clarification.

References

1. Aliannejadi, M., Kiseleva, J., Chuklin, A., Dalton, J., Burtsev, M.: ConvAI3: Generating clarifying questions for open-domain dialogue systems (ClariQ) (2020)
2. Aliannejadi, M., Zamani, H., Crestani, F., Croft, W.B.: Asking clarifying questions in open-domain information-seeking conversations. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 475–484 (2019)
3. Alkhaldi, G., Hamilton, F.L., Lau, R., Webster, R., Michie, S., Murray, E.: The effectiveness of prompts to promote engagement with digital interventions: a systematic review. J. Med. Internet Res. **18**(1), e6 (2016)
4. Anand, A., Cavedon, L., Joho, H., Sanderson, M., Stein, B.: Conversational search (Dagstuhl Seminar 19461). In: Dagstuhl Reports, vol. 9. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)
5. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
6. Braslavski, P., Savenkov, D., Agichtein, E., Dubatovka, A.: What do you mean exactly? Analyzing clarification questions in CQA. In: Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval, pp. 345–348 (2017)
7. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
8. Cawley, G.C., Talbot, N.L.: On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. **11**, 2079–2107 (2010)
9. Coden, A., Gruhl, D., Lewis, N., Mendes, P.N.: Did you mean A or B? Supporting clarification dialog for entity disambiguation. In: SumPre-HSWI@ ESWC (2015)
10. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)

11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1423, https://www.aclweb.org/anthology/N19-1423. (Long and Short Papers)
12. Dhall, A., Kaur, A., Goecke, R., Gedeon, T.: Emotiw 2018: audio-video, student engagement and group-level affect prediction. In: Proceedings of the 20th ACM International Conference on Multimodal Interaction, pp. 653–656 (2018)
13. Di Gangi, P.M., Wasko, M.M.: Social media engagement theory: exploring the influence of user engagement on social media usage. J. Organ. End User Comput. (JOEUC) **28**(2), 53–73 (2016)
14. Drucker, H., Burges, C.J., Kaufman, L., Smola, A.J., Vapnik, V.: Support vector regression machines. In: Advances in Neural Information Processing Systems, pp. 155–161 (1997)
15. Hashemi, H., Zamani, H., Croft, W.B.: Guided transformer: leveraging multiple external sources for representation learning in conversational search. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1131–1140 (2020)
16. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
17. Kelly, D., Teevan, J.: Implicit feedback for inferring user preference a bibliography. ACM SIGIR Forum, pp. 18–28. ACM, New York (2003)
18. Kiesel, J., Bahrami, A., Stein, B., Anand, A., Hagen, M.: Toward voice query clarification. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 1257–1260 (2018)
19. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv preprint: arXiv:1412.6980
20. Krasakis, A.M., Aliannejadi, M., Voskarides, N., Kanoulas, E.: Analysing the effect of clarifying questions on document ranking in conversational search. In: Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval, pp. 129–132 (2020)
21. Lalmas, M., O'Brien, H., Yom-Tov, E.: Measuring user engagement. Synth. Lect. Inform. Concepts Retr. Serv. **6**(4), 1–132 (2014)
22. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: a lite BERT for self-supervised learning of language representations. In: Proceedings of ICLR (2020)
23. Nogueira, R., Cho, K.: Passage re-ranking with BERT (2019). arXiv preprint: arXiv:1901.04085
24. O'Brien, H.L.: Antecedents and learning outcomes of online news engagement. J. Assoc. Inform. Sci. Technol. **68**(12), 2809–2820 (2017)
25. O'Brien, H.L., Arguello, J., Capra, R.: An empirical study of interest, task complexity, and search behavior on user engagement. Inform. Process. Manage. **57**(3), 102226 (2020)
26. O'Brien, H.L., Toms, E.G.: What is user engagement? A conceptual framework for defining user engagement with technology. J. Am. Soc. Inform. Sci.Technol. **59**(6), 938–955 (2008)
27. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, pp. 8026–8037 (2019)

28. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
29. Penha, G., Balan, A., Hauff, C.: Introducing MANtIS: a novel multi-domain information seeking dialogues dataset (2019). arXiv preprint: arXiv:1912.04639
30. Penha, G., Hauff, C.: Challenges in the evaluation of conversational search systems. In: KDD Workshop on Conversational Systems Towards Mainstream Adoption (2020)
31. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
32. Radlinski, F., Craswell, N.: A theoretical framework for conversational search. In: Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval, pp. 117–126 (2017)
33. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2383–2392. Association for Computational Linguistics, Austin, Texas (Nov 2016)
34. Ren, P., Chen, Z., Ren, Z., Kanoulas, E., Monz, C., De Rijke, M.: Conversations with search engines. ACM Trans. Inform. Syst. **1**(1), 1–19 (2020)
35. Sano, S., Kaji, N., Sassano, M.: Prediction of prospective user engagement with intelligent assistants. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Vol. 1, pp. 1203–1212 (2016). (Long Papers)
36. Sekulić, I., Aliannejadi, M., Crestani, F.: Extending the use of previous relevant utterances for response ranking in conversational search. In: Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC (2020)
37. Sekulić, I., Soleimani, A., Aliannejadi, M., Crestani, F.: Longformer for MS MARCO document re-ranking task (2020). arXiv preprint: arXiv:2009.09392
38. Stoyanchev, S., Liu, A., Hirschberg, J.: Towards natural clarification questions in dialogue systems. In: AISB Symposium on Questions, Discourse and Dialogue, Vol. 20 (2014)
39. Walker, M.A., Whittaker, S.: Mixed initiative in dialogue: an investigation into discourse segmentation. In: ACL (1990)
40. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.: Glue: a multitask benchmark and analysis platform for natural language understanding. In: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 353–355 (2018)
41. Wolf, T., et al.: HuggingFace's transformers: state-of-the-art natural language processing, pp. arXiv-1910 (2019)
42. Xue, G.R., et al.: Optimizing web search using web click-through data. In: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, pp. 118–126 (2004)
43. Zamani, H., Dumais, S., Craswell, N., Bennett, P., Lueck, G.: Generating clarifying questions for information retrieval. Proc. Web Conf. **2020**, 418–428 (2020)
44. Zamani, H., Lueck, G., Chen, E., Quispe, R., Luu, F., Craswell, N.: Mimics: a large-scale data collection for search clarification. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 3189–3196 (2020)
45. Zamani, H., et al.: Analyzing and learning from user interactions for search clarification (2020). arXiv preprint: arXiv:2006.00166

Sentiment-Oriented Metric Learning for Text-to-Image Retrieval

Quoc-Tuan Truong(✉) and Hady W. Lauw

Singapore Management University, Singapore, Singapore
{qttruong.2017,hadywlauw}@smu.edu.sg

Abstract. In this era of multimedia Web, text-to-image retrieval is a critical function of search engines and visually-oriented online platforms. Traditionally, the task primarily deals with matching a text query with the most relevant images available in the corpus. To an increasing extent, the Web also features visual expressions of preferences, imbuing images with sentiments that express those preferences. Cases in point include photos in online reviews as well as social media. In this work, we study the effects of sentiment information on text-to-image retrieval. Particularly, we present two approaches for incorporating sentiment orientation into metric learning for cross-modal retrieval. Each model emphasizes a hypothesis on how positive and negative sentiment vectors may be aligned in the metric space that also includes text and visual vectors. Comprehensive experiments and analyses on *Visual Sentiment Ontology (VSO)* and *Yelp.com* online reviews datasets show that our models significantly boost the retrieval performance as compared to various sentiment-insensitive baselines.

Keywords: Text-to-image retrieval · Cross-modal retrieval · Metric learning · Sentiment orientation

1 Introduction

The Web is awash in visual imagery. Millions of images are added daily to the billions already existing in various image-oriented platforms such as Instagram, Pinterest, Flickr, etc. In addition, product reviews in virtually any category, be it of restaurants on Yelp or consumer electronics on Amazon, frequently feature photos accompanying (complementing and even enhancing) the textual content of the reviews. In the face of such abundance and diversity, finding images relevant to one's purpose remains a pertinent challenge. While images are now a cornerstone modality on the Web, the manner in which most users express their intent is still predominantly textual. In this paper, we focus on text-to-image retrieval, i.e., retrieving images from a textual query. This is distinct from image retrieval, i.e., retrieving images from an image query [10], which is an active research topic in its own right.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 634–649, 2021.
https://doi.org/10.1007/978-3-030-72113-8_42

The presumption by many previous works on cross-modal retrieval (involving multiple modalities, such as text and image) [8,41] is that queries, and by extension the images the queries are aimed at, are generally of an objective nature. For instance, a user may be looking for pictures of a cat, a car, a specific person, etc. In reality, images are not universally devoid of sentiment. To the contrary, recent literature on visual sentiment analysis [31,36–38,47] attests to the manifestation of sentiments within some images. Within reviews for a restaurant or a hotel for example, someone may post an image of "restroom" in the positive sense (perhaps an especially clean or well-appointed specimen) or in the negative sense (such as the case where hygiene is less than desired). Conceivably, an "objective" query may turn out images of varied sentiments, due to its lack of specificity of which sentiment is fit and proper for the occasion at hand.

Problem. For a more holistic and expressive capacity for retrieving relevant images, we posit that in some scenarios the query intent may indeed have a sentiment dimension. For simplicity of discourse, we assume binary sentiment classes of *positive* and *negative* respectively. In other words, a query is now a tuple of *(textual keywords, sentiment class)*, and we seek to return a ranked list of images (from a corpus), which are relevant to the specified keywords *and* sentiment. It is worth noting that the corpus of interest consists of mere images, unadorned explicitly with text nor sentiment.

There are several challenges to this problem. One challenge inherent to cross-modality learning is how to learn associations among different modalities with distinct feature spaces, in this case text and images. Another challenge pertinent to retrieval is how to model relevance between varied modalities. Over and above these that plague cross-modal retrieval, we also have the peculiar challenge of modeling the third modality of interest, namely sentiment.

Approach. To deal with these challenges, we propose a framework called *Sentiment-Oriented Metric Learning* or *SML*. To overcome the variety in modalities, we learn modality-specific feature mappings that respectively map text and image inputs onto a common space. Presuming training data in the form of *text-sentiment-image* triples, we preserve relevant associations in these triples through proximity constraints relating texts, sentiments, as well as images in the resulting common feature space. Of particular interest are the manners in which we model sentiments as directional vectors in the common metric space, giving rise to two variants, SML_{OPPO} and SML_{FLEX}, based on different assumptions in bringing sentiment-oriented queries closer to the relevant images.

Contributions. In this work, we make several contributions. First, to our best awareness, this is the first work to study the effect of sentiment information for better understanding of text-to-image retrieval. Second, to characterize the effect of sentiments, we develop two models, namely SML_{OPPO} and SML_{FLEX}, that learn metric spaces in which the sentiments are represented by directional vectors. Third, we conduct comprehensive experiments comparing the proposed models with other cross-modal retrieval approaches. Experiments on real-life datasets, which include *Visual Sentiment Ontology* (Flickr images) and online review images from *Yelp.com*, show that our models significantly outperform the sentiment-insensitive baselines, underlining the import of sentiment on text-to-image retrieval.

2 Related Work

In this section, we review the related work along the two broad lines of metric learning as well as multi-view learning.

Metric Space Learning. The notion of distance is fundamental to many machine learning algorithms. Metric representation learning [22] deals with learning representations of objects so as to reflect the relationships among those objects in terms of distances in the metric space, i.e., putting relevant objects in proximity while distancing irrelevant ones. It finds applications in various contexts, such as image classification [26,35], image retrieval [21,40], text retrieval [46] and collaborative filtering [15], whereby in each case context-specific constraints may apply.

In the context of cross-modal retrieval, the constraints may include minimizing distances between positive pairs while maximizing distances between negative pairs [24,25]. Additional considerations may include preserving geometric structures such as global consistency and local smoothness [48] or making the feature learning modality-specific [42,49]. Orthogonally, we investigate metric learning for sentiment-oriented text-to-image retrieval, whereby the sentiment-orientation is particularly novel. We further propose a framework incorporating recent developments in deep representation learning, with new objectives to factor sentiment into the learned metric space (in addition to text and image modalities).

Multi-view Learning. An object may have multiple "views", i.e., observations in distinct feature spaces. In cross-modal retrieval, we have text and images. Multi-view learning finds object representations across several views, which would preserve the associations among different views of an object as well as among objects within a view.

A classical technique for feature learning across spaces is Canonical Correlation Analysis (CCA) [2,14]. The crux is to find linear projections of two vectors (one for each view), so as to maximize their correlations. To incorporate non-linearity, one approach is based on kernel methods [1,5,11,27]. A more recent approach is to use deep neural networks [3,16,23], of which DCCA [3] is the most recent work presenting a complete learning framework. In experiments, we compare to both CCA and DCCA.

Aside from correlation analysis, neural networks are also used for multi-view learning in different ways. Within the autoencoder framework, the objective is usually to find a feature representation in a common space that could reconstruct the inputs in the respective feature spaces [8,29,43]. In turn, [30,41] employ adversarial learning framework. As a recent competitive method for cross-modal retrieval based on adversarial learning, ACMR [41] is included as a baseline.

Note that ours has a different problem setting from those [17,32,41] that learn discriminative common representations by exploiting labels to distinguish between semantic categories. For one, sentiment can be seen as an independent modality, rather than labels during learning. For another, sentiment itself is a part of the query. Also incidentally related are approaches based on cross-modal hashing [7,33,34,45,50] that focus primarily on retrieval efficiency, while tolerating some loss in accuracy due to potential loss of information.

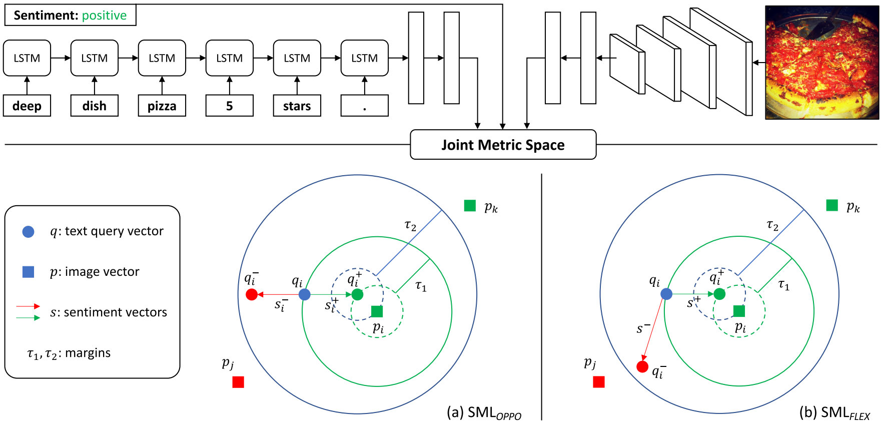


Fig. 1. Illustration of the SML framework. Image and text are embedded into the metric space using deep neural networks. For **SML**$_{OPPO}$ (a), sentiment vectors are in opposite directions, while sentiment vectors in **SML**$_{FLEX}$ (b) are unconstrained. Given that the query is *positive*, the sentiment margin constraint, $d(\mathbf{q}_i^+, \mathbf{p}_i) < d(\mathbf{q}_i, \mathbf{p}_i) - \tau_1$, is demonstrated in green color (*negative* is in red color). In turn, the distance margin constraint between correct and incorrect query-photo pairs, $d(\mathbf{q}_i^+, \mathbf{p}_i) < d(\mathbf{q}_i^+, \mathbf{p}_j) - \tau_2$, is demonstrated in blue color. (Color figure online)

3 Sentiment-Oriented Metric Learning (SML)

An input data collection $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{z}_i, \mathbf{y}_i)\}_{i=1}^N$ contains N instances of *text-sentiment-image* triples. Here, $\mathbf{z}_i$ is binary $\{$*positive*, *negative*$\}$. Our objective is to infuse the text with sentiment in order to form a sentiment-sensitive query $(\mathbf{x}_i, \mathbf{z}_i)$ that would better match the desired image $\mathbf{y}_i$ than $\mathbf{x}_i$ could on its own.

In essence, we propose SML framework which seeks to find two functions f and g transforming queries and images, respectively, into a metric space in which their similarities can be measured. Specifically, $f_{\boldsymbol{\theta}}$ and $g_{\boldsymbol{\psi}}$, parameterized by $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$, independently map $(\mathbf{x}_i, \mathbf{z}_i)$ and $\mathbf{y}_i$ to a D-dimensional Euclidean space $\mathbb{R}^D$, in which the distance between query $(\mathbf{x}_i, \mathbf{z}_i)$ and image $\mathbf{y}_i$ is measured as:

$$d_{\boldsymbol{\theta},\boldsymbol{\psi}}((\mathbf{x}_i, \mathbf{z}_i), \mathbf{y}_i) = \|f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{z}_i) - g_{\boldsymbol{\psi}}(\mathbf{y}_i)\|_2 \quad (1)$$

In this framework, we posit that sentiments are high-level abstraction concepts which should be represented as independent vectors in the metric space. We model a sentiment-infused query in additive form $f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{z}_i) = \mathbf{q}_i + \mathbf{s}_i$, where $\mathbf{q}_i$ and $\mathbf{s}_i$ are vectors in the metric space representing text and sentiment respectively. In turn, $g_{\boldsymbol{\psi}}(\mathbf{y}_i) = \mathbf{p}_i$ is a vector representing the image in the same metric space. The specific instantiation of $\mathbf{s}_i$ manifests slightly differently in two models, SML$_{OPPO}$ and SML$_{FLEX}$, which will be discussed subsequently. The learning output consists of transformations for $\mathbf{q}_i$ and $\mathbf{p}_i$, as well as the sentiment vectors $\{\mathbf{s}_i\}$ that allow us to measure distances for new queries and images.

3.1 Opposing Sentiment Vectors (SML$_{OPPO}$)

In the first model, referred to as SML$_{OPPO}$, we propose learning opposing sentiment vectors in a metric space. In other words, the two sentiment vectors (*positive* and *negative*) are in opposite directions and having the same magnitude. Thus, we only need to learn a single vector $\mathbf{s}$. It follows that the positive vector is $+\mathbf{s}$ and the negative vector is $-\mathbf{s}$. For each query tuple $(\mathbf{x}_i, \mathbf{z}_i)$, the sentiment vector $\mathbf{s}_i$ is in the form of:

$$\mathbf{s}_i = \alpha_i * \Gamma(\mathbf{z}_i) * \mathbf{s} \tag{2}$$

$$\alpha_i = \ln(1 + \exp(\mathbf{W}_\alpha^T \mathbf{q}_i)) \tag{3}$$

$$\Gamma(\mathbf{z}_i) = \begin{cases} +1 & \text{if } \mathbf{z}_i = \textit{positive} \\ -1 & \text{if } \mathbf{z}_i = \textit{negative} \end{cases} \tag{4}$$

where $\mathbf{s}$ is the sentiment basis vector shared across queries, $\Gamma(*)$ is a sign function, α_i is query-specific scale factor controlling the magnitude of the sentiment vector $\mathbf{s}_i$. Hypothetically, α_i is a function of $\mathbf{q}_i$ as different semantic concepts in different text queries require different intensity for the sentiment to be expressed. The choice of *softplus* [9] function for α_i is because of its smoothness and to ensure the value domain $\alpha_i \in (0, +\infty)$ for vector magnitude.

Our model learning can be specified as a *constrained* optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\theta}, \boldsymbol{\psi}, \mathbf{W}_\alpha, \mathbf{s}} \quad & \lambda(r(\boldsymbol{\theta}) + r(\boldsymbol{\psi})) + \sum_{i=1}^{N} d(\mathbf{q}_i + \mathbf{s}_i, \mathbf{p}_i) \\ \text{s.t.} \quad & d(\mathbf{q}_i + \mathbf{s}_i, \mathbf{p}_i) < d(\mathbf{q}_i, \mathbf{p}_i) - \tau_1 \\ & d(\mathbf{q}_i + \mathbf{s}_i, \mathbf{p}_i) < d(\mathbf{q}_i + \mathbf{s}_i, \mathbf{p}_j) - \tau_2, \forall j \neq i \end{aligned} \tag{5}$$

where $r(*)$ is regularizer on the model parameters $\{\boldsymbol{\theta}, \boldsymbol{\psi}\}$, $d(*)$ is the loss due to Euclidean distance, and λ is the trade-off between regularizer and loss. The first constraint is margined relative distance between sentiment-oriented-query and neutral-query towards the correct image. The second constraint is margined relative distance between correct and incorrect query-photo pairs. The relationships amongst vectors and constraints are demonstrated in Fig. 1a.

We transform this constrained optimization into a regularized empirical risk minimization problem. The constraints are enforced using the standard hinge loss $[\delta]_+ = \max(0, \delta)$. We then derive an unconstrained loss function with $l2$-regularization as follows:

$$\begin{aligned}\mathcal{L} = \; & \lambda\Big(\|\mathbf{W}_\alpha\|_F^2 + \sum_{l=1}^{L_f}(\|\mathbf{W}_f^l\|_F^2 + \|\mathbf{b}_f^l\|_2^2) + \sum_{l=1}^{L_g}(\|\mathbf{W}_g^l\|_F^2 + \|\mathbf{b}_g^l\|_2^2)\Big) \\ & + \sum_{i=1}^{N}\Big[\|(\mathbf{q}_i + \mathbf{s}_i) - \mathbf{p}_i\|_2^2 + \max\big(0, \tau + \|(\mathbf{q}_i + \mathbf{s}_i) - \mathbf{p}_i\|_2^2 - \|\mathbf{q}_i - \mathbf{p}_i\|_2^2\big) \\ & \quad + \sum_{j=1}^{N}\mathbb{1}(i \neq j)\,\max\big(0, 1 + \|(\mathbf{q}_i + \mathbf{s}_i) - \mathbf{p}_i\|_2^2 - \|\mathbf{q}_i - \mathbf{p}_j\|_2^2\big)\Big] \end{aligned} \tag{6}$$

where L_f and L_g are the numbers of layers of the two neural networks characterizing $f_{\boldsymbol{\theta}}$ and $g_{\boldsymbol{\psi}}$, respectively.

Parameters of the model can be optimized via minimizing the loss function using stochastic gradient descent. In practice, we optimize the model using mini-batch to speed up the learning process. For each mini-batch of triples $\mathcal{B} = \{(\mathbf{x}_i, \mathbf{z}_i, \mathbf{y}_i)\}_{i=1}^{|\mathcal{B}|}$ sampled from the collection $\mathcal{T}$, each query will be paired with other images within the mini-batch to form negative pairs instead of considering all possible negative combinations from the whole collection $\mathcal{T}$. This stochastic process drastically reduces convergence time, and in expectation achieves our global objective (Eq. 6). Algorithm 1 describes the optimization procedure with the mini-batch gradient descent.

3.2 Flexible Sentiment Vectors (SML$_{FLEX}$)

In some ways, the previous assumption by SML$_{OPPO}$ could be quite restrictive, as the opposing directions of the sentiment vectors are enforced on every single dimension of the learned metric space.

To allow for greater flexibility, we arrive at another variant, which we refer to as SML$_{FLEX}$, by allowing the positive sentiment vector and negative sentiment vector to take their own independent directions. That way, they can be opposing in some dimensions, but not necessarily across all D dimensions. Thus, it provides another degree of freedom for the model to allocate coordinates judiciously between the objective of capturing sentimental concepts as well as that of capturing textual-visual semantic concepts. SML$_{FLEX}$ decouples and learns two global sentiment vectors $\mathbf{s}^+$ and $\mathbf{s}^-$ separately. Figure 1b illustrates the learned metric space of SML$_{FLEX}$. The constrained optimization is as follows:

$$\begin{aligned} \min_{\boldsymbol{\theta}, \boldsymbol{\psi}, \mathbf{s}^+, \mathbf{s}^-} \quad & \lambda(r(\boldsymbol{\theta}) + r(\boldsymbol{\psi})) + \sum_{i=1}^{N} d(\mathbf{q}_i + \mathbf{s}^{\Gamma(\mathbf{z}_i)}, \mathbf{p}_i) \\ \text{s.t.} \quad & d(\mathbf{q}_i + \mathbf{s}^{\Gamma(\mathbf{z}_i)}, \mathbf{p}_i) < d(\mathbf{q}_i, \mathbf{p}_i) - \tau_1 \\ & d(\mathbf{q}_i + \mathbf{s}^{\Gamma(\mathbf{z}_i)}, \mathbf{p}_i) < d(\mathbf{q}_i + \mathbf{s}^{\Gamma(\mathbf{z}_i)}, \mathbf{p}_j) - \tau_2, \forall j \neq i \end{aligned} \tag{7}$$

Similarly to SML$_{OPPO}$, we can derive an unconstrained loss function and proceed minimization with the stochastic gradient descent algorithm.

3.3 Implementation Details

In this work, we use two neural networks, recurrent and convolution, to learn text and image transformations. The former uses LSTM cell [13], which had been shown to be effective in learning textual representation in many machine learning tasks. Word embeddings to the LSTM are initialized from pre-trained Word2vec [28] of 300 dimensions. For the latter, we employ ResNet-50 [12] architecture, which has also been used extensively for obtaining image representation of numerous vision-related tasks. The output representations from LSTM and ResNet-50 are both projected into the metric space using two-layer perceptrons (each layer is followed by the *hyperbolic tangent* activation function). The implementation of SML is made available at https://code.preferred.ai/sml/.

Algorithm 1. Parameter learning with mini-batch gradient descent

```
Input: $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{z}_i, \mathbf{y}_i)\}_{i=1}^{N}$, learning rate $\eta$
Output: Learned parameters $\{\boldsymbol{\theta}, \boldsymbol{\psi}, \mathbf{s}\}$

initialization
    $\boldsymbol{\theta}, \boldsymbol{\psi}, \mathbf{s} \leftarrow$ randomly initialized
while not converged do
    $\mathcal{T}_{batch} = \{\mathcal{B}_b\}_{b=1}^{num\_batch} \leftarrow$ uniformly sampled from $\mathcal{T}$
    for all $\mathcal{B}_b \in \mathcal{T}_{batch}$ do
        $g\boldsymbol{\theta} = 0;\ g\boldsymbol{\psi} = 0;\ g\mathbf{s} = 0;$
        for all $(\mathbf{x}_i, \mathbf{z}_i, \mathbf{y}_i) \in \mathcal{B}_b$ do
            for all $(\mathbf{x}_j, \mathbf{z}_j, \mathbf{y}_j) \in \mathcal{B}_b$ where $(j \neq i)$ do
                $g\boldsymbol{\theta} = g\boldsymbol{\theta} + \frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}(\mathbf{x}_i, \mathbf{s}_i, \mathbf{y}_i, \mathbf{y}_j);$
                $g\boldsymbol{\psi} = g\boldsymbol{\psi} + \frac{\partial}{\partial \boldsymbol{\psi}} \mathcal{L}(\mathbf{x}_i, \mathbf{s}_i, \mathbf{y}_i, \mathbf{y}_j);$
                $g\mathbf{s} = g\mathbf{s} + \frac{\partial}{\partial \mathbf{s}} \mathcal{L}(\mathbf{x}_i, \mathbf{s}_i, \mathbf{y}_i, \mathbf{y}_j);$
            end for
        end for
        $\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \cdot \frac{g\boldsymbol{\theta}}{|\mathcal{B}_b|};\ \boldsymbol{\psi} = \boldsymbol{\psi} - \eta \cdot \frac{g\boldsymbol{\psi}}{|\mathcal{B}_b|};\ \mathbf{s} = \mathbf{s} - \eta \cdot \frac{g\mathbf{s}}{|\mathcal{B}_b|};$
    end for
end while
return $\{\boldsymbol{\theta}, \boldsymbol{\psi}, \mathbf{s}\}$
```

4 Experiments

The objectives are to investigate the impact of sentiment on text-to-image retrieval and to assess the efficacy of sentiment-oriented metric learning framework via comparison with various cross-modal retrieval baselines.

4.1 Experimental Setup

Datasets. We conduct experiments on two datasets including *Visual Sentiment Ontology (VSO)* [6] and online reviews crawled from *Yelp.com*.

VSO dataset consists of adjective-noun pairs (ANP), e.g., *delicious drink* or *angry face*, associated with sentiment scores. Images are retrieved from Flickr when using these ANPs as queries. Firstly, sentiment is binarized based on the sign of the scores. Secondly, to reduce sentiment biases, we neutralize the queries by only using the nouns. Images from all ANPs belonging to the same noun are merged together. To remove the biases, we balance the number of images between two sentiments within each query via uniform sampling. These would then form $(\mathbf{x}_i, \mathbf{z}_i, \mathbf{y}_i)$ triples in $\mathcal{T}$, which is randomly split into 5 folds for model cross-validation. Statistics of the VSO dataset after being processed is shown in Table 1. The numbers of triples are not identical as not all queries have divisible-by-5 number of triples. A small fraction of images appear in multiple queries, thus, the number of images is smaller than the number of triples.

Table 1. Data statistics

VSO		Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Total
	#images	29,745	30,033	29,654	30,039	29,614	149,085
	#triples	29,798	30,088	29,704	30,080	29,660	149,330
Yelp		BO	CH	LA	NY	SF	Total
	#images	19,054	19,054	19,054	19,054	19,054	95,270
	#triples	38,303	37,643	38,816	37,762	38,654	191,178

Yelp dataset consists of reviews of businesses in 5 US cities: Boston (BO), Chicago (CH), Los Angeles (LA), New York (NY), and San Francisco (SF). Each review has a rating, review text, and one or more images taken by the user. Sentiment is derived from the rating score, whereby ratings 1 and 2 are considered negative, ratings 4 and 5 are considered positive, while rating 3 is dropped as being ambiguous. Review text is split into shorter passages; each sentence is considered a text query. An image can be paired with multiple queries from the same review. To identify the best-matching text-image pairs, we rank the text queries based on cosine similarity of their TF-IDF vectors to that of the user-provided image caption, and consider up to 3 highest-ranked text queries to be relevant. These form the $(\mathbf{x}_i, \mathbf{z}_i, \mathbf{y}_i)$ triples in $\mathcal{T}$. To neutralize a text query $\mathbf{x}_i$, words strongly suggestive of sentiment (i.e., *objective_score* < 0.5 by SentiWordNet [4]) are replaced by a special token *-MSK-*. We balance the number of images between the two sentiments and across the cities via uniform sampling. Table 1 shows statistics of the Yelp dataset after being processed. The numbers of triples are not identical as not all queries have 3 matched images.

Evaluation Protocols. We adopt a similar test procedure as [18,39]. In our case, we conduct 5-fold validation, where for the *Yelp* dataset, four cities are used for training and one city is used for testing. During the test phase, for each query we construct a sample of 1,000 images, which include the correct images as well as uniformly sampled images in the test set. For each experiment, we report average result across 10 independent runs as well as the standard deviation.

Comparative Methods. We compare the proposed methods SML$_{OPPO}$ and SML$_{FLEX}$ with the following approaches:

- *Random* is the simplest baseline without learning,
- *CCA* [14] is one of the strongest statistical methods for cross-modal retrieval, which learns linear projections from input features, i.e., average Word2vec embeddings for text query and ResNet-50 features for images,
- *DCCA* [3] is the most recent extension of CCA transforming the same input features using multilayer perceptrons (i.e., we follow the original architecture of MLP in the original work) to capture non-linear interactions,
- *ACMR* [41] is a competitive method for cross-modal retrieval based on adversarial learning, in which modality-invariant representation in the common space is achieved by confusing a modality discriminator. We use the same neural network architectures for ours and ACMR for parity.

For all methods, the size of latent space is set to $D = 300$. For models that use stochastic gradient optimization, their parameters are updated with Adam [19] adaptive rule, batch size of 256, and learning rate of 0.001. Upon grid search for regularization $\lambda \in \{1e^{-5}, 1e^{-4}, \ldots, 1e^{-1}\}$ and margins $\tau_* \in \{0.0, 0.1, \ldots, 1.0\}$, the best hyper-parameters are obtained with cross-validation.

Metrics. We employ three established ranking metrics to measure the retrieval performance of the compared methods.

- *Percentile Rank (PR)* measures how well the correct images are being ranked amongst the image population. $PR = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{|D_i|}\sum_{j \in D_i}\frac{rank_j}{M}\right)$, where D_i denotes the set correct images for the query i, $rank_j$ is the rank of image j by the model, and M is the total number of images being ranked.
- *Normalized Discounted Cumulative Gain (NDCG)* measures the quality of ranking. $NDCG = \frac{1}{N}\sum_{i=1}^{N}\frac{DCG_i}{idealDCG_i}$, where $DCG_i = \sum_{j \in D_i}\frac{1}{\log{(rank_j+1)}}$, is the gain of image i relative to its position in a ranked list, and $idealDCG_i$ is the best achievable DCG_i in which all the correct images are at the top.
- *Recall@K (R@K)* denotes the ratio of correct images in the top-K retrieved images to the total number of correct images. $R@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{j \in D_i}\mathbb{1}[j \in L_i]}{|D_i|}$, where $\mathbb{1}[*]$ is the indicator function and L_i is the top-K retrieved images.

4.2 Quantitative Evaluation

Comparison Among Baselines. For an overall sense of the retrieval accuracy, Tables 2 and 3 report the results of comparative approaches on different metrics on the two datasets, respectively. Random is the ground-level reference for relative comparisons with other methods.

The statistical method CCA shows a competitive performance. Starting with pre-trained embeddings from Word2Vec and ResNet-50, it benefits from the richly-compressed features from those underlying models, even though the projections it learns on top of these features are linear. DCCA obtains better results, attributable to further adaptation by learning non-linear transformations optimized for the same CCA objective. Even so, the gap between CCA and DCCA seems to be close on VSO dataset as the text queries are simpler (single nouns).

Considered a strong method for cross-modal retrieval, ACMR outperforms DCCA across all metrics on VSO and also on Yelp except for *Recall@10*. However, by adopting adversarial learning with less stable optimization [20], the variances of ACMR tend to be higher than other methods. That explains why

Table 2. Performance of comparative methods on VSO dataset.

	Method	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Avg.
PR	Random	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00
	CCA	80.94 ± 0.00	81.05 ± 0.01	80.82 ± 0.01	80.67 ± 0.01	80.86 ± 0.01	80.87 ± 0.00
	DCCA	81.22 ± 0.11	81.33 ± 0.11	81.14 ± 0.05	81.00 ± 0.10	81.06 ± 0.09	81.15 ± 0.04
	ACMR	84.35 ± 0.21	84.43 ± 0.22	84.01 ± 0.14	84.16 ± 0.22	84.02 ± 0.26	84.19 ± 0.08
	SML$_{OPPO}$	**85.38** ± 0.10†	**85.42** ± 0.08†	85.04 ± 0.12†	**85.15** ± 0.06†	85.11 ± 0.11†	85.22 ± 0.04†
	SML$_{FLEX}$	85.34 ± 0.08†	**85.42** ± 0.11†	**85.10** ± 0.07†	85.14 ± 0.09†	**85.13** ± 0.07†	**85.23** ± 0.03†
NDCG (%)	Random	12.30 ± 0.02	12.32 ± 0.03	12.32 ± 0.03	12.31 ± 0.03	12.32 ± 0.03	12.31 ± 0.01
	CCA	19.55 ± 0.02	19.70 ± 0.02	19.59 ± 0.03	19.55 ± 0.04	19.62 ± 0.02	19.60 ± 0.01
	DCCA	20.08 ± 0.07	20.20 ± 0.10	19.96 ± 0.05	20.04 ± 0.07	20.06 ± 0.08	20.07 ± 0.03
	ACMR	20.64 ± 0.22	20.67 ± 0.20	20.41 ± 0.11	20.61 ± 0.21	20.56 ± 0.25	20.58 ± 0.09
	SML$_{OPPO}$	**21.95** ± 0.14†	21.93 ± 0.14†	21.74 ± 0.19†	21.80 ± 0.13†	21.89 ± 0.15†	21.86 ± 0.05†
	SML$_{FLEX}$	21.93 ± 0.15†	**21.97** ± 0.15†	**21.84** ± 0.12†	**21.87** ± 0.09†	**21.92** ± 0.13†	**21.91** ± 0.05†
R@10 (%)	Random	0.99 ± 0.07	1.00 ± 0.08	1.02 ± 0.05	0.99 ± 0.09	0.99 ± 0.05	1.00 ± 0.02
	CCA	12.00 ± 0.06	12.26 ± 0.06	12.01 ± 0.06	11.89 ± 0.09	12.19 ± 0.08	12.07 ± 0.03
	DCCA	13.25 ± 0.20	13.52 ± 0.25	13.08 ± 0.16	13.17 ± 0.13	13.23 ± 0.18	13.25 ± 0.08
	ACMR	14.01 ± 0.45	13.98 ± 0.41	13.61 ± 0.22	13.98 ± 0.49	13.94 ± 0.49	13.91 ± 0.18
	SML$_{OPPO}$	**16.75** ± 0.29†	16.77 ± 0.27†	16.43 ± 0.41†	16.50 ± 0.25†	**16.72** ± 0.38†	16.63 ± 0.13†
	SML$_{FLEX}$	**16.75** ± 0.36†	**16.85** ± 0.31†	**16.54** ± 0.26†	**16.57** ± 0.18†	16.71 ± 0.25†	**16.68** ± 0.12†

†improvements of *SML* models over the second-best baseline are statistically significant (p-value < 0.01).

Table 3. Performance of comparative methods on Yelp dataset.

	Method	BO	CH	LA	NY	SF	Avg.
PR	Random	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00	50.00 ± 0.00
	CCA	69.45 ± 0.01	68.65 ± 0.00	68.59 ± 0.01	69.01 ± 0.00	69.25 ± 0.01	68.99 ± 0.00
	DCCA	79.22 ± 0.26	78.67 ± 0.24	78.79 ± 0.34	79.01 ± 0.27	78.44 ± 0.28	78.83 ± 0.19
	ACMR	83.76 ± 0.89	83.32 ± 0.96	83.63 ± 0.65	83.67 ± 0.53	83.12 ± 0.80	83.50 ± 0.36
	SML$_{OPPO}$	**85.51** ± 0.09†	**84.84** ± 0.12†	84.89 ± 0.17†	84.92 ± 0.14†	84.32 ± 0.24†	84.89 ± 0.10†
	SML$_{FLEX}$	85.48 ± 0.12†	84.81 ± 0.10†	**84.93** ± 0.17†	**84.96** ± 0.13†	**84.38** ± 0.12†	**84.91** ± 0.07†
NDCG (%)	Random	12.65 ± 0.03	12.76 ± 0.03	12.38 ± 0.03	12.41 ± 0.03	12.60 ± 0.02	12.56 ± 0.01
	CCA	19.82 ± 0.04	19.21 ± 0.02	18.80 ± 0.02	18.89 ± 0.02	18.97 ± 0.01	19.14 ± 0.01
	DCCA	21.06 ± 0.21	20.85 ± 0.20	20.38 ± 0.24	20.54 ± 0.21	20.40 ± 0.20	20.64 ± 0.14
	ACMR	20.88 ± 0.91	21.00 ± 0.92	20.29 ± 0.70	20.59 ± 0.54	21.01 ± 0.76	20.75 ± 0.38
	SML$_{OPPO}$	**22.83** ± 0.14†	22.51 ± 0.26†	21.66 ± 0.21†	21.95 ± 0.31†	22.20 ± 0.46†	22.23 ± 0.16†
	SML$_{FLEX}$	22.82 ± 0.09†	**22.57** ± 0.25†	**21.77** ± 0.33†	**22.10** ± 0.40†	**22.44** ± 0.19†	**22.34** ± 0.16†
R@10 (%)	Random	0.96 ± 0.06	1.02 ± 0.08	0.99 ± 0.06	0.98 ± 0.06	1.01 ± 0.04	0.99 ± 0.02
	CCA	12.78 ± 0.05	11.31 ± 0.05	11.80 ± 0.04	11.56 ± 0.04	11.55 ± 0.04	11.80 ± 0.02
	DCCA	14.75 ± 0.47	13.96 ± 0.43	14.39 ± 0.49	14.62 ± 0.52	13.77 ± 0.40	14.30 ± 0.31
	ACMR	13.43 ± 1.94	13.45 ± 1.86	13.19 ± 1.52	13.81 ± 1.19	14.25 ± 1.60	13.62 ± 0.81
	SML$_{OPPO}$	**17.49** ± 0.27†	16.44 ± 0.58†	16.04 ± 0.51†	16.59 ± 0.69†	16.65 ± 0.92†	16.64 ± 0.34†
	SML$_{FLEX}$	17.45 ± 0.22†	**16.62** ± 0.56†	**16.21** ± 0.67†	**16.93** ± 0.84†	**17.10** ± 0.40†	**16.86** ± 0.32†

†improvements of *SML* models over the second-best baseline are statistically significant (p-value < 0.01).

DCCA can surpass ACMR on *Yelp-Recall@10*, which takes into account only *top-10* items rather than global ranking by *Percentile Rank* and *NDCG*.

Effect of Proposed Sentiment-Orientation. By leveraging sentiment information, both SML$_{OPPO}$ and SML$_{FLEX}$ significantly outperform all the sentiment-insensitive baselines across virtually all metrics and datasets. On average, SML$_{FLEX}$ model is slightly better than SML$_{OPPO}$. This is not unexpected as SML$_{OPPO}$ makes a stricter assumption on the direction of sentiment vectors.

Figure 2 visualizes the learned metric spaces of SML with four sample queries: "bill", "service", "drink", and "toilet", and their sentiment-infused queries, by projecting their vectors onto 2D using PCA [44]. For SML$_{OPPO}$, we observe opposing directions between positive and negative sentiments. For SML$_{FLEX}$, they are not directly opposing but still form obtuse angles. This indicates a strong contrast of the sentiment concepts captured by the models. In addition, with the relaxation, SML$_{FLEX}$ can pull "bill" and "service" together, i.e., they are considered closer semantically as compared to "drink" or "toilet". This could be an explanation for the higher accuracies exhibited by SML$_{FLEX}$.

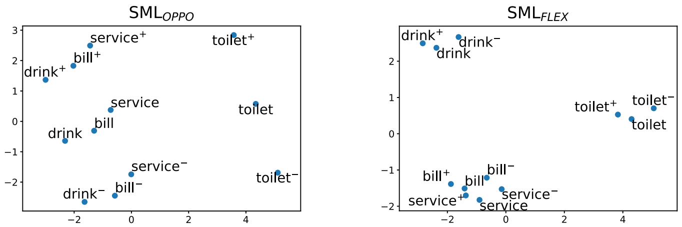


Fig. 2. Learned metric spaces of SML visualized in 2D using PCA.

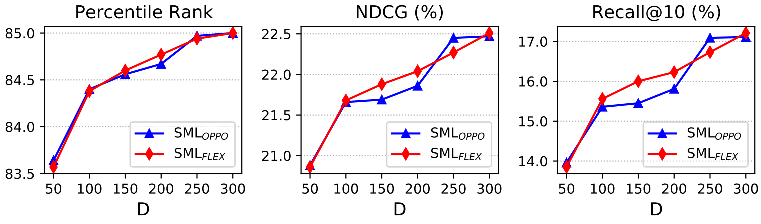


Fig. 3. Performance with varying the number of dimensions D of metric spaces.

Effect of Dimensionality. To further understand how the size of the metric space affects SML models, we conduct an experiment with different settings of dimensionality D on Yelp dataset. Figure 3 illustrates performance of the SML$_{OPPO}$ and SML$_{FLEX}$ when D ranges from 50 to 300. Across all metrics, the model performances are sharply boosted when D increases from $50 - 200$ and tends to converge around the values of $250 - 300$, especially so in terms of *Percentile Rank*. Even though the performance of SML$_{FLEX}$ is potentially better if D goes beyond 300, it does not seem to be the case for SML$_{OPPO}$. Thus, we stop at $D = 300$, and all experiments are also conducted under this setting.

4.3 Case Studies

To gain more insights on the SML models, especially when the notion of sentiment is visually prominent, we illustrate examples from *Yelp-LA* dataset. Figure 4 shows retrieved images with different queries and sentiments. In addition, we include ACMR as a reference baseline. In each ranking (top-4 are vertically positioned), the ground-truth is marked with a dotted rectangle. First of all, we notice that SML$_{FLEX}$ can retrieve the correct image in both cases and SML$_{OPPO}$ in one case. This observation concurs with the higher retrieval performance of SML$_{FLEX}$ in the previous quantitative analysis. Interestingly, in the second example, not only can SML$_{FLEX}$ pull the correct one into top-4, but it also illustrates a strong notion of sentiment when the first-ranked image, *"burned pizza"*, is evidently negative. Meanwhile, ACMR retrieves images based on the concepts implied by text queries, but not the ground-truth in both cases, presumably as it might not have captured the sentiment aspects well.

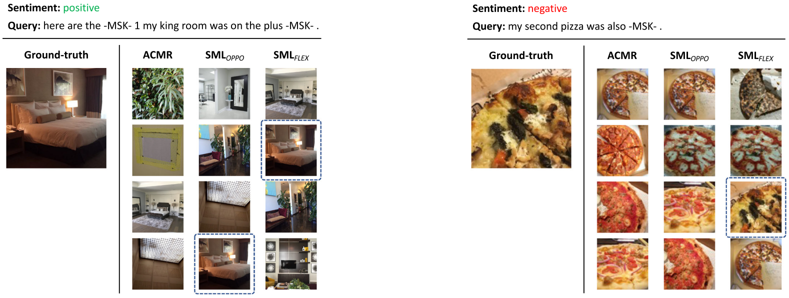


Fig. 4. Top retrieved images organized along queries.

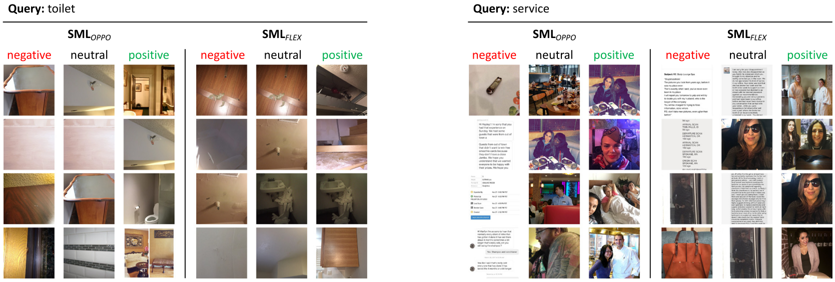


Fig. 5. Top retrieved images while changing sentiments.

For understanding the notion of sentiments captured by SML$_{OPPO}$ and SML$_{FLEX}$, in Fig. 5 we analyze 2 queries *"toilet"* and *"service"*, while alternating the sentiment input. Neutral means the sentiment vectors are set to zeros. For both queries, there are contrasts between *"negative"* and *"positive"* images. SML$_{OPPO}$ demonstrates that effect more clearly, especially on *"toilet"* query. This is due to desired constraint of the model, and can also be explained via Fig. 2 (i.e., sentiment vectors of *"toilet"* query are slightly longer in magnitude than the other queries'). For *"service"* query, negative images show complaint notes which imply customer unhappiness. Surprisingly, the positive images turn out to be smiling faces showing customer satisfaction. With such sentimental concepts captured via SML models, the case studies shed some light on understanding how the models work as well as how the performance could be interpreted.

5 Conclusion

We propose Sentiment-Oriented Metric Learning framework to incorporate sentiments into text-to-image retrieval. Our models SML$_{OPPO}$ and SML$_{FLEX}$ outperform comparable baselines on experiments involving images obtained from Flickr (VSO) as well as from online reviews (Yelp). As future work, the proposed framework could potentially be further extended to learn other visual concepts (e.g., human emotions, fashion styles) for text-to-image retrieval.

Acknowledgement. This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

References

1. Araki, T., Hino, H., Akaho, S.: A kernel method to extract common features based on mutual information. In: Loo, C.K., Yap, K.S., Wong, K.W., Teoh, A., Huang, K. (eds.) ICONIP 2014. LNCS, vol. 8835, pp. 26–34. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12640-1_4
2. Anderson, T.: An Introduction to Multivariate Statistical Analysis. Wiley, Hoboken (1984). [una introducción al análisis estadístico multivariado]
3. Andrew, G., Arora, R., Bilmes, J.A., Livescu, K.: Deep canonical correlation analysis. ICML **28**, 1247–1255 (2013)
4. Baccianella, S., Esuli, A., Sebastiani, F.: Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In: Calzolari, N., et al. (eds.) LREC (2010)
5. Bach, F.R., Jordan, M.I.: Kernel independent component analysis. J. Mach. Learn. Res. **3**, 1–48 (2002)
6. Borth, D., Ji, R., Chen, T., Breuel, T., Chang, S.F.: Large-scale visual sentiment ontology and detectors using adjective noun pairs. In: ACM Multimedia (2013)
7. Cao, Y., Long, M., Wang, J., Yang, Q., Yu, P.S.: Deep visual-semantic hashing for cross-modal retrieval. In: Krishnapuram, B., Shah, M., Smola, A.J., Aggarwal, C.C., Shen, D., Rastogi, R. (eds.) SIGKDD (2016)

8. Feng, F., Wang, X., Li, R.: Cross-modal retrieval with correspondence autoencoder. In: ACM Multimedia (2014)
9. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS (2011)
10. Gordo, A., Almazán, J., Revaud, J., Larlus, D.: Deep image retrieval: learning global representations for image search. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 241–257. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_15
11. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: an overview with application to learning methods. Neural Comput. **16**(12), 2639–2664 (2004)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Hotelling, H.: Relations between two sets of variates. Biometrika **28**(3/4), 321–377 (1936)
15. Hsieh, C., Yang, L., Cui, Y., Lin, T., Belongie, S.J., Estrin, D.: Collaborative metric learning. In: Barrett, R., Cummings, R., Agichtein, E., Gabrilovich, E. (eds.) WWW (2017)
16. Hsieh, W.W.: Nonlinear canonical correlation analysis by neural networks. Neural Netw. **13**(10), 1095–1105 (2000)
17. Jiang, Q., Li, W.: Deep cross-modal hashing. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, pp. 3270–3278. IEEE Computer Society (2017)
18. Karpathy, A., Li, F.: Deep visual-semantic alignments for generating image descriptions. In: CVPR (2015)
19. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2015)
20. Kodali, N., Abernethy, J., Hays, J., Kira, Z.: On convergence and stability of gans (2017). arXiv preprint: arXiv:1705.07215
21. Koestinger, M., Hirzer, M., Wohlhart, P., Roth, P.M., Bischof, H.: Large scale metric learning from equivalence constraints. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2288–2295. IEEE (2012)
22. Kulis, B.: Metric learning: a survey. Found. Trends Mach. Learn. **5**(4), 287–364 (2013)
23. Lai, P.L., Fyfe, C.: A neural implementation of canonical correlation analysis. Neural Netw. **12**(10), 1391–1397 (1999)
24. Li, Z., Lin, D., Meng, H.M., Tang, X.: Discriminant mutual subspace learning for indoor and outdoor face recognition. In: 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), 18–23 June 2007, Minneapolis, Minnesota, USA. IEEE Computer Society (2007)
25. Lin, D., Tang, X.: Inter-modality face recognition. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 13–26. Springer, Heidelberg (2006). https://doi.org/10.1007/11744085_2
26. Liu, W., Tsang, I.W.: Large margin metric learning for multi-label prediction. In: Bonet, B., Koenig, S. (eds.) Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25–30, 2015, Austin, Texas, USA, pp. 2800–2806. AAAI Press (2015)

27. Melzer, T., Reiter, M., Bischof, H.: Nonlinear feature extraction using generalized canonical correlation analysis. In: Dorffner, G., Bischof, H., Hornik, K. (eds.) ICANN 2001. LNCS, vol. 2130, pp. 353–360. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44668-0_50
28. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, pp. 3111–3119 (2013)
29. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal deep learning. In: Getoor, L., Scheffer, T. (eds.) Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28–July 2, 2011, pp. 689–696. Omnipress (2011)
30. Peng, Y., Qi, J.: CM-GANs: cross-modal generative adversarial networks for common representation learning. ACM Trans. Multimed. Comput. Commun. Appl. (TOMM) **15**(1), 1–24 (2019)
31. Ragusa, E., Cambria, E., Zunino, R., Gastaldo, P.: A survey on deep learning in image polarity detection: balancing generalization performances and computational costs. Electronics **8**(7), 783 (2019)
32. Sharma, A., Kumar, A., Daume, H., Jacobs, D.W.: Generalized multiview analysis: a discriminative latent space. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2160–2167. IEEE (2012)
33. Shen, F., Zhou, X., Yang, Y., Song, J., Shen, H.T., Tao, D.: A fast optimization method for general binary code learning. IEEE Trans. Image Process. **25**(12), 5610–5621 (2016)
34. Song, J., Yang, Y., Yang, Y., Huang, Z., Shen, H.T.: Inter-media hashing for large-scale retrieval from heterogeneous data sources. In: Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, pp. 785–796 (2013)
35. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: closing the gap to human-level performance in face verification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1701–1708 (2014)
36. Truong, Q.T., Lauw, H.W.: Visual sentiment analysis for review images with item-oriented and user-oriented CNN. In: Proceedings of the 25th ACM International Conference on Multimedia, pp. 1274–1282 (2017)
37. Truong, Q.T., Lauw, H.W., Aumüller, M., Nitta, N.: Reproducibility companion paper: visual sentiment analysis for review images with item-oriented and user-oriented CNN, pp. 4444–4447 (2020)
38. Vadicamo, L., et al.: Cross-media learning for image sentiment analysis in the wild. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 308–317 (2017)
39. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
40. Wan, J., et al.: Deep learning for content-based image retrieval: a comprehensive study. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 157–166 (2014)
41. Wang, B., Yang, Y., Xu, X., Hanjalic, A., Shen, H.T.: Adversarial cross-modal retrieval. In: Proceedings of the 25th ACM International Conference on Multimedia, pp. 154–162 (2017)

42. Wang, J., He, Y., Kang, C., Xiang, S., Pan, C.: Image-text cross-modal retrieval via modality-specific feature learning. In: Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, pp. 347–354 (2015)
43. Wang, W., Arora, R., Livescu, K., Bilmes, J.: On deep multi-view representation learning. In: International Conference on Machine Learning, pp. 1083–1092 (2015)
44. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemom. Intell. Lab. Syst. **2**(1–3), 37–52 (1987)
45. Xu, X., Shen, F., Yang, Y., Shen, H.T., Li, X.: Learning discriminative binary codes for large-scale cross-modal retrieval. IEEE Trans. Image Process. **26**(5), 2494–2507 (2017)
46. Xu, Z.E., Chen, M., Weinberger, K.Q., Sha, F.: From sBoW to dCoT marginalized encoders for text representation. In: Chen, X., Lebanon, G., Wang, H., Zaki, M.J. (eds.) 21st ACM International Conference on Information and Knowledge Management, pp. 1879–1884. ACM (2012)
47. You, Q., Luo, J., Jin, H., Yang, J.: Robust image sentiment analysis using progressively trained and domain transferred deep networks. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25–30, 2015, Austin, Texas, USA, pp. 381–388. AAAI Press (2015)
48. Zhai, D., Chang, H., Shan, S., Chen, X., Gao, W.: Multiview metric learning with global consistency and local smoothness. ACM Trans. Intell. Syst. Technol. **3**(3), 53:1–53:22 (2012)
49. Zhai, X., Peng, Y., Xiao, J.: Heterogeneous metric learning with joint graph regularization for cross-media retrieval. In: Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence. AAAI Press (2013)
50. Zheng, F., Tang, Y., Shao, L.: Hetero-manifold regularisation for cross-modal hashing. IEEE Trans. Pattern Anal. Mach. Intell. **40**(5), 1059–1071 (2018)

Metric Learning for Session-Based Recommendations

Bartłomiej Twardowski[1,2], Paweł Zawistowski[1](✉), and Szymon Zaborowski[3]

[1] Institute of Computer Science, Warsaw University of Technology, Warsaw, Poland
pawel.zawistowski@pw.edu.pl

[2] Computer Vision Center, Universitat Autónoma de Barcelona, Bellaterra, Spain

[3] Sales Intelligence, Wrocław, Poland

Abstract. Session-based recommenders, used for making predictions out of users' uninterrupted sequences of actions, are attractive for many applications. Here, for this task we propose using metric learning, where a common embedding space for sessions and items is created, and distance measures dissimilarity between the provided sequence of users' events and the next action. We discuss and compare metric learning approaches to commonly used learning-to-rank methods, where some synergies exist. We propose a simple architecture for problem analysis and demonstrate that neither extensively big nor deep architectures are necessary in order to outperform existing methods. The experimental results against strong baselines on four datasets are provided with an ablation study.

Keywords: Session-based recommendations · Deep metric learning · Learning to rank

1 Introduction

We consider the session-based recommendation problem, which is set up as follows: a user interacts with a given system (e.g., an e-commerce website) and produces a sequence of events (each described by a set of attributes). Such a continuous sequence is called a session, thus we denote $s_k = e_{k,1}, e_{k,2}, \ldots, e_{k,t}$ as the k-th session in our dataset, where $e_{k,j}$ is the j-th event in that session. The events are usually interactions with items (e.g., products) within the system's domain. In comparison to other recommendation scenarios, in the case of session-based recommendations—information about the user across sessions is not available (in contrast to session-aware recommendations). Also, the browsing sessions originate from a single site (which is different from task-based recommendations).

The sequential nature of session-based recommendations means that it shares some similarities with tasks found within natural language processing (NLP), where sequences of characters, words, sentences, or paragraphs are analyzed. This connection leads to a situation where many methods that are successful

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 650–665, 2021.
https://doi.org/10.1007/978-3-030-72113-8_43

in NLP are later applied to the field of recommendations. One such example is connected with recurrent neural networks (RNNs), which have led to a variety of approaches applied to recommender systems [9,36,41]. Another, one is connected with the transformer model [3] applied to model users' behavior [38].

Despite the apparent steady progress connected with neural methods, there are some indications that properly applied classical methods may very well beat these approaches [21]. Therefore in this paper, we propose combining the classical KNN algorithm with a neural embedding function based on an efficient neighborhood selection of top-n recommendations. The method learns embeddings of sessions and items in the same metric space, where a given distance function measures dissimilarity between the user's current session, and next items. For this task, a metric learning loss function and data sampling are used for training the model. During the evaluation, the nearest neighbors are found for the embedded session. This makes the method attractive for real-life applications, as existing tools and methods for neighborhood selection can be used. The main contributions of this paper are as follows:

- we verify selected metric learning tools for session-based recommendations,
- we present a comparison of the metric learning approach and learning to rank, where some potential future directions for recommender systems can be explored based on the latest progress in deep metric learning,
- we introduce a generic model for recommendations, which allows the impact of different architectures of session and item encodings on the final performance to be evaluated—which we do in the provided ablation studies,
- we evaluate our approach using known protocols from previous session-based recommendation works against strong baselines over four datasets; for the sake of reproducibility and future research[1].

2 Related Works

Session-Based Recommendations. Time and sequence models in context-aware recommender systems were used before the deep learning methods emerged. Many of these approaches can be applied to session-based recommendation problems with some additional effort to represent time, e.g., modeling it as a categorical contextual variable [10,29] or explicit bias while making predictions [16]. The sequential nature of the problem can also be simplified and used with other well-known methods, i.e., Markov chains [31], or applying KNNs combined with calculating the session items sets' similarities [12].

The Gru4Rec method [9] has been an important milestone in applying RNNs to session-based recommendation tasks. The authors focused on sessions solely represented by interactions with items and proposed a few contributions: using GRU cells for session representation, negative exemplars mining within mini-batch, and a new `TOP1` loss function. In the followup work [8] authors proposed further improvements to loss functions. Inspired by the successful application of

[1] https://github.com/btwardow/dml4rec.

convolutional neural networks (CNNs) for textual data [14], new methods were proposed. One example is the Caser approach [39], which uses a CNN-based architecture with max pooling layers for top-n recommendations for sessions. Another, proposed in [47], utilises dilated 1D convolutions similar to WaveNet [27]. The embedding techniques known from NLP, e.g. skip-gram and CBOW, were also extensively investigated for recommender systems. Methods such as item2vec and prod2vec were proposed for embedding-based approaches. However recently conducted experiments with similar approaches, were unsuccessful in obtaining better results than simple neighbourhood methods for session-based recommendations [20].

Metric Learning. Metric learning has a long history in information retrieval. Among the early works, the SVM algorithm was used to learn from relative comparisons in [32]. Such an approach directly relates to Mahalanobis distance learning, which was pursued in [22] and [17]. Even though new and more efficient architectures emerge constantly, the choice of loss functions and training methods still plays a significant role in metric learning. In [6], the authors proposed the use of contrastive loss, which minimizes the distance between similar pairs while ensuring the separation of non-similar objects by a given margin. For some applications, it was found hard to train, and in [11], the authors proposed improvement by using an additional data point—*anchor*. All three data points make an input to the triplet loss function, where the objective is to keep the negative examples further away from the anchor than the positive ones with a given margin. Recently more advanced loss functions were proposed: using angular calculation in triplets [42], signal-to-noise ratio [48], and multi-similarity loss [43]. Still, contrastive and triple losses in many applications have proven to be a strong baseline when trained correctly [7]. Nevertheless, the high computational complexity of data preparation (i.e. creating point tuples for training) for contrastive and triplet approaches cannot be solved by changing only the loss function. These problems are addressed by different dataset sampling strategies and efficient mining techniques. One notable group here is online approaches, which try to explore relations in a given mini-batch while training, e.g., hard mining [7], n-pairs [37], the lifted structure method [26], and weighting by distance [46]. Many combinations of sampling and mining techniques, along with the loss functions, can be created, which makes a fair comparison hard [4,13,24].

3 Metric Learning vs. Ranking Learning for Session-Based Recommendations

An ordered output of the session-based recommender in the form of a sorted list for a given input s_k is the ranking r_k. In learning-to-rank, as well as recommender systems, the main difficulty is the direct optimization of the output's quality measures (e.g., recall, mean average precision, or mean reciprocal rank). The task is hard for many (gradient-based) methods due to the non-smoothness of the optimized function [1]. This problem can be resolved either by minimizing a convex upper bound of the loss function, e.g., SVM-MAP [49], or by optimizing a smoothed version of an evaluation measure, e.g., SoftRank [40]. Many approaches exist, which depend on the model of ranking: pointwise (e.g. SLIM [25], MF [15], FM [29]), pairwise (BPR [30], pLPA [18], GRU4Rec [9]), or listwise (e.g. CLIMF/xCLIMF [2,33], GAPfm [34], TFMAP [35]). However, not all are applicable to session-based recommendations. Pairwise approaches for ranking top-N items are the most commonly used, along with neural network approaches. In the GRU4Rec method, two pairwise loss functions for training were used—*Bayesian Personalized Ranking* (BPR) [30] and TOP-1:

$$l_{\mathrm{BPR}}(s_k, i_p, i_n) = -\ln\big(\sigma(\hat{y}_{s_k,i_p} - \hat{y}_{s_k,i_n})\big) \tag{1}$$

$$l_{\mathrm{TOP1}}(s_k, i_p, i_n) = \sigma(\hat{y}_{s_k,i_n} - \hat{y}_{s_k,i_p}) + \sigma(\hat{y}^2_{s_k,i_n}) \tag{2}$$

where s_k denotes the session for which i_p is a positive example of the next item, and i_n is a negative one. The $\hat{y}_{s_k,i}$ is a score value predicted by the model for the session s_k and the item i. The score value allows items to be compared and the ordered list r_k to be produced, where i.e. $i_p >_{r_k} i_n$, and $>_{r_k} \subset I^2$ denotes total order [30].

In metric learning, the main goal is to align distance with dissimilarity of objects. In [6], the contrastive loss function for two vectors $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^{\mathrm{d}}$ is given as:

$$l_{\mathrm{Cont}}(\mathbf{x}_i, \mathbf{x}_j) = y d(\mathbf{x}_i, \mathbf{x}_j) + (1-y)\max\big(0, d(\mathbf{x}_a, \mathbf{x}_n) - m\big) \tag{3}$$

where y is an indicator variable, 1 if both vectors are from the same class, 0 otherwise, $m \in R_+$ is the margin, and $d(\mathbf{x_i}, \mathbf{x_j})$ is a distance function, e.g., Euclidian or cosine. This loss function *pulls* similar items ($y = 1$) and *pushes* dissimilar ones. A direct extension – the triplet loss [11] – is defined as follows:

$$l_{\mathrm{Triplet}}(\mathbf{x}_a, \mathbf{x}_p, \mathbf{x}_n) = \max\big(0,\ d(\mathbf{x}_a, \mathbf{x}_p) - d(\mathbf{x}_a, \mathbf{x}_n) + m\big) \tag{4}$$

where $\mathbf{x}_p$ and $\mathbf{x}_n$ are respectively positive and negative items for a given anchor $\mathbf{x}_a$ and $m \in R_+$ is the margin.

Both contrastive and triplet losses can be used to optimize the goal of the total ordering of objects [19,30] as induced by the learned metric. If $d(\mathbf{x_i}, \mathbf{x_j}) = 0$ does not imply $\mathbf{x_i} = \mathbf{x_j}$, d is then a pseudo metric [32], and total order cannot be induced. If we assume that two functions $\varphi(s_a) = x_a$ and item $\omega(i_k) = x_k$ are given to embed the session and item to the same $\mathbb{R}^{\mathrm{d}}$ space, where scoring is done by cosine similarity $\hat{y}_{s,i} = 1 - d(\varphi(s), \omega(i))$, then previously defined ranking losses and metric can be presented as:

$$l_{\mathrm{BPR}}(s_k, i_p, i_n) = -\ln(\sigma(d_{kn} - d_{kp}))\ , \tag{5}$$

$$l_{\mathrm{TOP1}}(s_k, i_p, i_n) = \sigma(d_{kp} - d_{kn}) + \sigma((1 - d_{kn})^2)\ , \tag{6}$$

$$l_{\mathrm{Triplet}}(s_k, i_p, i_n) = \max(0, d_{kp} - d_{kn} + m) \tag{7}$$

where $d_{kj} = d(\varphi(s_k), \omega(i_j))$. A direct connection can be seen: that minimizing each of the loss functions will try to keep i_p closer to s_k than i_n. In all cases, for session-based recommendations, positive items are known, while the negatives are sampled from the rest of the items (e.g., uniformly or by a given heuristic). In both BPR and TOP1, a sigmoid $\sigma(x)$ function is used for optimizing AUC in place of a non-differentiable Heaviside function directly, as explained in [30]. In TOP1, the authors added a regularization term for negative predictions, which further constrains the embedding space by keeping negatives close to zero. Metric learning losses use a rectifier nonlinearity ($\max(0, x)$) to prevent from moving data points that are already in order. When considering partial derivative w.r.t distances between our anchor session s_k and positive and negative items, they contribute equally, as was discussed in [43]. If in a single calculation, more relations are explored (usually inside the same mini-batch), techniques like lifted structures [26] are used. However, the relations are made between known classes of examples. In learning to rank, each instance inside a selected set can be ordered, which can be used i.e., to estimate overall ranking, like in Weighted Approximated-Ranking Pairwise (WARP) [44]. All losses have one more important thing in common: they do not take into account the relationship between positive and negative items (without the anchor). This is a subject of further improvements in metric learning methods like [42,43]. In our solution, we propose using a simple weighting for ranking to address this shortcoming.

4 Proposed Method

We propose a method for session-based recommendations using deep metric learning, where the main input is the sequence of user's actions (i.e. the session) $s_k = \{e_{k,1}, e_{k,2}, \ldots, e_{k,t}\} \in S$, and items $i \in I$. At the high-level the network's architecture can be described as $\hat{y}_{s_k,i} = d(\varphi(s_k), \omega(i))$, where φ and ω denote the session and item encoders respectively, and $\hat{y}_{s_k,i}$ denotes how score for recommending item i in the context of session s_k. We decided on a simple and modular approach in order to investigate the impact of each module on the final outcome—focusing mainly on the session encoder and different metric loss functions. The only constraint of the model towards the used network is the used dimensionality of $\varphi(s_k),\ \omega(i) \in \mathbb{R}^{\mathrm{d}}$ for learning a common metric space. The outputs of networks are normalized and cosine distance functions $d(\varphi(s_k), \omega(i))$ are used in final scoring $\hat{y}_{s_k,i}$ calculation.

4.1 Metric Loss for Ranking

Triplet Loss. The overall triplet loss function is calculated over the prepared training dataset. Assuming that session s_k has L positive items, the final triplet loss function for balanced positive-negative sampling is as follows:

$$L = \frac{1}{|S|} \sum_{s_k \in S} \sum_{j=0}^{L} w_j \max\left(0, d\left(\varphi(s_k), \omega(i_p)\right) - d\left(\varphi(s_k), \omega(i_n)\right) + m\right) \tag{8}$$

where weight w_j is weight used for particular position. In experiments, we used $\sqrt{1/(1+j)}$ for weighting, which is expected to change the magnitude of the calculated gradient based on the ranking position. To incorporate the relation between positives and negatives items we used a *swaping* technique for a triplet loss, where anchor is exchanged with positive and the final distance to a negative point is taken as a minimum $d'_{kn} = min\,(d_{kn}, d_{pn})$.

Neighborhood Component Analysis with Smoothing (NCAS) Loss. Based on the NCA loss [5,23] used commonly in deep metric learning we introduce a version prepared for ranking session-based recommendations as follows:

$$p(i_j|s_k) = \frac{exp\,(-d\,(\varphi(s_k), \omega(i_j)))}{\sum_{i_j \in Z} exp\,(-d(\varphi(s_k), \omega(i_j)))} \tag{9}$$

$$L_{NCAS} = \frac{1}{|S|} \sum_{s_k \in S} KLD\,(p(i|s_k)||p'(i)) \tag{10}$$

where predictions of true labels inside N-sized mini-batches are smoothed with: $p'(i) = (1-\epsilon)p(i) + \epsilon/N$ and Z is a sampled set containing positive and negative examples for each session s_k. The main goal of using this loss function was to compare the triplet loss to other functions that can be applied for our setting in order to get more insight of its applicability and results.

4.2 Session Encoder Networks

We use several neural network architectures for the session encoder module. Each one of these networks takes as an input a sequence of session events, which are clicked items in all used datasets, and encodes it to a vector of embedding size d. Used network architectures as session encoders go as follows:

- Pooling – this architecture embeds the sequence of clicked items to a vector of size d by pooling the maximum or average value in each dimension. Inspired by how pre-trained embeddings (e.g. word2vec) are used in NLP downstream tasks. However, all relations in a sequence are lost.
- CNN based approaches including TextCNN [14], TagSpace [45], Caser [39].
- RNN-based approaches—these use one of the chosen recurrent networks (GRU, LSTM, RNN) to encode the sequence followed by multiple fully connected layers to generate recommendation scores for individual items.

4.3 Positive and Negative Sampling

Training data is prepared from all available users' sessions S. We want to predict the user's next action for a given session s_k. Thus, training data preparation tries to enforce this for the model. Each session is split randomly—the first part is used as an input for the network s_k, and the following actions with items are used as positive examples for that session $i_{p,1}, \ldots, i_{p,l}$. For each l positive, the same number of negatives are sampled randomly. We investigated a few different strategies in case the session after random split has not enough positive examples. One of the successful approaches we used is to prepare more positives before training using KNN method. Sampling is done at a beginning of each training epoch. However, the improved MRR score is counterbalanced by lower items' coverage.

Table 1. Experimental dataset stats—(before) and after preprocessing.

Dataset	Source	Items		Sessions		Events	
RR/5	Retail rocket	32K	(117K)	64K	(380K)	242K	(606K)
RSC15/64	2015 RecSys challange	17K	(34K)	118K	(1.7M)	495K	(6.6M)
SI-T	Proprietary e-commerce data #1	2K	(2K)	114K	(119K)	305K	(315K)
SI-D	Proprietary e-commerce data #2	3K	(3K)	25K	(34K)	94K	(106K)

In other works, the negative sampling is done randomly from all non-positive items, e.g., [29]. From the optimization perspective [9] took a different approach and sample negative examples from the same mini-batch given to the network. What relates to online samples mining used in deep metric learning techniques, but here without enforcing a margin of error like in hard negative mining [7].

5 Experiments

To conduct our experiments, we have followed the procedure utilized by [21], and used five splits for `RR` and one 64'th of `RSC15` data. For each dataset, we have split the events into individual user sessions and removed the ones that contained only a single event. Furthermore, in our experiments, we have included only items that occurred at least five times in the data. A train-test split was prepared by taking the last 10% of sessions. We further evaluated our models by using common information retrieval and ranking evaluation metrics: mean average precision, mean reciprocal rank, recall, precision, and hit ratio. All metrics were computed on a list of top 20 recommendations. Following [21] and [9], in case of MRR@20 and HR@20 only the next item was used as the ground truth. This *no look-ahead* evaluation can be considered as a more adequate, when after each of a user's action a session state is updated and predictions for the next user's step is given.

5.1 Datasets and Baselines

To conduct the experiments, we used four datasets from the e-commerce domain, which are summarized in Table 1. Two of these (`RR/5` and `RSC15/64`) are standard benchmarks for session-based recommender systems, while the remaining ones are smaller, real-world proprietary datasets with data gathered in the early 2020. The difference between `SI-T` and `SI-T` is the category of products for which data were collected. In all datasets users' events are represented only by interactions with a products (i.e., view, click), thus $e_{k,l} \sim I$.

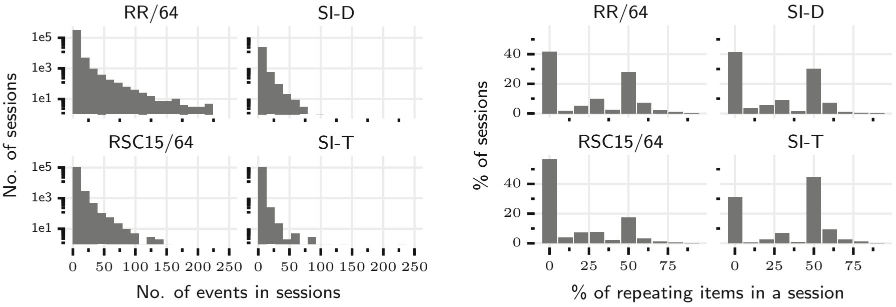


Fig. 1. Session length distribution (left) and repeating items (right) for each dataset.

Figure 1(left) presents a histograms of session lengths for the preprocessed datasets, which shows that short sessions seem to dominate in all datasets. This might be more challenging for methods that focus on the sequential nature of the users' data. Furthermore, when analyzing the percentage of recurring items among the sessions presented on Fig. 1(right), it may be noticed that the session frequently contain multiple interactions with the same products. The data suggests that users seem to revisit already seen items quite often. However, this also poses an interesting question from the perspective of recommender systems: should such a system suggest items that a user has already seen in the given session or only new ones? The answer will depend on the specific use case and whether the system should provide a more explorative or exploitative user experience.

We compared our Session-based Metric Learning (`SML`) method against six baseline algorithms. Starting from the simplest ones, `POP` denotes a simple popularity-based algorithm, which simply recommends the top-n most popular items. `SPOP` recommend items already seen in the session ordered by number of occurrences and fills the rest with popular ones. This recommender performs well when predictions are expected to be repetitions. The `KNN` algorithm was the basis of the next two baseline methods: `SKNN` and `VSKNN`. The `SKNN` approach for a given session recommends the top-n most frequent items among the `K`-most similar sessions from the training data, for which a cosine distance is used. The `VSKNN` [21] approach works similarly, however it puts more weight on more recent events in a given session. The last two methods include a Markov first-order recommender reported as `MARKOV-1` and `GRU4Rec+` [8].

5.2 Implementation Details

All the variations of the proposed model were implemented using the PyTorch [28] library and trained in an end-to-end fashion with Adam optimizer using $lr = 0.001$, for max 150 epochs (early stop after lowering lr three times when improvement on 5% validation data is lower than 0.5%) with batch size of 32, and 8 positive/negative samples per session. Max session length was 15 for `RR/5` and `RSC15/64`, and 8 for SI—this plays an important role for CNNs where all sessions are padded to exactly the same size. For item embedding simple feed-forward network with *tanh* activation is used. The embedding dimension is set to 400 for all the methods. The margin value m for triplet loss is set to 0.3. Smoothing parameter for NCAS is set to $\epsilon = 0.3$. For the RNN encoder a GRU cells are used with, 400 dimensions. For the TextCNN convolution filters of sizes $1, 3, 5$ were used.

5.3 Performance Comparison

Evaluation. In Table 2 we present the results obtained during the experiments conducted with the proposed method and compare them against the baselines. Not all combinations of session encoders with loss function are presented, only the most promising or interesting ones from the future research perspective (e.g., NCAS for RSC15/64 and RR/5).

The modification introduced by VSKNN to the non-weighted version of the method (i.e., SKNN) seems to be effective for all the datasets, thus making VSKNN a strong baseline indeed. Nevertheless, in some cases (like RR/5), the simpler SKNN method still obtains better results. Dataset specifics and used metrics play an important role here, as can be seen in Fig. 1 (right)—RR/5 in comparison with other datasets (especially RSC15/64) contains more repeating items. If we place them at the beginning of our recommendation and fill up the rest with the most popular items, we can receive high MRR@20 values. However, the practical usefulness of such recommendations can be questionable.

The low results of MARKOV-1 for all datasets show that a simple association of the item and the next following action is not enough to obtain good results. Extracting additional information from entire sequences is needed to improve recommendations, which is the basis on which the sequential modeling with GRU4Rec+ method stands. Still, in most cases, it is less accurate in the meaning of used metrics than the simple heuristic of VSKNN. One possible explanation is that the VSKNN model additionally incorporates *recency* in the scoring function. We can consider that as a simply encoded contextual information about when the sequence occurred. This information is not used in other models. When scoring sequences within short periods of time this may not introduce a big difference, but becomes important as the time difference increases, as e.g., some trends arise, and others fade out.

From the overall results, our SML family methods are the best for two datasets, the proprietary SI-T and the open available RSC15/64. For SI-T the proposed triplet loss function seems to be the right choice, wherein the case of RSC15/64, training with NCAS is more stable and is giving overall better results. This situation can be caused by far bigger inventory size and number of events in this dataset. Moreover, on SI-D and RR/5 our methods position themselves as the second-best ones with a minimal margin to kNN based methods, VSKNN and SKNN, respectively. For SI-D only the PREC@20 is lower, due to the fact of far better results of SKNN (which we double-checked for the correctness with such good results for both SI datasets). The Retail-Rocket dataset presents con-

Table 2. Results obtained during the experiments. The baseline SKNN, VSKNN and GRU4Rec+ values for RR/5 and RSC15/64 are taken from supplementary materials to [21]. Best results for each measure – dataset pair are in **boldface**, while the second bests are underlined; ▽ indicates the sort column. The SML naming convention is: SML-SessionEncoder-LossFunction, with: RNN and MaxPool denote encoders described in Sect. 4.2, and three loss functions: Contrastive, Triplet and smoothed NCA—NCAS.

Dataset	Method	MAP	PREC	▽REC	HR	MRR
SI-T	SML-TextCNN-Triplet	0.0407	0.0526	**0.7463**	**0.8525**	0.6050
	SML-RNN-Triplet	0.0407	0.0526	0.7462	0.8523	0.6015
	VSKNN	**0.0410**	0.0619	0.7455	0.8511	**0.6088**
	SML-MaxPool-Triplet	0.0400	0.0518	0.7368	0.8414	0.5974
	SML-MaxPool-NCAS	0.0389	0.0504	0.7195	0.8246	0.5857
	SML-RNN-NCAS	0.0389	0.0503	0.7193	0.8248	0.5910
	SML-TextCNN-NCAS	0.0386	0.0499	0.7152	0.8206	0.5844
	SPOP	0.0336	0.0437	0.6384	0.7317	0.5724
	SML-TagSpace-Triplet	0.0335	0.0435	0.6352	0.7311	0.5602
	GRU4Rec+	0.0318	0.0463	0.5948	0.7578	0.5437
	SKNN	0.0293	**0.4524**	0.5609	0.6533	0.5640
	POP	0.0192	0.0257	0.3600	0.3954	0.1369
	MARKOV-1	0.0376	0.0218	0.2433	0.2875	0.1965
SI-D	VSKNN	**0.0394**	0.1228	**0.6499**	**0.7484**	**0.4483**
	SML-RNN-Triplet	0.0374	0.0536	0.6401	0.7350	0.4468
	SML-TextCNN-Triplet	0.0371	0.0531	0.6375	0.7334	0.4445
	SML-MaxPool-NCAS	0.0360	0.0515	0.6215	0.7185	0.4379
	SML-RNN-NCAS	0.0355	0.0509	0.6153	0.7130	0.4358
	SML-TextCNN-NCAS	0.0346	0.0496	0.6033	0.6990	0.4171
	SKNN	0.0350	**0.1502**	0.5942	0.6878	0.4321
	SML-MaxPool-Triplet	0.0340	0.0489	0.5929	0.6822	0.3718
	SML-TagSpace-Triplet	0.0309	0.0445	0.5470	0.6261	0.3733
	SPOP	0.0285	0.0406	0.5180	0.5853	0.4263
	GRU4Rec+	0.0332	0.0680	0.4966	0.6450	0.3043
	MARKOV-1	0.0348	0.0515	0.2582	0.3038	0.1741
	POP	0.0122	0.0197	0.2040	0.2248	0.0655
RSC15/64	SML-RNN-NCAS	0.0358	0.0639	**0.5248**	0.6557	**0.2884**
	SML-MaxPool-NCAS	0.0355	0.0634	0.5213	0.6502	0.2841
	SML-TextCNN-NCAS	0.0351	0.0627	0.5145	0.6393	0.2766
	SML-RNN-Triplet	0.0348	0.0623	0.5126	0.6371	0.2824
	VSKNN	**0.0386**	**0.0928**	0.5009	**0.6961**	0.2879
	SML-MaxPool-Triplet	0.0337	0.0607	0.4975	0.6143	0.2680
	SKNN	0.0363	0.0881	0.4780	0.6423	0.2522
	GRU4Rec+	0.0285	0.0721	0.4009	0.6528	0.2752
	SML-TextCNN-Triplet	0.0244	0.0462	0.3793	0.4615	0.1389
	SML-TagSpace-Triplet	0.0218	0.0416	0.3398	0.4082	0.1396
	MARKOV-1	0.0333	0.0446	0.3011	0.3912	0.1771
	SPOP	0.0164	0.0318	0.2879	0.3464	0.2205
	POP	0.0063	0.0129	0.1075	0.1264	0.0292
RR/5	SKNN	**0.0283**	**0.0532**	**0.4704**	**0.5788**	0.3370
	SML-MaxPool-NCAS	0.0273	0.0443	0.4673	0.5692	0.3340
	VSKNN	0.0278	0.0531	0.4632	0.5745	0.3395
	SML-RNN-NCAS	0.0270	0.0437	0.4609	0.5600	0.3379
	GRU4Rec+	0.0272	0.0502	0.4559	0.5669	0.3237
	SML-RNN-Triplet	0.0270	0.0456	0.4542	0.5560	0.3605
	SML-TextCNN-NCAS	0.0264	0.0427	0.4516	0.5476	0.3197
	SML-MaxPool-Triplet	0.0245	0.0392	0.4357	0.5243	0.3451
	SPOP	0.0201	0.0331	0.3773	0.4614	**0.3985**
	SML-TextCNN-Triplet	0.0183	0.0293	0.3278	0.3988	0.2156
	SML-TagSpace-Triplet	0.0117	0.0192	0.2321	0.2817	0.2397
	MARKOV-1	0.0183	0.0428	0.1557	0.1964	0.1162
	POP	0.0009	0.0020	0.0181	0.0227	0.0058

sistent results with [21], where many new methods cope to beat SKNN. With SML-MaxPooling-NCAS, we get close to the position of being the leader.

Between the investigated encoders, we can observe from the results that a simple max-pooling performs well and falls very close to the best score for SI-* datasets. Intuitively, GRU and CNN based methods should be better in encoding longer sequences of actions, like RSC15/64 and RR/5 (see Fig. 1 (left)). However, this proved to be true only for RSC15/64 results, where CNN and RNN based methods are among the best ones. For RR/5 simple pooling with the proposed NCAS loss function is the best one from the SML method family. Additionally, in practical terms, CNN-based models can be preferred from GPU utilization perspective, as the architecture and many libraries are optimized for computer vision and image processing.

Table 3. Ablation results obtained for the RNN and MaxPool session encoders. Columns labels in order: (1) True/False is common embedding was used; (2) Sampler: SW – sliding window, Pos-Neg – session positive negative sampling as described in Sect. 4.3; (3) Triplet loss with: N – L2 normalization, M – 0.3 margin used, S – swaping anchor-session with positive item. Results are sorted by REC@20.

Comm.			RNN		MaxPool	
Emb.	Sampler	Loss	▽REC@20	MRR@20	REC@20	MRR@20
True	Pos–Neg	N-M	0.7435	0.5973	0.7402	0.5978
True	Pos–Neg	N	0.7377	0.5973	0.7340	0.5932
True	Pos–Neg	N-M-S	0.7371	0.5908	0.7359	0.5888
False	Pos–Neg	N	0.6565	0.5746	0.6508	0.5727
False	Pos–Neg	N-M	0.6341	0.5783	0.6192	0.5767
False	Pos–Neg	N-M-S	0.6192	0.5678	0.6247	0.5796
False	SW	N-M-S	0.0022	0.0006	0.0525	0.0191

Coverage and Popularity Bias. Similar to [20,21] we investigated the distribution of predicted items for the selected approaches. Interestingly, our metric learning based methods usually give wider spectrum of recommended items. Even checking simple statistic of overall unique items being recommended, for SI datasets our methods return almost twice as much unique items as VSKNN method (666 to 1,542 and 1,522 to 2,542 for a sample run, all items 2k, 3k respectively, see Table 1), while for RR/5 and RSC the difference is not so big (16,334 to 19,063, 12,232 to 11,216 for a sample run).

Ablation Study. To verify the impact of each component in our proposed solution, we run a series of experiments on SI-T dataset for two encoders: RNN and MaxPool, enabling each improvements one by one. The results with REC@20 and MRR@20 are shown in Table 3.

One of the first sampling methods evaluated with SML was a simple sliding window-based technique. For a defined number of events (padded if necessary), we take only the next following items as positive examples, and negative ones are randomly sampled. We quickly switched to sampling presented in Sect. 4.3, as we notice that the windowing technique is not reflecting how the system is utilized in real use cases. Specifically, for various sub-sequences from the beginning of a session, predictions are also required, disregarding the sliding window size. As the next step, we evaluated the impact of the inner elements from triplet loss, like normalization (which is very common), margin usage (which for some datasets are set to very small values), and swapping of anchor and positive elements. To our surprise, swapping is not always giving good results for a session-based recommendations setting.

A crucial role for improving our model was the use of common embeddings for both session encoder $\varphi(s_k)$ and items encoder $\omega(i_j)$ for the prediction. This lowered the number or all parameters to train and positively influenced the overall results. We think that even further improvements can be made to the proposed method by a more global network parameters search. But this was out of scope of our computational possibilities. Thus, we constrained some of the network's hyper-parameters that are related (e.g., GRU hidden state dimension and following feed-forward network dimension to be the same).

6 Conclusions

In this paper, we have presented a novel approach to session-based recommendations that utilizes concepts from the field of metric learning. The proposed method has a clear and modular architecture that combines session and item embeddings with a metric loss function. Each of these elements may be individually tweaked and thus defines a potential direction for further research. We test our approach against independent results obtained for strong baseline methods using a well-established evaluation procedure and receive state-of-the-art results. The analysis is also extended by ablation studies, which confirm that the proposed solution does not have unnecessary elements.

Our approach's main advantage is a modular design and extensibility that makes it possible to tweak its components to best match the dataset or incorporate some prior knowledge. Moreover, the fact that SML is based on principles originating from metric learning, many improvements from that field can still be transferred and evaluated for session-based recommendations. From usage perspective, our approach can be attractive in combination with existing pipelines (KNN recommendations) and libraries (optimized CNN).

We can identify two main weaknesses of our method. Firstly, sampling has a significant impact on the results both in terms of quality and computational efficiency, so careful GPU usage and memory management is required. Secondly, many improvements that can be taken for granted within computer vision do not necessarily improve the final model when combined with other elements for session-based recommendations, which was presented in the ablation study.

Although we achieved promising results with the current method, this work only touched the subject of applying metric learning to session-based recommendations, and much more is to be explored. Apart from the already mentioned embeddings, the positive/negative sampling strategy used during the training phase seems to deserve more attention. Based on good experimental results achieved by some baselines, an introduction of a missing users' actions time context into session-based recommendation also seems worth exploring. Further investigation of improvements in the deep metric learning field can result in even better session-based recommendations, and similar synergy can be found, like in the case of NLP.

Acknowledgments and Disclosure of Funding. We acknowledge the support from Sales Intelligence and co-funding by European Regional Development Fund, project number: POIR.01.01.01-00-0632/18.

References

1. Burges, C.J.C., Ragno, R., Le, Q.V.: Learning to rank with nonsmooth cost functions. Mach. Learn. **19**, 193–200 (2007). https://doi.org/10.1007/s10994-010-5185-8
2. Chen, H., Karger, D.R.: Less is more: probabilistic models for retrieving fewer relevant documents. In: SIGIR, pp. 429–436 (2006). https://doi.org/10.1145/1148170.1148245
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
4. Fehervari, I., Ravichandran, A., Appalaraju, S.: Unbiased evaluation of deep metric learning algorithms. arXiv preprint arXiv:1911.12528 (2019)
5. Goldberger, J., Hinton, G.E., Roweis, S.T., Salakhutdinov, R.R.: Neighbourhood components analysis. In: Advances in Neural Information Processing Systems, pp. 513–520 (2005)
6. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1735–1742 (2006)
7. Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737 (2017)
8. Hidasi, B., Karatzoglou, A.: Recurrent neural networks with top-k gains for session-based recommendations. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 843–852 (2018)
9. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. Technical report
10. Hidasi, B., Tikk, D.: General factorization framework for context-aware recommendations. Data Min. Knowl. Disc. **30**(2), 342–371 (2015). https://doi.org/10.1007/s10618-015-0417-y
11. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Feragen, A., Pelillo, M., Loog, M. (eds.) SIMBAD 2015. LNCS, vol. 9370, pp. 84–92. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24261-3_7
12. Jannach, D., Ludewig, M.: When recurrent neural networks meet the neighborhood for session-based recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, pp. 306–310. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3109859.3109872

13. Kaya, M., Bilge, H.Ş.: Deep metric learning: a survey. Symmetry **11**(9), 1066 (2019)
14. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
15. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426–434 (2008). https://doi.org/10.1145/1401890.1401944
16. Koren, Y.: Collaborative filtering with temporal dynamics. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 447–456 (2009)
17. Lim, D., Lanckriet, G.: Efficient learning of Mahalanobis metrics for ranking. In: International Conference on Machine Learning, pp. 1980–1988 (2014)
18. Liu, N., Zhao, M., Yang, Q.: Probabilistic latent preference analysis for collaborative filtering. In: Proceedings of the 18th ACM Conference on on Information and Knowledge Management, pp. 759–766 (2009). https://doi.org/10.1145/1645953.1646050
19. Liu, T.Y., et al.: Learning to rank for information retrieval. Found. Trends Inf. Retrieval **3**, 225–331 (2009)
20. Ludewig, M., Jannach, D.: Evaluation of session-based recommendation algorithms. User Model. User-Adap. Inter. **28**, 331–390 (2018). https://doi.org/10.1007/s11257-018-9209-6
21. Ludewig, M., Mauro, N., Latifi, S., Jannach, D.: Performance comparison of neural and non-neural approaches to session-based recommendation. In: RecSys 2019–13th ACM Conference on Recommender Systems, pp. 462–466. Association for Computing Machinery, Inc. (2019). https://doi.org/10.1145/3298689.3347041
22. McFee, B., Lanckriet, G.R.: Metric learning to rank. In: Proceedings of the 27th International Conference on Machine Learning (ICML 2010), pp. 775–782 (2010)
23. Movshovitz-Attias, Y., Toshev, A., Leung, T.K., Ioffe, S., Singh, S.: No fuss distance metric learning using proxies. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 360–368 (2017)
24. Musgrave, K., Belongie, S., Lim, S.N.: A metric learning reality check. arXiv preprint arXiv:2003.08505 (2020)
25. Ning, X., Karypis, G.: SLIM: Sparse LInear Methods for top-N recommender systems. In: Proceedings of the IEEE International Conference on Data Mining, ICDM, pp. 497–506 (2011). https://doi.org/10.1109/ICDM.2011.134
26. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4004–4012 (2016)
27. van den Oord, A., et al.: WaveNet: a generative model for raw audio (2016)
28. Paszke, A., et al.: Pytorch: an imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32, pp. 8024–8035. Curran Associates, Inc. (2019)
29. Rendle, S.: Factorization machines. In: Proceedings of the IEEE International Conference on Data Mining, ICDM, pp. 995–1000 (2010). https://doi.org/10.1109/ICDM.2010.127
30. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-thieme, L.: BPR : Bayesian personalized ranking from implicit feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, vol. cs.LG, pp. 452–461 (2009)

31. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized Markov chains for next-basket recommendation. In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, pp. 811–820. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1772690.1772773
32. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Advances in Neural Information Processing Systems, pp. 41–48 (2004)
33. Shi, Y., Karatzoglou, A., Baltrunas, L.: xCLiMF: optimizing expected reciprocal rank for data with multiple levels of relevance. In: Proceedings of the 7th ACM conference on Recommender systems, pp. 0–3 (2013)
34. Shi, Y., Karatzoglou, A., Baltrunas, L., Larson, M., Hanjalic, A.: GAPfm. In: Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, CIKM 2013, pp. 2261–2266 (2013). https://doi.org/10.1145/2505515.2505653
35. Shi, Y., Karatzoglou, A., Baltrunas, L., Larson, M., Hanjalic, A., Oliver, N.: TFMAP: Optimizing MAP for top-n context-aware recommendation. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 155–164 (2012). https://doi.org/10.1145/2348283.2348308
36. Smirnova, E., Vasile, F.: Contextual sequence modeling for recommendation with recurrent neural networks. In: Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems, DLRS 2017, pp. 2–9. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3125486.3125488
37. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: Advances in Neural Information Processing Systems, pp. 1857–1865 (2016)
38. Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., Jiang, P.: BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1441–1450 (2019)
39. Tang, J., Wang, K.: Personalized Top-N sequential recommendation via convolutional sequence embedding. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM 2018, pp. 565–573 (2018). https://doi.org/10.1145/3159652.3159656
40. Taylor, M., Guiver, J., Robertson, S., Minka, T.: SoftRank: optimizing non-smooth rank metrics. In: WSDM 2008, pp. 77–86 (2008). https://doi.org/10.1145/1341531.1341544
41. Twardowski, B.: Modelling contextual information in session-aware recommender systems with neural networks. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 273–276 (2016)
42. Wang, J., Zhou, F., Wen, S., Liu, X., Lin, Y.: Deep metric learning with angular loss. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2593–2601 (2017)
43. Wang, X., Han, X., Huang, W., Dong, D., Scott, M.R.: Multi-similarity loss with general pair weighting for deep metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5022–5030 (2019)
44. Weston, J., Bengio, S., Usunier, N.: WSABIE: scaling up to large vocabulary image annotation. In: Twenty-Second International Joint Conference on Artificial Intelligence (2011)

45. Weston, J., Chopra, S., Adams, K.: #TagSpace: Semantic embeddings from hashtags. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1822–1827. Association for Computational Linguistics, Doha (2014). https://doi.org/10.3115/v1/D14-1194
46. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2840–2848 (2017)
47. Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J.M., He, X.: A simple convolutional generative network for next item recommendation. In: Proceedings of the 12th ACM International Conference on Web Search and Data Mining, WSDM 2019, pp. 582–590 (2019). https://doi.org/10.1145/3289600.3290975
48. Yuan, T., Deng, W., Tang, J., Tang, Y., Chen, B.: Signal-to-noise ratio: a robust distance metric for deep metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4815–4824 (2019)
49. Yue, Y., Finley, T., Radlinski, F., Joachims, T.: A support vector method for optimizing average precision. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 271–278 (2007)

Machine Translation Customization via Automatic Training Data Selection from the Web

Thuy Vu(✉) and Alessandro Moschitti

Amazon Alexa AI, Manhattan Beach, CA, USA
{thuyvu,amosch}@amazon.com

Abstract. Machine translation (MT) systems, especially when designed for an industrial setting, are trained with general parallel data derived from the Web. Thus, their style is typically driven by word/structure distribution coming from the average of many domains. In contrast, MT customers want translations to be specialized to their domain, for which they are typically able to provide text samples. We describe an approach for customizing MT systems on specific domains by selecting data similar to the target customer data to train neural translation models. We build document classifiers using monolingual target data, e.g., provided by the customers to select parallel training data from Web crawled data. Finally, we train MT models on our automatically selected data, obtaining a system specialized to the target domain. We tested our approach on the benchmark from WMT-18 Translation Task for News domains enabling comparisons with state-of-the-art MT systems. The results show that our models outperform the top systems while using less data and smaller models.

Keywords: Web data · Language customization · Text classifier

1 Introduction

Industrial MT services have greatly impacted multiple commercial applications, e.g., Google Translate and Amazon Translate. It has also become an indispensable technological component worldwide during the current pandemic to disseminate COVID-19's public service announcements to the public [15]. The result has been collectively attained by leveraging Web data: training examples (parallel text) can indeed be automatically built by aligning sentences from multilingual pages, which naturally occur on the web [7,18,19,21].

The harvesting of parallel data from the web has been shown successfully by [4,18], resulting in highly heterogeneous collected data, as sampled from the

The original version of this chapter was revised: table 3 was not correct. This has been corrected. The correction to this chapter is available at https://doi.org/10.1007/978-3-030-72113-8_51

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 666–679, 2021.
https://doi.org/10.1007/978-3-030-72113-8_44

entire web. Thus, the distribution of the content is inevitably dominated by the commercial websites working in a multi-language setting. On the one hand, this distribution may reflect the average expected demand submitted to an MT service by web users; on the other hand, it can hardly capture the specificity of less represented domains. In particular, users working with domains that traditionally do not require multilingual content, e.g., documentation of local administration or businesses having no internationalization interest, may find a general-purpose translation inadequate.

For example, if we use general terms, such as *project meeting* and *sport meeting*, which occur in many websites, a standard MT system provides rather accurate Italian translations, *incontro di progetto* and *incontro sportivo*, respectively. However, if we try terms less frequent in multilingual web data, for example, *condo meeting* or *condominium meeting*, we may obtain the following wrong translations: *riunione del condominio* or *condominio incontro*, instead of the correct one, *riunione di condominio*[1]. In particular, the MT system cannot select the right preposition *di* since (i) the most typical Italian construction uses *del*, and (ii) *condo meeting* is infrequent in web parallel data. In contrast, *project meeting* is correctly translated in *incontro di progetto* by most MT services: we did not observe mistakes of the type *incontro del progetto* or a less used term *incontro progettuale.* We speculate that such term, being more frequent, is typically supported by more training examples.

Current MT systems deal with the problem of under-represented domains by averaging the patterns observed in all available domains. Thus, the bias in generating translation towards the populated domain persists. This causes a translation targeting low-frequent phrases to use irrelevant or inappropriate words. In extreme cases, such problems may create embarrassing biased translations [5]; for example, *pornographic domains* appear very frequently on the web [1], if not adequately filtered, common terms may be interpreted in a sex key.

This paper explores automatic customization/personalization of MT systems by automatically selecting training data *similar* to the text in a target customer application. Such data will carry terminology and syntactic constructions specific to the target domain.

Our main assumption, supported by general machine learning theory, is that we can customize neural network models by training them with this selected data. Such an approach can produce three main benefits:

- The MT system requires less data to learn to translate in the target domain than when using general data. Indeed, specific domains are characterized by less lexical variability due to the need to express specific concepts/situations. The use of less data produces efficiency benefits at training time, with possibly a better translation quality in the domain.
- The fine-tuning step with customized domain data can increase accuracy in translating text from such domain in neural MT. In particular, infrequent

[1] As of May 2020, Google Translate provided *riunione condominiale*, which, although correct, is a bit too formal term for this kind of meeting.

patterns with respect to the average web distribution will better emerge from the model in the target domain as they will occur relatively more often.
- A positive side effect of this approach is that specific data can automatically diminish the bias on undesired domains, e.g., political inclinations or explicit content, when operating in a critical setting, e.g., kid protected content. Indeed, amplifying the term distribution of the kid domain can help mitigate the impact of very different and undesired training data.

To customize an MT system on a target domain, we assume to know the monolingual data of the domain in advance. This is a realistic assumption as the customer can specify their target data/domain, e.g., providing their website or textual documentation. Simultaneously, the MT service provider can continue to refresh their parallel data repository asynchronously and periodically. Therefore, the *customization* process is reduced to selecting the parallel data portion similar to the one from the target domain to train/fine-tune the MT models on the target context. We propose the design of topical classifiers to recognize the target domain data among the extremely large web crawled data. We note three important aspects:

- First, the data provided for the customization domain does not need to be parallel. We only need monolingual text data similar to the target domain to train the topic classifier. This is very important, as acquiring parallel data can be a key limitation to any customization approach's applicability. In contrast, monolingual data can be easily acquired from the customer's website, documentation or other related data.
- Our classifier is built to predict webpages instead of sentences as carried out in previous MT domain adaptation works based on language model [2]. Using entire pages allows for reaching a high accuracy in selecting data potentially similar to the target data since the document content distribution is not sparse and richer than the content of individual sentences.
- The negative examples can be generated by randomly sampling webpages from the entire crawled data. Indeed, given the very low occurrence probability of the documents of the target domain in comparison with billions of pages in the crawled data, the number of false negatives would be extremely low.

We tested the following research questions:

$\mathbf{q}_1$: Can we build efficient document classifiers to select large training data for MT systems specific to target domains?
$\mathbf{q}_2$: Are the classifiers accurate enough to select training data for the target domain from web crawled data?
$\mathbf{q}_3$: Does the data selected by the classifiers produce improvement of the MT systems when tested on the target domain?

To answer the questions above, we compared our selection approach against the state-of-the-art MT systems of the WMT-18 News Translation benchmark. The results show that using the data selected by our classifier, we can train a much simpler model and still be on par with the state-of-the-art approaches, e.g., those proposed by RWTH and Microsoft Research. These use a Big Transformer and are much more expensive. Our results show that (i) our approach for selecting target data is effective; and (ii) it is possible to customize MT systems on a target domain, i.e., the news domain. Although wider experimentation over different domains of possibly different sizes is needed to claim that our is a general-purpose approach to MT customization and personalization, our paper provides examples in such directions, enabling promising future work. It also shows interesting evidence on the potential of IR techniques for converting web data in specific applications without going through knowledge-based methods.

2 Domain Customization Approach

Our approach consists in (i) acquiring monolingual data for a target domain; (ii) training a topic classifier for such domain, using the acquired data as positive examples and randomly sampled web data as negative examples; (iii) selecting parallel data of the target domain by applying the built classifier to the monolingual text part of the crawled data; (iv) training or fine-tuning the MT system on the data selected by the classifier; and finally (v) applying the trained MT system for user data.

We describe the details in the following subsections.

2.1 Components and Notation

Our model requires the following components:

- a general large repository $\mathcal{C}$ of crawled parallel data for MT training.
- Several domains $D_1^+, .., D_n^+$ for different applications, businesses, and users.
- A sampling procedure S to get the negative examples from $\mathcal{C}$ not in D_i^+, denoted $D_i^- = S\left(\mathcal{C}, D_i^+\right)$.
- A linear fast topic classifier R_{D_i}, which we will train on $D_i = \{D_i^+, D_i^-\}$.
- A vanilla state-of-the-art MT model, $T_{\mathcal{C}}$, to be trained on parallel data.

The customized MT system will then be T_{C_i}, trained on $C_i \subset \mathcal{C}$, where $C_i = R_{D_i}(\mathcal{C})$. Specifically, R_{D_i} selects relevant parallel data from $\mathcal{C}$ based on D_i characteristics. Note that R_{D_i} is trained using D_i^+ as positive examples and $D_i^- = S(\mathcal{C}, D_i^+) \subset \mathcal{C}$ as negative examples.

2.2 Customization Pipeline

Figure 1 describes our pipeline to build an MT system customized for a particular user/domain. The diagram displays three different processes: (i) the training of a classifier R_{D_i}, (ii) the data selection, (iii) the MT training, and (iv) the customized translation.

In the first phase, the user provides a sample of the *Target Data* constituted by monolingual documents. These are positive examples (blue squares) used to train a classifier for the target data. The negative examples (grey squares) are sampled from the *Heterogeneous Dataset* (parallel data crawled from the web).

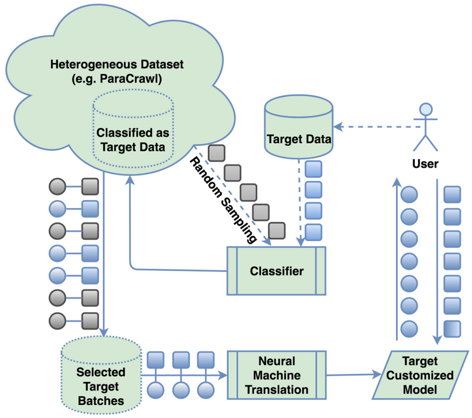

Heterogeneous Dataset (e.g. ParaCrawl)
Classified as Target Data
Target Data
User
Random Sampling
Classifier
Selected Target Batches
Neural Machine Translation
Target Customized Model

Fig. 1. Customization process of MT Systems

Table 1. Training data for WMT-18 for English–German

Corpus	Sent. (MM)
News commentary v13	0.3
Rapid (press releases)	1.3
Common crawl	1.9
Europarl v7	2.4
ParaCrawl (Zipporah)	40.6
ParaCrawl (BiCleaner)	27.7

In the second phase, the trained classifier produces a classification score for all *Heterogeneous Dataset* documents. The classification is done by exploiting only the monolingual side of the parallel data (in the same language of the target domain data). Although the Heterogeneous Dataset can be potentially very large, the classifier runs in linear time and can be parallelized.

In the third phase, the pairs of parallel documents, i.e., the circle and square pairs, are ranked with respect to the classifier score. The top k *Selected Target Batches* are split in pair of parallel sentences, and used to train the Neural MT model. Note that using ranked data we (i) avoid to tune up a classification threshold, which can be rather challenging as it requires the annotation of crawled data; and (ii) can select higher quality data from the top until we need or until the MT system does not improve anymore.

Finally, the users can apply the *Target Customized Model* (MT system) on their new monolingual text and receive translated data.

2.3 Target Data Classifier

As we need to process millions of instances, we implement our standard text classifier with Support Vector Machines (SVMs). As previously mentioned, the positive examples are created by randomly sampling a fixed amount of text from the target data provided by the customer. In contrast, the negative examples are randomly sampled from the heterogeneous background dataset.

The instance representation is based on the bag-of-word model, using the weighting scheme for the terms described below. Given a document d, the term frequency tf of a word $\omega_i \in d$ is normalized by the following equation:

$$tf\left(\omega_1^n, d\right) = \frac{count\left(\omega_1^n, d\right)}{\max_{\left(\overline{\omega}_1^n, \bar{d}\right)} count\left(\overline{\omega}_1^n, \bar{d}\right)}$$

where, $count\left(\omega_i, d\right)$ is the number of ω_i occurrences in d.

In general, the classifier scores indicate the likelihood of a text sampled from a source to be in the same domain of the target data.

2.4 Selection Approach

In principle, a binary topic classifier would be appropriate to select relevant data. However, estimating the threshold associated with an effective F1 could be cumbersome as we do not have a development set reflecting the target data required by the MT system. Thus, we do not even know the amount of the needed data and the Precision required to train the MT system effectively. Therefore, instead of a classifier, we use a ranker. This can be formally defined as a function

$$R : \mathcal{C} \rightarrow \mathcal{P}(\mathcal{C}),$$

which takes the set of documents, $\mathcal{C} = \{d_1, .., d_{|\mathcal{C}|}\}$, and returns a subset of size k, i.e., $R(\mathcal{C}) = [d_{i1}, ..., d_{ik}]$. To implement the reranker, we can still use a binary SVM classifier, which will learn a point-wise reranker: this outputs a score $s(\boldsymbol{d}) = \boldsymbol{w} \cdot \boldsymbol{d} + b$. The ranker is supposed to compute the set of indices as $[i1, ..., ik] = \text{k-argmax}_i \; s(\boldsymbol{d}_i)$, where k-argmax returns the indices of the top scored k documents.

R selects domain data from a heterogeneous dataset (e.g., the crawled data) based on the classifier's scores when applied to the monolingual documents. The top k documents associated with their parallel counterparts are selected for training, or fine-tuning, the MT systems.

3 Experiments

We demonstrate the effectiveness of our proposed method step-wise in a typical pipeline to build state-of-the-art MT models using data selected by our proposed classifier. For this purpose, we first study the performance of the domain classifier separately. We then show its concrete impact in training both standard MT systems and a large-scale well-known MT benchmark, the WMT-18 News Translation Shared Task. This experiment enables us to explain empirically the performance of our approach in comparison with other MT systems trained on the exact benchmark setting and using the same experimental dataset. The setting includes a large, noisy parallel data crawled from the web.

3.1 Experimental Setup

We use the evaluation setting of the News Shared Task from WMT-2018 [6]. In particular, we carry out experiments on two translation tasks: English–German and German–English.

Data. The data provided by WMT-2018 is summarized in Table 1. The first four datasets are considered of high quality or *clean* in this experiment. The next two datasets, newly introduced as part of the WMT-2018 benchmarks, are ParaCrawl cleaned by two different filtering methods. They are parallel sentences extracted automatically from crawled web data and subsequently cleaned by Zipporah and BiCleaner.

In our experiment, we propose the following setting to implement our diagram in Fig. 1:

- The `News Commentary v13`'s text in English side is used as Target Data as we set news translation as the target domain application.
- The `ParaCrawl (BiCleaner)` data is considered as the Heterogeneous Dataset, given its web nature, large size, and noise quality.
- Our neural MT models are trained with all clean data (the first four datasets) in Table 1 and an automatically selected portion from the Heterogeneous Dataset.

It should be noted that this data comes in the form of individual paired-sentences. We simulated documents by grouping sentences in batches to train our document classifier. The procedure is a key factor as we can (i) avoid possible topical bias regarding individual documents but (ii) also capture sufficient thematic or stylistic information of the target domain. In other words, we do not classify individual sentences but sentence batches.

Domain Classifier Data. We generate positive and negative examples for building a classifier for news domains as follows:

- for positive examples, we form an example by randomly selecting n English sentences without repetition from the news data, `News Commentary v13`. The example may contain sentences from different source documents yet they are from the news domain. This helps capture the journalistic signal in news reports while discouraging possible topical text from a particular story or section.
- For negative examples, we alternatively sample from the `ParaCrawl` dataset cleaned by `BiCleaner` while keeping the size of n sentences per example. Even though journalistic text can appear in the example, the probability with respect to all the other content of the web makes the contribution of false-negative examples negligible.
- We also set the ratio between negative/positive to 2:1 to have enough positive examples.

3.2 Domain Classifier Results

We study the performance of the proposed classifier in this section. Specifically, we set n to 100 for the number of sentences per example. This results in 2,828 and 5,656 positive and negative examples, respectively, from `News Commentary v13`.

We apply a split of 30% for training and 70% for testing. As the original sentences from `News Commentary v13` are distinct, the generated examples for training and testing should also share no content overlapping. We used SVMs to build the classifier/reranker. We set the probability parameter to enable Platt scaling calibration on the classifier score. The feature set consists of 70,000 most frequent words with stop-words being removed in the dataset.

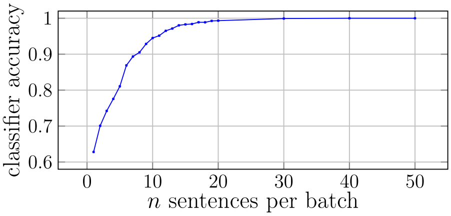


Fig. 2. Accuracy of the classifier in different setting of n.

Table 2. Accuracy comparison of the proposed method and other baselines.

	Accuracy
Sentence-based classifier	62.8%
Batch-based sentence majority	77.8%
Batch-based classifier (our method)	99.0%

We use the default setting for the other SVM parameters of the `sklearn.svm` toolkit. We compare the effectiveness of our proposed selection method, *Batch-based Classifier*, with two related yet different configurations as baselines:

- *Sentence-based Classifier*: we build a classifier similar to the above configuration, except for the size n of each batch set to 1. This is equivalent to building a classifier, where the documents are constituted by just individual sentences.
- *Batch-based Sentence Majority*: we classify a batch of $n = 100$ sentences via majority voting, i.e., we apply the *Sentence-based Classifier* to all sentences of the batch, and we classify the batch according to the majority of positive or negative classifications.

The accuracy of the classifier and the baselines is presented in Table 2. Training and classification at document level is much more advantageous than the one at sentence level. Because the word distribution from a larger text is more statistically reliable – the basic theory of large samples provides support for such intuition, where *the samples* in our case are constituted by set of words. Note that the distribution of positive and negative batches is still 1:2, i.e., the same sentence distribution; thus the results are comparable.

To better show the intuition that the larger is the sentence batch, the higher is the accuracy, we have plotted the accuracy of our batch classifier with respect to the batch size in Fig. 2. We see that as soon as the batch content is larger than 10 sentences, the accuracy exceeds 95%. With batches of 20 sentences or more, the classifier reaches perfect accuracy. This can be explained by the fact that random documents from the Web (approximated by the ParaCrawl) are statistically very different from those of the target domain. At the same time, we built our training and test sets with a positive/negative example distribution of 1:2. The classification accuracy over the entire ParaCrawl, which shows a much more skewed distribution can be significantly lower. However, the purpose of this experiment was to show that we can build an accurate classifier. Given the above positive result, we can use the classifier for reranking our data. The effectiveness of the classifier in selecting data will be shown in the next sections.

3.3 Machine Translation Results

We study the impact of the proposed data selection approach in MT tasks. In particular, we conducted experiments to address the following two questions:

(i) Can the classifier select relevant data for the target domain?
(ii) Can the selected data be used to improve the state-of-the-art in MT on a specific domain?

Table 3. BLEU-based evaluation of CSE on WMT-18

ParaCrawl	Buckets		Clean & Bucket	
	2017	2018	2017	2018
0%	–	–	27.2	32.4
0%–25%	**28.1**	**34.3**	**29.8**	**36.2**
25%–50%	23.4	27.8	27.3	32.8
50%–75%	12.7	14.7	25.2	30.3
75%–100%	5.8	6.6	25.0	29.7
0%–100%	23.7	29.21	28.22	34.41

Table 4. Average-z of human evaluation scores for WMT-18 systems, including 5 anonymized translation services.

System	EN-DE	DE-EN
RWTH Aachen	–	0.413
Microsoft Research	0.551	–
University of Cambridge	0.537	0.395
University of Edinburgh	0.352	0.261
JHU MT Systems	0.377	0.317
Universitat Politècnica de València	–	0.321
ONLINE-A	0.561	0.346
ONLINE-B	0.396	0.310
ONLINE-C	0.060	0.268
ONLINE-D	−0.385	−0.296
ONLINE-E	−0.416	−0.074

To reliably answer the second question, we used the WMT-18 benchmark as it is well-known both in academic and industrial MT communities. We performed two main experiments: the first aims at exploring the quality of the candidates with respect to their position in the rank generated by the topic classifier. The second aims at measuring the potential of our selected data with respect to the state of the art.

Data Quality in the Ranked Examples. In these experiments, we used an efficient MT approach, namely, the LSTM cell by [3,14], as we were interested in relative values of the accuracy and carrying out a fast experimentation.

We order documents and thus sentences in ParaCrawl in the descendent order of the classifier score described in Sect. 2.4. We then split the rank into four buckets of the same size. We used one bucket at a time to train an MT model using the default setting of Sockeye[2] (LSTM cell). We evaluated such models against the standard WMT-2017 and WMT-2018 test sets, using BLEU as our evaluation metric. The results are reported in Table 3, under the column *Buckets*, using the evaluation tool, `sacrebleu` [17]. Each row, labeled with an interval percentage, corresponds to a different MT system trained with the rank interval data. As expected, the systems trained with higher ranked data show a larger BLEU score. The system trained with the bottom bucket shows a very low performance. It is also interesting to compare with the second column showing the results using the 6M clean sentence pairs from WMT-2018: the MT system trained with our selected data in the first interval, 0%–25%, shows a higher accuracy. This is important as the crawled data is generally rather noisy, meaning that our classifier can select clean MT data.

Additionally, we combined the bucket data with the clean WMT-2017/2018 data. The results are reported under column *Clean & Bucket*, starting from the second row. We note that the combination can improve the system using just the clean data, e.g., from 29.8 to 36.2 on the WMT-2018 test set. This confirms that our approach can improve MT systems. The combination of clean data with all the other buckets also does not improve the clean data-based system or decreases accuracy. In particular, when all crawled data is used together with the clean data, the MT systems improve their accuracy only 50% of what they do when trained on our smaller selected data.

WMT-18 Shared Task: Machine Translation of News. To compare with the state-of-the-art, we needed a powerful model, which can approach the results of the best MT systems. Thus, we used the Transformer [20], a more expensive model in terms of computation than the LSTM-based but it is still largely less costly than the top performant systems in the WMT competition.

We trained our MT model with the clean data and the top 6M pairs from ParaCrawl selected with our classifier. We follow the typical model building pipeline described in [12]. We use the setting from Marian toolkit[3]. Table 5 shows the result. We note that our model, which uses a relatively much simpler neural network than the state-of-the-art approaches, e.g., RWTH and Microsoft Research (using a Big Transformer), is just 1.6 BLEU score points behind. This shows that our approach can build more efficient models with less data since the crawled data we used is closer to the target domain.

Discussion. Besides automatic evaluation, the WMT-18 Shared Task also conducted a human evaluation of the participating systems. Specifically, translations from individual systems were manually validated by assessors, comprised of both researchers and crowd-sourced workers from Mechanical Turk. The assessment was based on how well a translation replicates the meaning of the reference translation. The scores from an assessor are first standardized individually, according

[2] https://github.com/awslabs/sockeye [11].

[3] https://github.com/marian-nmt/marian-examples/tree/336740065d9c23e53e912a1befff18981d9d27ab/wmt2017-transformer.

to their overall mean and standard deviation. Then, the average standardized scores for translations rated by an assessor for a system are computed. The overall score, Average z, is finally computed as the average of its scores from the assessors.

Table 5. Comparison of our model with the results reported by WMT-18 using the BLEU score.

System	Clean pairs	Noisy pairs	Monolingual for back-translation	Model	EN-DE	DE-EN
RWTH Aachen	6M	18M	18M	Trans.-Big	–	48.4
Microsoft Research	6M	10M	10M	Trans.-Big	48.3	–
University of Cambridge	6M	15M	20M	Trans.-Big	46.6	46.8
University of Edinburgh	6M	4M	20M	Trans.-Base	44.4	43.9
JHU MT systems	6M	All	UNK	RNN	43.4	45.3
Universitat Politècnica de València	6M	10M	20M	Trans.-Base	–	45.1
Our model	6M	6M	10M	Trans.-Base	46.7	46.1

Table 4 shows a human evaluation carried out by WMT-2018 organizers. They consider the systems in Table 5 and five anonymized commercial translation services, named ONLINE-A, B, C, D and E. We note that the ranking produced by the manual evaluation is close to the one automatically carried out with BLEU score reported in Table 5. Most critically, the table also shows that almost all online services underperform the top MT participant systems, which are comparable to our approach.

This is an important comparison as it indirectly shows that the results of our approach are better than those of the services mentioned above. Additionally, the news domain is not under-represented in MT domains, suggesting that a larger gap between our approach and MT services could be observed when dealing with more specific domains. In other words, translations from online services may consider moving toward customization, not only for better translations [9] but also for better satisfying requests of different groups of users.

4 Related Work

Previous work has studied methods for selecting effective data for MT. Some of the approaches include:

- perplexity-based selection: this approach ranks sentences based on the perplexity scores given by a targeted language model [10,16,22]. Only sentences within a certain perplexity threshold are selected.

– Language model and translation model combination: this approach ranks sentence-pairs by both the target language model and the translation model trained by general and specific data [2,13]. The selection is based on the total cross-entropy difference from both sides.

The core difference with our proposed approach is that we use (i) documents (or at least grouped-sentences) rather than individual sentences [8], and (ii) negative examples randomly selected from a heterogeneous dataset from the web.

In contrast with methods aiming at selecting sentences with the same language models, our approach selects documents and thus sentences that belong to the same topics, i.e., approaching the data distribution of specific domains. In particular, the use of statistics of an entire large document enables a much more robust approach and an accurate selection of data related to the target domain.

Finally, the role of negative examples is also fundamental as patterns present in negative documents are automatically filtered out by the machine learning approach together with the negative sentences.

The business advantage of our approach is clear: given a customer request, we only require their monolingual examples in the target domain, e.g., their websites, documentations, etc. A classifier for selecting similar training data can be automatically built on their data, as we generate negative examples from the crawled data. We then apply the classifier to select parallel data from a large repository of parallel data from the Web. Finally, we train an MT model using the selected data, to obtain a system specialized on the target customer data. This model, being trained on the target domain data, will generate translations using style and text construction typical from the target domain. In addition to language customization our approach also enables the use of smaller models, which have less hardware requirement to fulfill the needs of small or medium enterprises.

5 Conclusion

We have proposed our strategy for customizing MT systems' training using data selected from a heterogeneous parallel corpus. This way, customers can provide their data as examples of the text on which the MT system should provide high accurate translations. Specifically, we propose a supervised classifier trained on a small sample of monolingual target data. The classifier makes predictions per batch of sentences to better capture the target domain's patterns and terms.

We show the effectiveness of our method by comparing it with the state-of-the-art on well-known MT benchmarks. The results demonstrate that we can achieve competitive performance on WMT-18 Shared Tasks, but our approach only requires a small monolingual sample of the target data. Finally, we believe our proposed method can be applied to customize other IR or Natural Language Processing applications exploiting Web data and IR techniques.

In the future, we are exploring the possibility to apply our method for selecting locale-sensitive training data and thus building locale-specific translation engines. We will also explore other data dimensions that are orthogonal to the topical categories. Indeed, we can build a classifier to select particular text styles, ranging from formal (thus building MT systems for translating formal documents), to informal languages, e.g., for more colloquial or less formal text applications, such as blog translation. We may also be able to target sublanguages and jargons as we can train the MT system with such kind of data, e.g., forums, or non native speaker languages. We can also build more powerful data selection classifiers that can be learned on customer data in different languages, i.e., neural multilingual topic/style classifiers.

References

1. Ahmed, F., Shafiq, M.Z., Liu, A.X.: The internet is for porn: measurement and analysis of online adult traffic. ICDCS **2016**, 88–97 (2016)
2. Axelrod, A., He, X., Gao, J.: Domain adaptation via pseudo in-domain data selection. EMNLP **2011**, 355–362 (2011)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: ICLR 2015 (2015)
4. Bañón, M., et al.: ParaCrawl: web-scale acquisition of parallel corpora. ACL **2020**, 4555–4567 (2020)
5. Biesinger, R.: Is your software racist? Politico (2018). https://www.politico.com/agenda/story/2018/02/07/algorithmic-bias-software-recommendations-000631
6. Bojar, O., et al.: Findings of the 2018 Conference on Machine Translation (WMT 2018), Belgium, Brussels, pp. 272–307 (2018)
7. Buck, C., Koehn, P.: Quick and reliable document alignment via TF/IDF-weighted cosine distance. In: WMT 2016, Berlin, Germany, pp. 672–678 (2016)
8. Chen, B., Huang, F.: Semi-supervised convolutional networks for translation adaptation with tiny amount of in-domain data. In: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning. Association for Computational Linguistics, Berlin (2016)
9. Dinu, G., Mathur, P., Federico, M., Al-Onaizan, Y.: Training neural machine translation to apply terminology constraints. In: ACL 2019, Florence, Italy, pp. 3063–3068 (2019)
10. Gao, J., Goodman, J., Li, M., Lee, K.F.: Toward a unified approach to statistical language modeling for chinese. In: ACM TALIP (2002)
11. Hieber, F., et al.: Sockeye: a toolkit for neural machine translation. CoRR (2017)
12. Junczys-Dowmunt, M., et al.: Marian: fast neural machine translation in C++. CoRR (2018)
13. Liu, L., Hong, Y., Liu, H., Wang, X., Yao, J.: Effective selection of translation model training data. In: ACL 2014, Baltimore, Maryland (2014)
14. Luong, T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: EMNLP 2015, Lisbon, Portugal (2015)
15. McCulloch, G.: Covid-19 is history's biggest translation challenge. wired.com (2020). https://www.wired.com/story/covid-language-translation-problem/
16. Moore, R.C., Lewis, W.: Intelligent selection of language model training data. In: ACL 2010, Uppsala, Sweden, pp. 220–224 (2010)
17. Post, M.: A call for clarity in reporting BLEU scores. In: WMT 2018 (2018)

18. Smith, J.R., Saint-Amand, H., Plamada, M., Koehn, P., Callison-Burch, C., Lopez, A.: Dirt cheap web-scale parallel text from the common crawl. In: ACL 2013, Sofia, Bulgaria, pp. 1374–1383 (2013)
19. Uszkoreit, J., Ponte, J., Popat, A., Dubiner, M.: Large scale parallel document mining for machine translation. In: COLING 2010, Beijing, China (2010)
20. Vaswani, A., et al..: Attention is all you need. In: NIPS 2017, pp. 5998–6008 (2017)
21. Vu, T., Moschitti, A.: CDA: a cost efficient content-based multilingual web document aligner. In: EACL 2021 (2021)
22. Yasuda, K., Zhang, R., Yamamoto, H., Sumita, E.: Method of selecting training data to build a compact and efficient translation model. In: IJCNLP 2008 (2008)

GCE: Global Contextual Information for Knowledge Graph Embedding

Chen Wang(✉) and Jiang Zhong

College of Computer Science, Chongqing University,
Chongqing 400030, People's Republic of China
{chenwang,zhongjiang}@cqu.edu.cn

Abstract. Most existing large-scale knowledge graphs are suffering from incompleteness, and many research efforts have been devoted to the task of knowledge graph completion. One popular approach is to learn low-dimensional representations for all entities and relations, and then employ them to infer new facts. However, we find that most of the current knowledge graph embedding models are lack of suitable strategy to utilize global contextual information. In this paper, we propose an embedding model, named GCE, to explore the capability of global contextual information to the task of knowledge graph completion. In GCE, we carefully design a global contextual information module with the attention mechanism. This module could aggregate global contextual information adaptively, thus enhancing feature representation for knowledge graph completion. To demonstrate the effectiveness of our proposed GCE, we conduct extensive experiments on two benchmark datasets FB15k-237 and WN18RR. Experimental results show that GCE achieves competitive results compared with the existing state-of-the-art embedding models on both datasets. The results validate our central hypothesis – that global contextual information is beneficial to knowledge graph completion performance.

Keywords: Global contextual information · Knowledge graph embedding · Knowledge graph completion · Link prediction · Attention mechanism

1 Introduction

Over the recent years, many famous large-scale knowledge graphs (KGs), such as Wordnet [19], Freebase [4], NELL [6], DBpedia [1], and YAGO3 [18], have been developed to store huge structured information about common facts. KGs can be represented as multi-relational directed graphs, in which the nodes represent entities and edges represent different relationships between entities. The information of entities and relations is modeled in the form of triples (subject, relation, object), denoted as (s, r, o), e.g., (Sydney, city of, Austria). These KGs are important resources for many information applications, such as semantic search [3,14,45], recommendation [33,41], data integration [15,26], question answering [39] and information retrieval [10,44].

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 680–693, 2021.
https://doi.org/10.1007/978-3-030-72113-8_45

Although these large-scale KGs have already contained millions of triples, they are still suffering from incompleteness, missing a lot of valid triples [37]. For example, in Freebase more than 60% of the person entities are missing their birthplaces. To this end, many research efforts have been devoted towards correcting errors as well as adding missing facts to KGs, commonly known as the task of knowledge graph completion or Knowledge Graph Augmentation. We can complete existing KGs by extracting new facts from external sources, such as Web corpora, or by inferring missing facts from those already in the KGs. The latter approach, called Link Prediction (LP), is the focus of our research.

In general, most existing LP models are based on knowledge graph embedding. These models first encode the semantics of entities and relations into a continuous low-dimensional vector space (called embedding), and then employ them to infer whether new triples are valid or not [5]. These embedding based models are broadly classified as translational distance models [5,12,16,36], compositional based models [23,30,38], graph based models [20,25,27] and convolutional neural network (CNN) based models.

TransE [5], TransH [36], TransR [12] and TransD [12] are examples of translational distance models. These models employ translational characteristic to model relationships between entities and are faster with fewer parameters. Compositional based models, such as RESCAL [23], DISTMULT [38], HOLE [24] and ComplEx [30], could capture rich interactions with a large number of parameters. Graph based models, such as R-GCN [25], SACN [27] and KBGAT [20], could capture the structure information and node attributes from relational data, and are always applied as an encoder. CNN based models could learn more expressive embedding due to their parameter efficiency and consideration of complex relations.

Recently, CNN based models have received significant research attention in knowledge graph embedding learning. ConvE [9] is the first model applying convolutional filters for the knowledge graph completion task. In ConvE, only the embedding of subjects and relations are reshaped and then fed to the convolution layer. To capture global relationships and translational characteristics between entities and relations, ConvKB [21] models the relationships among same dimensional entries of entity and relation embeddings. In CapsE [22], authors applied capsule network for modeling relationship triples by replacing the fully connected layer in ConvKB with two capsule layers. To further increase feature interactions between entity and relation embeddings, InteractE [31] augments the expressive power of ConvE through feature permutation, "checkered" feature reshaping, and circular convolution. From the above methods, it is obvious that to enhance discriminative feature representations is beneficial for the task of knowledge graph completion.

In this paper, we find that one major issue for current CNN based models is lack of suitable strategy to utilize global contextual information. Many link prediction errors for triples with complex relations are partially or completely related to global contextual information. To address the above issue, we propose GCE, a novel CNN based KG embedding model which aims to sufficiently incorporate the global contextual information. The local and global features together make the final prediction more reliable. Our main contributions in this paper are as follows:

- We propose GCE, a CNN based embedding model, to capture suitable global contextual information and selectively enhance discriminative feature representations. To the best of our knowledge, our work is the first consideration of exploring global contextual information to knowledge graph completion.
- An global contextual information encoding module with the attention mechanism is proposed to learn rich global contextual information over local features. This module could adaptively aggregate global contextual information, thus enhancing feature representations for knowledge graph completion.
- We evaluate our proposed GCE for knowledge graph completion on two benchmark datasets including FB15k-237 [29] and WN18RR [9]. Experimental results demonstrate the effectiveness of our proposed GCE.

The rest of the paper is organised as follows. In Sect. 2 we provide a review of related work on knowledge graph completion and global contextual information modelling. In Sect. 3 we describe our GCE in detail. In Sect. 4 we report the experimental results and analysis. Finally, in Sect. 5 we draw our conclusion and future research direction.

2 Related Work

In this section, we briefly review some related works on knowledge graph embedding and global contextual information.

2.1 Knowledge Graph Embedding

Recently, knowledge graph embedding learning has been an active research area with applications directly in knowledge graph completion and relation extraction. Several knowledge graph embedding models have been proposed. These methods can be broadly classified as: (i) compositional based models, (ii) translational distance models, (iii) graph based models, and (iv) CNN based models.

RESCAL [23], DISTMULT [38], HOLE [24] and ComplEx [30] are the examples of compositional based models. RESCAL use tensor product to capture rich interactions, but require a large number of parameters. To reduce model complexity, DISTMULT simplifies RESCAL by restricting relation matrices to *diagonal matrices*. ComplEx generalizes DISTMULT by using complex embeddings and Hermitian dot products to better model asymmetric relations [34]. HOLE combines the expressive power of RESCAL with the efficiency and simplicity of DistMult with circular correlation of entity embeddings.

In comparison, translational distance models like TransE [5], TransH [36], TransR [16] and TransD [12] are arguably simpler models. TransE considers the translation operation between head and tail entities for relations. TransH enables an entity to have distributed representations in different relations.

TransR projects entities from entity space to corresponding relation space and then builds translations between projected entities. These translational distance models are faster with fewer parameters and relatively easier to train with competitive performance.

R-GCN [25], SACN [27] and KBGAT [20] are examples of graph based models. These models apply graph neural networks to relation data and learn features of entities and relations in a joint manner [20,27]. R-GCN applies a convolution operation to the neighborhood of each entity and assigns them equal weights. SACN applies a weighted graph convolution network (WGCN) as the encoder. It weighs the different types of relations differently when aggregating connected entities. KBGAT extends graph attention mechanisms to capture entity and relation features in a multi-hop neighbourhood of a given entity. These GCN based models could leverage graph structure and capture semantically rich, latent relations among the triples.

Recently, CNN based models have received significant research attention in knowledge graph embedding learning. ConvE [9] uses 2-D convolution over embeddings to predict links. It comprises of a convolution layer, a fully connected projection layer and an inner product layer for the final predictions. ConvKB [21] applies 1-D convolution over same dimensional entries of an embedding triple, so that it could capture transitional characteristic between entities and relations. CapsE [22] extends ConvKB by using a capsule network to model relationship triples for knowledge graph competition. InteractE [31] extends ConvE by further increasing feature interactions between entity and relation embeddings through feature permutation, “checkered” feature reshaping, and circular convolution. These models are parameter efficient, fast to compute and robust to control over-fitting.

2.2 Global Contextual Information

Understanding and utilizing global contextual information is vitally important for many tasks, such as machine translation, information extraction, visual question answering, image caption, video classification, semantic segmentation, object detection, and image de-raining [2,8,11,17,35]. For text and sequences data, we commonly apply recurrent neural networks, such as RNN, GRU, and LSTM, to capture global contextual information. Then methods based on attention mechanism [2,17,32] were proposed for better global contextual information. For computer vision task, several model variants were proposed to enhance contextual aggregation. Deeplabv3+ [7] proposes an atrous spatial pyramid pooling (ASPP) module to capture multi-scale contextual information. PSPNet [43] designs a pyramid spatial pooling module to collect the effective global contextual information. OCNet [40] adopts self-attention mechanism with ASPP(or PPM) to exploit the context dependencies. EncNet [42] introduces a channel attention mechanism to capture global context. DANet [11] proposes two attention modules to enhance the discriminant ability of feature representations.

Different from previous works, we explore the global contextual information in the task of knowledge base completion. We carefully design a global contextual information module to adaptively aggregate global contextual information and enhance feature representations. Comprehensive experimental results verify the effectiveness of our proposed method.

3 The Proposed GCE

In this section, we first provide some notations and definitions used in the rest of the paper, followed by a general framework of our proposed GCE. Finally, we introduce the global contextual information module in detail followed by how to combine them together for knowledge graph embedding learning.

3.1 Background

Given a Knowledge Graph (KG) $\mathcal{G} = (\mathcal{E}, R)$, where $\mathcal{E}$ and $\mathcal{R}$ denote the set of entities (nodes) and relations (edges), respectively. The task of *link prediction* is to predict new triple (s', r', o'), where $s', o' \in \mathcal{E}$ and $r' \in \mathcal{R}$, based on the existing triples in KGs. Formally, the link prediction task can be considered as a ranking problem. Knowledge graph embedding models try to learn effective representations of entities, relations and a scoring function $f(s, r, o)$, such that for a given triple $t = (e_s, e_r, e_o)$, $f(t)$ is defined to measure the validity of triple t. We generally assume that valid triples probably receive higher scores than invalid triples [9,21,22].

3.2 Overview of Our GCE

Following ConvKB and CapsE [21,22], we view each embedding triple (e_s, e_r, e_o) as a matrix $A = [\boldsymbol{e}_s, \boldsymbol{e}_r, \boldsymbol{e}_o] \in \mathbb{R}^{k\times 3}$. In the convolution layer, we use a filter $\boldsymbol{\omega} \in \mathbb{R}^{1\times 3}$ to capture some relation-specific attribute of triples. We denote $\boldsymbol{\Omega}$ as the set of filters and N as the number of filters, thus we have feature maps as a matrix $F \in \mathbb{R}^{k\times \mathsf{N}}$, for which each feature map can capture one single characteristic among triples at the same dimension [22]. These features are local features and lack of enough associations among each other. To address this issue, we try to explore global contextual information by building associations among features with the attention mechanism.

We illustrate our GCE in Fig. 1, where embedding size $k = 4$, the number of filters $\mathsf{N} = 5$. We design a global contextual information module with attention mechanism to draw global contextual information over local features, thus obtaining better representations for link prediction. Specifically, we feed the original features into the global contextual information module and generate enhanced features through the following three steps. The first step is to calculate the attention matrix from the original features. Next, we perform a matrix multiplication between the original features and the attention matrix. Third, we perform an element-wise sum operation on the above multiplied resulting matrix and original features to obtain the enhanced representations reflecting global contextual information.

Finally, we feed the enhanced features into two single capsule layers following CapsE [22]. And the length of the output from second capsule layer is used as the score for the input triple. Formally, we define the score function f for the input triple (s, r, o) as follows:

$$f(s, r, o) = ||capsnet(G(g([e_s, e_r, e_o] * \boldsymbol{\Omega})))|| \tag{1}$$

where Ω is shared parameters in the convolution layer; $*$ denotes a convolution operator; G means the global contextual information with attention operator; *capsnet* denotes a capsule operator. We use Adam optimizer [13] to train GCE by minimizing the loss function [21,22,30] with L_2 regularization as follows:

$$\mathcal{L} = \sum_{(s,r,o)\in\{\mathcal{G}\cup\mathcal{G}'\}} \log\left(1 + \exp\left(-t_{(s,r,o)} \cdot f(s, r, o)\right)\right) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2 \tag{2}$$

in which,

$$t_{(s,r,o)} = \begin{cases} 1 \text{ for } (s, r, o) \in \mathcal{G} \\ -1 \text{ for } (s, r, o) \in \mathcal{G}' \end{cases} \tag{3}$$

here $\mathcal{G}$ and $\mathcal{G}'$ are collections of valid and invalid triples, respectively. $\mathcal{G}'$ is generated by corrupting valid triples in $\mathcal{G}$.

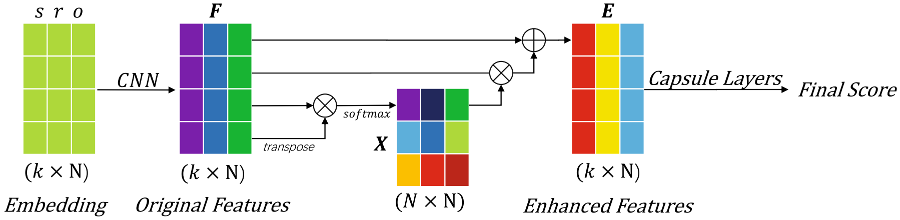


Fig. 1. An overview of the GCE with $k = 4, \mathsf{N} = 3$.

3.3 Global Contextual Information Module

Each channel map of high level features can be regarded as a relation-specific attributes [22], and different semantic attributes are associated with each other [11]. By exploiting the inter-dependencies between relation-specific attributes, we could emphasize some interdependent feature and enhance the feature representations of specific semantics. Therefore, we introduce a global contextual information module with attention mechanism to draw global contextual information and enhance feature representations.

As illustrated in Fig. 2, we directly calculate the attention map $X \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$ from the original feature $F \in \mathbb{R}^{k\times\mathsf{N}}$. Specially, we perform a matrix multiplication between the transpose of F and original F. Then we apply a softmax layer

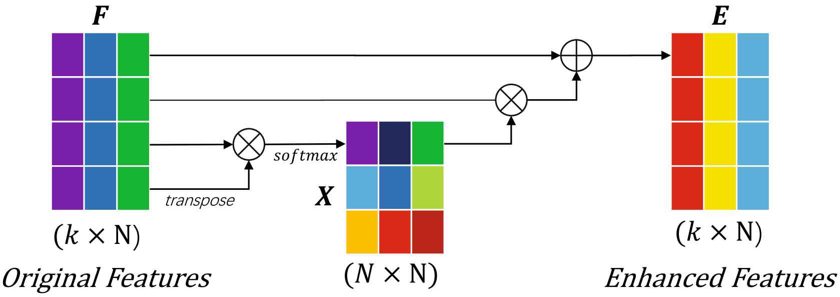


Fig. 2. The global contextual information module.

to obtain the attention map $X \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$:

$$x_{ji} = \frac{\exp\left(F_i \cdot F_j\right)}{\sum_{i=1}^{\mathsf{N}} \exp\left(F_i \cdot F_j\right)} \tag{4}$$

where x_{ji} measures the i^{th} filter's impact on the j^{th} filter. Then we perform a matrix multiplication between X and the transpose of F and reshape their result to $\mathbb{R}^{k\times\mathsf{N}}$. Then we multiply the result by a scale parameter α and perform an element-wise sum operation with original feature F to obtain the final output $E \in \mathbb{R}^{k\times\mathsf{N}}$:

$$E_j = \alpha \sum_{i=1}^{\mathsf{N}} \left(x_{ji} F_i\right) + F_j \tag{5}$$

where α gradually learns a weight from 0. The Eq. 5 shows that the final enhanced features E of each channel is a weighted sum of the features of all channels and original features F, which models the semantic dependencies between feature maps.

It is notable that our global contextual information module is simple and can be directly inserted in the existing CapsE pipeline. We do not increase too much computational consumption while effectively enhancing feature representations.

4 Experiments and Results

4.1 Benchmark Datasets

To evaluate the performance of our proposed GCE, we carry out comprehensive experiments on two benchmark datasets including WN18RR [9] and F15k-237 [29]. And Table 1 lists the statistics of WN18RR and FB15k-237.

WN18RR [9] is created from WN18 [5], which is a subset of WordNet [19]. It is first pointed in [29] that 97% of the training triples in WN18 have inverse relations linked to test set. Thus WN18RR [9] is introduced to remove the inverse relations. In summary, WN18RR contains 93,003 triples with 40,943 entities and 11 different relations.

FB15k-237 [29] is a subset of FB15K [5], originally derived from Freebase [4]. It is first pointed in [29] that FB15K suffers from test leakage through inverse relations. And then FB15k-237 is introduced, in which inverse relations have been removed. FB15k-237 contains 310,116 triples with 14,541 entities and 237 different relations.

Table 1. Statistics of the experimental datasets.

Dataset	Entities	Relations	Triples			
			Training	Validation	Test	Total
WN18RR	40,943	11	86,835	3034	3,134	93,003
FB15k-237	14,541	237	272,115	17,535	20,466	310,116

4.2 Evaluation Protocol

Following [5], for each valid test triple (s, r, o), we replace either s or o by other entities to create a set of corrupted triples. We use the "Filtered" setting protocol produced by [5], i.e., and filter out all corrupted triples before ranking. We rank the valid test triples and corrupted triples in descending order of their scores. To evaluate the performance of the models, we employ five common evaluation metrics in link prediction task, mean rank (MR), mean reciprocal rank (MRR), Hits@10, Hits@3 and Hits@1 (the proportion of the valid test triples ranking in top 10, 3 and 1 predictions). Lower MR, higher MRR, higher Hits@N indicates better performance of models. We report average results across 5 runs. We find that the variance is substantially low on all the metrics.

4.3 Training Protocol

During the training process, we use the common Bernoulli strategy [16,21,22,36] to generate invalid triples. Following ConvKB [21] and CapsE [22], we use the pre-trained entity and relation embeddings produced by TransE [5] to initialize the entity and relation embeddings in our GCE for WN18RR and FB15K237. We employ the TransE training implementations provided by [21,22].

In our GCE, we set the batch size as 128, the number of neurons with the capsule in the second capsule layer as 10, the number of iterations in the routing algorithm as 1, ReLU as the activation function, Adam [13] as the optimizer. We run GCE up to 100 epochs and monitor the MRR and Hits@10 score after each 10 training epochs to choose optimal hyper-parameters. The convolution filters are initialized by a truncated normal distribution or by $[0.1, 0.1, -0.1]$.

For WN18RR, the highest MRR and Hits@10 score on the validation set are obtained when using $d = 100, lr = 1e^{-}5, N = 40$, the L_2-regularization $\lambda = 0.01$, and the truncated normal distribution for convolution filter initialization.

For FB15K237, the highest MRR and Hits@10 scores on the validation set are obtained when using $d = 100, lr = 1e^{-}4, N = 50$, the L_2-regularization $\lambda = 0.001$, and $[0.1, 0.1, -0.1]$ for convolution filter initialization.

4.4 Ablation Experiments

Global Contextual Information Module. We employ the global contextual information module on the top of the convolution layer to capture global contextual information for better link prediction. To verify the performance of the global contextual information module, we conduct experiments with different settings in Table 2.

As shown in Table 2, the global contextual information module remarkably improves the link prediction performance on both FB15k237 and WN18RR. Compared with the baseline CapsE, our GCE performs better on both experimental datasets. On FB15k-237, GCE gains 0.043 improvement in MRR, and 3% absolute improvement in Hits@10. On WN18RR, GCE obtains 0.016 improvement in MRR, and 1.2% absolute improvement in Hits@10. These improvements conduct the effectiveness and robustness of the global contextual information module.

Table 2. Ablation study on FB15k237 and WN18RR val sets. GCIM represents the global contextual information module.

Method	GCIM	FB15K237			WN18RR		
		MR	MRR	Hits@10	MR	MRR	Hits@10
CapsE		303	0.523	0.593	719	0.415	0.560
GCE	✓	324	0.566	0.623	1648	0.429	0.572

4.5 Comparing with State-of-the-art Methods

To evaluate the effectiveness of our proposed GCE, we perform a comprehensive comparison with the existing state-of-the-art methods. These TransE [5], DistMult [38], ComplEx [30], R-GCN [25], ConvE [9], ConvKB [21], CapsE [22], RotatE [28] and InteractE [31]. Table 3 shows the experimental results of our GCE with previous state-of-the-art models on two standard link prediction datasets under the same evaluation protocols. The results of all the baselines are taken directly from the values reported in the papers [9,21,22,28,31]. Since our GCE is built on CapsE, we specially compare against it. We find that our GCE outperforms CapsE on four out of five metrics on both FB15k237 and WN18RR. Specially, on FB15k-237, GCE gains 0.043 improvement in MRR, 3% absolute improvement in Hits@10, 4.6% absolute improvement in Hits@3, and 4.3% absolute improvement in Hits@1. On WN18RR, GCE obtains 0.016 improvement in

MRR, 1.2% absolute improvement in Hits@10, 1.6% absolute improvement in Hits@3 and 1.7% absolute improvement in Hits@1. Compared with other baseline methods, our GCE obtains the highest MRR, highest Hits@10, highest Hits@3 and highest Hits@1 on FB15k-237, highest Hits@10 and second highest Hits@3 on WN18RR.

Table 3. Experimental results on WN18RR and FB15K-237 test sets. The best score is in **bold** and second best score is underlined.

	FB15K-237					WN18RR				
			Hits@N					Hits@N		
Model	MR	MRR	@10	@3	@1	MR	MRR	@10	@3	@1
TransE [5]	323	0.279	0.441	0.376	0.198	2300	0.243	0.532	0.441	0.042
DistMult [38]	254	0.241	0.419	0.263	0.155	5110	0.444	0.504	0.470	0.412
ComplEx [30]	339	0.247	0.428	0.275	0.158	5261	0.449	0.531	0.469	0.410
R-GCN [25]	216	0.248	0.417	0.258	0.153	6700	0.123	0.207	0.138	0.083
ConvE [9]	244	0.325	0.501	0.356	0.237	4187	0.433	0.521	0.442	0.410
ConvKB [21]	245	0.407	0.529	0.426	0.348	763	0.253	0.567	0.445	0.051
SACN [27]	–	0.352	0.541	0.393	0.259	–	0.473	0.541	0.480	**0.431**
CapsE [22]	303	0.523	0.593	0.528	0.493	**719**	0.415	0.560	0.466	0.351
RotatE [28]	**177**	0.338	0.533	0.375	0.241	3340	**0.476**	0.571	**0.492**	0.428
InteractE [31]	172	0.354	0.535	–	0.263	5202	0.463	0.528	–	0.430
Our GCE	324	**0.566**	**0.623**	**0.569**	**0.536**	1648	0.429	**0.572**	0.482	0.368

4.6 Evaluation on Different Relation Types

Following [5,22,36], we categorize the relations r in FB15k-237 into four categories: one-to-one (1-1), one-to-many (1-M), many-to-one (M-1) and many-to-many (M-M). we calculate the averaged number η_s of head entities per tail entity and the averaged number η_o of tail entities per head entity. A given relation is 1-1 if η_s <1.5 and η_o <1.5, 1-M if η_s <1.5 and η_o ≥1.5, M-1 if η_s ≥1.5 and η_o <1.5, M-M if η_s ≥1.5 and η_o ≥1.5. Then there are 17, 26, 81 and 113 relations labelled 1-1, 1-M, M-1 and M-M, respectively. And we obtain that FB15k-237 test set has 0.9% of 1-1, 6.3% of 1-M, 20.5% of M-1 and 72.3% of M-M.

Figure 3 shows the Hits@10 and MRR results for head and tail prediction w.r.t each relation category on FB15k-237 test set. It is notable that GCE woks better than CapsE in entities prediction on the "side M" of triples (e.g., head prediction in 1-M, M-1 and M-M; tail prediction in 1-M and M-M). On M-M triples, our GCE gains 3.86% and 3.29% absolute improvements in Hits@10 for head and tail prediction, 0.047 and 0.048 improvements in MRR for head and tail prediction. And these improvements means better predictions on M-M triples are possible with our GCE. Considering that 72.3% of FB15k-237 test triples are M-M triples, the performance of GCE on FB15k-237 maybe benefit from the better prediction on M-M triples.

Figure 4 shows the Hits@10 and MRR results w.r.t each relation on WN18RR dataset. *also_see*, *similar_to*, *verb_group* and *derivationally_related_form* are symmetric relations and could be considered as M-M relations [22]. From Fig. 4, our GCE also performs a little better than CapsE on triples with these M-M relations. Thus, results in Fig. 3 and Fig. 4 are consistent and demonstrate that our GCE has better performance on the prediction of triples with M-M relations.

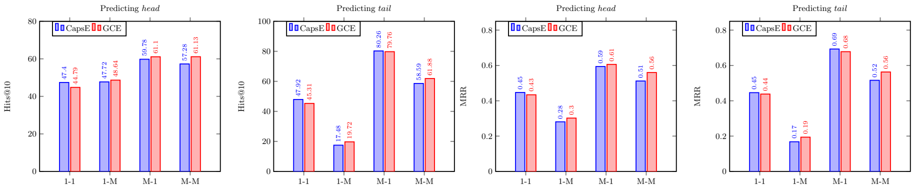


Fig. 3. Hits@10 and MRR on the FB15k-237 test set w.r.t each relation category.

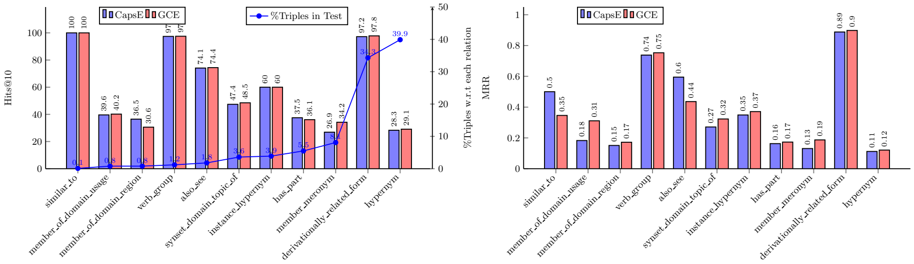


Fig. 4. Hits@10 and MRR on the WN18RR test set w.r.t each relation. The right y-axis is the percentage of triples corresponding to relations.

5 Conclusion

In this paper, we propose GCE—a simple and effective embedding model to explore the capability of global contextual information in the task of knowledge graph completion. We carefully design a global contextual information module with the attention mechanism. This module could adaptively aggregate global contextual information and enhance the feature representations. It could be directly inserted in the existing pipelines while increasing a little computational consumption. Experimental results show that our GCE achieves competitive results on two benchmark datasets FB15K237 and WN18RR for knowledge base completion. The ablation experiments show that our GCE has better link prediction performance on the triples with complex relations. This implies that our

GCE would be a potential candidate for applications which contain many M-M relations such as search personalization.

Acknowledgements. This research was partially supported by the National Key Research and Development Program of China (2017YFB1402400 and 2017YFB1402401), the Key Research Program of Chongqing Science and Technology Bureau (cstc2020jscx-msxmX0149), the Key Research Program of Chongqing Science and Technology Bureau (cstc2019jscx-mbdxX0012), and the Key Research Program of Chongqing Science and Technology Bureau (cstc2019jscx-fxyd0142).

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
3. Bast, H., et al.: Semantic search on text and knowledge bases. Found. Trends Inf. Retrieval® **10**(2–3), 119–271 (2016)
4. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250. ACM (2008)
5. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, vol. 26, pp. 2787–2795 (2013)
6. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Twenty-Fourth AAAI Conference on Artificial Intelligence (2010)
7. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018, pp. 833–851. Springer, Cham (2018)
8. Cho, K., Courville, A., Bengio, Y.: Describing multimedia content using attention-based encoder-decoder networks. IEEE Trans. Multimedia **17**(11), 1875–1886 (2015)
9. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
10. Dietz, L., et al.: Utilizing knowledge graphs for text-centric information retrieval. In: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1387–1390. ACM (2018)
11. Fu, J., et al.: Dual attention network for scene segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3146–3154 (2019)
12. Ji, G., He, S., Xu, L., Liu, K., Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 687–696 (2015)

13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
14. Li, Q., Cao, Z., Zhong, J., Li, Q.: Graph representation learning with encoding edges. Neurocomputing **361**, 29–39 (2019)
15. Li, Q., Dong, J., Zhong, J., Li, Q., Wang, C.: A neural model for type classification of entities for text. Knowl.-Based Syst. **176**, 122–132 (2019)
16. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Twenty-Ninth AAAI Conference on Artificial Intelligence (2015)
17. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1412–1421 (2015)
18. Mahdisoltani, F., Biega, J., Suchanek, F.: YAGO3: a knowledge base from multilingual Wikipedias. In: 7th Biennial Conference on Innovative Data Systems Research. CIDR Conference (2014)
19. Miller, G.A.: WordNet: a lexical database for English. Commun. ACM **38**(11), 39–41 (1995)
20. Nathani, D., Chauhan, J., Sharma, C., Kaul, M.: Learning attention-based embeddings for relation prediction in knowledge graphs. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics (2019)
21. Nguyen, D.Q., Nguyen, T.D., Nguyen, D.Q., Phung, D.: A novel embedding model for knowledge base completion based on convolutional neural network. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pp. 327–333 (2018)
22. Nguyen, D.Q., Vu, T., Nguyen, T.D., Nguyen, D.Q., Phung, D.: A capsule network-based embedding model for knowledge graph completion and search personalization. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 2180–2189 (2019)
23. Nickel, M., LMU, I., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: ICML, vol. 11, pp. 809–816 (2011)
24. Nickel, M., et al.: Holographic embeddings of knowledge graphs. In: Thirtieth AAAI Conference on Artificial Intelligence, pp. 1955–1961 (2016)
25. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
26. Shahzad, M., et al.: Virtual data integration of heterogeneous genomic biological knowledge base. Bahria Univ. J. Inf. Commun. Technol. (BUJICT) 8(2) (2015)
27. Shang, C., Tang, Y., Huang, J., Bi, J., He, X., Zhou, B.: End-to-end structure-aware convolutional networks for knowledge base completion. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 3060–3067 (2019)
28. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. In: International Conference on Learning Representations (2019)
29. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, pp. 57–66 (2015)

30. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning, pp. 2071–2080 (2016)
31. Vashishth, S., Sanyal, S., Nitin, V., Agrawal, N., Talukdar, P.: Interacte: improving convolution-based knowledge graph embeddings by increasing feature interactions. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence, pp. 3009–3016. AAAI Press (2020). https://aaai.org/ojs/index.php/AAAI/article/view/5694
32. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
33. Wang, H., Zhao, M., Xie, X., Li, W., Guo, M.: Knowledge graph convolutional networks for recommender systems. In: The World Wide Web Conference, pp. 3307–3313. ACM (2019)
34. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: a survey of approaches and applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017)
35. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7794–7803 (2018)
36. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Twenty-Eighth AAAI Conference on Artificial Intelligence (2014)
37. West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., Lin, D.: Knowledge base completion via search-based question answering. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 515–526. ACM (2014)
38. Yang, B., Yih, W.T., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)
39. Yih, W.t., Richardson, M., Meek, C., Chang, M.W., Suh, J.: The value of semantic parse labeling for knowledge base question answering. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 201–206 (2016)
40. Yuan, Y., Wang, J.: OCNet: Object context network for scene parsing. arXiv preprint arXiv:1809.00916 (2018)
41. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 353–362. ACM (2016)
42. Zhang, H., et al.: Context encoding for semantic segmentation. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7151–7160 (2018)
43. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6230–6239 (2017)
44. Zhong, J., Wang, C., Li, Q., Li, Q.: A new graph-partitioning algorithm for large-scale knowledge graph. In: Gan, G., Li, B., Li, X., Wang, S. (eds.) ADMA 2018. LNCS (LNAI), vol. 11323, pp. 434–444. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-05090-0_37
45. Zhu, G., Iglesias, C.A.: Sematch: semantic similarity framework for knowledge graphs. Knowl.-Based Syst. **130**, 30–32 (2017)

Consistency and Coherency Enhanced Story Generation

Wei Wang[1,2], Piji Li[3](✉), and Hai-Tao Zheng[1,2](✉)

[1] Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
w-w16@mails.tsinghua.edu.cn, zheng.haitao@sz.tsinghua.edu.cn
[2] Department of Computer Science and Technology,
Tsinghua University, Beijing, China
[3] Tencent AI Lab, Shenzhen, China
pijili@tencent.com

Abstract. Story generation is a challenging task, which demands to maintain consistency of the plots and characters throughout the story. Previous works have shown that GPT2, a large-scale language model, has achieved advanced performance on story generation. However, we observe that several serious issues still exist in the stories generated by GPT2, which can be categorized into two folds: consistency and coherency. In terms of consistency, on the one hand, GPT2 cannot guarantee the consistency of the plots explicitly. On the other hand, the generated stories usually contain coreference errors. In terms of coherency, GPT2 does not take account of the discourse relations between sentences of stories directly. To enhance the consistency and coherency of the generated stories, we propose a two-stage generation framework, where the first stage is to organize the story outline which depicts the story plots and events, and the second stage is to expand the outline into a complete story. Therefore, the consistency of the plots can be controlled and guaranteed explicitly. In addition, coreference supervision signals are incorporated to reduce coreference errors and improve coreference consistency. Moreover, we design an auxiliary task of discourse relation modeling to improve the coherency of the generated stories. Experimental results on a story dataset show that our model outperforms baseline approaches in terms of both automatic metrics and human evaluation.

1 Introduction

A story is anything that is narrated in the form of a causally/logically linked set of event plots involving some shared characters [21]. Story generation aims at automatically generating stories with such attributes. Different from other text generation tasks, story generation is more challenging because it is restricted to several constraints: (1) it must maintain **consistent plots** to form a reasonable story; (2) it must guarantee the **consistency** of the characters throughout the story; (3) the **coherency** of the text units such as the clauses or sentences

W. Wang—Work was done during internship at Tencent AI Lab.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 694–709, 2021.
https://doi.org/10.1007/978-3-030-72113-8_46

Table 1. A story generated by GPT2.

Prompt: The zombie apocalypse is upon us but all humans are immune to the virus . All other mammals are susceptible
Story: As I sat there I tried to remember how many days it had been since the last one died . [My wife and daughter were dead , I knew they would n't be alive any time soon .]$_{1a}$ The food ran out quickly , they had to be taken away to stay in the house , but that was no reason to be angry if there were a few people left living there . After a month of food being gone a little girl sat next to me . She was the only other survivor who was n't bitten , as it was hard for us to eat them , we were all sickly . That would be another month to go and if the other animals were still alive in the food, they would n't be able to eat her . " Hey little one . " I looked up and saw her . [I had a big smile on my face at this point .]$_{2a}$ [My wife was the only one who noticed , but her smile was all that was visible of me , and not of her .]$_{1b}$ [I looked at her , and smiled sadly .]$_{2b}$ She looked sad , but did n't care . She never cared for me .

should be concerned. In order to improve the quality of plot consistency, some previous works focus on the perspective of plot planning and then merge the text units according to the order of plots [15,16,24,25,28]. We observe that those approaches rely heavily on human annotations and are restricted to the abstract story representation level without surface realization in natural language, such as producing event verb sequence and sentence segments. Therefore, these methods need to work with sentence templates or rules to generate stories.

In the past few years, several end-to-end approaches based on Sequence-to-Sequence (Seq2Seq) models [1,31] are proposed, which can generate a story at a stroke in a left-to-right manner [7,9,12]. These methods are data-driven and can directly generate stories in natural language form instead of other abstract representations. However, these methods struggle to capture the high-level interactions between the plot points and maintain consistent plots throughout the story. Thus, several two-stage models for story generation have recently been proposed [5,10,20,33,35]. These models usually decompose story generation into two stages: generating the middle form first and then generating the final story.

Recently, the OpenAI GPT2/3 language model [4,27] achieves strong performance on several language generation tasks such as dialogue systems [18,19,32]. [30] and [11] verify the performance of GPT2 on story generation and GPT2 outperforms both end-to-end methods and two-stage methods. However, after analyzing the generated stories carefully, we observe that there are still some serious issues in the generated stories by GPT2. Take a story generated by GPT2 as shown in Table 1 for example. The story is about survivors at the end of the world. First, plot consistency cannot be guaranteed among multiple sentences of the story, such as blue sentences in Table 1. The sentence $1a$ describes "My wife and daughter were dead". But the sentence $1b$ talks about "My wife" again. It is contradictory. There is the same problem in the sentence $2a$ and $2b$. Second, there are still coreference errors in generated stories, such as red text in Table 1. It is not clear who *they* and *them* refer to. Moreover, Top-k sampling [4,27,30] is usually utilized as the decoding strategy in long text generation. The random operation in sampling will disturb the generation procedure by producing improper tokens which will decrease the quality. This phenomenon is more pronounced at the border of sentences, therefore we can sometimes observe the bad performance in discourse coherency.

To solve the aforementioned problems, we propose a two-stage generation model based on Transformer-based auto-regressive language models to improve the consistency and coherency of stories. Specifically, the first stage is to organize the story outline which depicts the story plots and events, and the second stage is to expand the outline into a complete story. Therefore, the consistency of the plots can be controlled and guaranteed explicitly. In addition, coreference supervision signals are incorporated to reduce coreference errors and improve coreference consistency. Moreover, we design an auxiliary task of discourse relation modeling to enhance the discourse coherency of the generated stories. Both the backbone models in the two stages are designed based on Transformer-based language models. Thus, on the one hand, the framework can still inherit the superior performance of GPT2, on the other hand, it can guarantee the plot consistency, coreference consistency, as well as discourse coherency. The main contributions of this paper are summarized as follows:

- A two-stage framework based on Transformer-based language models is designed to control the plots and improve the consistency of generated stories.
- A coreference constraint is applied to improve the coreference consistency of generated stories.
- We design a discourse relation modeling component as an auxiliary task during training to enhance the performance of discourse coherency.
- Experiments on a story dataset from Reddit demonstrate that our model outperforms baseline methods in terms of both automatic metrics and human evaluation.

2 Methodology

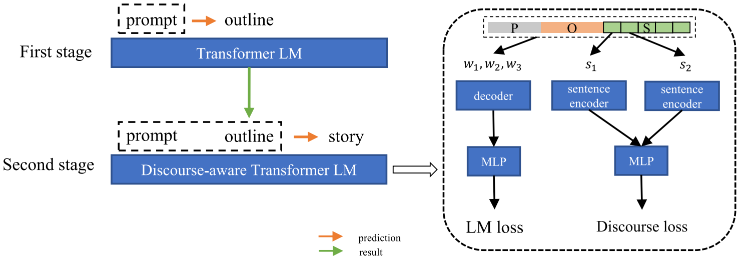


Fig. 1. The framework of our model for story generation.

To begin with, we state the problem of story generation as follows: given a prompt context $\mathbf{X} = \{x_1, ..., x_i ..., x_k\}$ where x_i denotes each word in the prompt, the model needs to generate a story $\mathbf{Y} = \{y_1, ..., y_i, ..., y_n\}$ following the prompt $\mathbf{X}$ by maximizing the conditional probability $p(\mathbf{Y}|\mathbf{X})$.

As shown in Fig. 1, to enhance the consistency and coherency of generated stories, we propose a two-stage framework for story generation. The first stage is story outline generation which can generate the plot outline based on the given prompt. Then in the second stage, the whole story is completed by embellishing the outline generated in the first stage. Transformer-based language models are introduced as backbone models for those two stages respectively.

2.1 Transformer-Based Language Model

Inspired by the popular pre-trained language models for text generation such as GPT2 [27], XLNET [34] and GPT3 [4], we also employ the Transformer-based auto-regressive language models as our backbone frameworks.

Transformer-based language models only contain a decoder. The decoder consists of N identical self-attention blocks and each block contains two sub-layers: a self multi-head attention layer and a feed-forward layer. A add & norm layer is employed around each of two sub-layers. Formally, given the input $\mathbf{H}^{n-1}$, the output $\mathbf{H}^n$ of each decoder block is computed as follows:

$$\mathbf{C}^n = \mathrm{LN}\left(\text{SELF-ATT}\left(\mathbf{H}^{n-1}\right) + \mathbf{H}^{n-1}\right) \tag{1}$$

$$\mathbf{H}^n = \mathrm{LN}\left(\mathrm{FFN}\left(\mathbf{C}^n\right) + \mathbf{C}^n\right) \tag{2}$$

where SELF-ATT$(\cdot)$, LN$(\cdot)$, and FFN$(\cdot)$ are respectively self-attention mechanism, layer normalization, and feed-forward network with ReLU activation in between. SELF-ATT$(\cdot)$ computes attention over the input $\mathbf{H}^{n-1}$ as follows:

$$\text{SELF-ATT}\left(\mathbf{H}^{n-1}\right) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{3}$$

where $\{\mathbf{Q}, \mathbf{K}, \mathbf{V}\}$ are query, key and value vectors that are transformed from the input $\mathbf{H}^{n-1}$. $\sqrt{d_k}$ is the scaling factor where d_k is the dimension size of the query and key vectors. Given the word embeddings $\mathbf{E} = \{e_1, e_2, ..., e_m\}$ and corresponding positional embeddings $\mathbf{P} = \{p_1, p_2, ..., p_m\}$ of a sentence with m words, the first block input $\mathbf{H}^0 = \mathbf{E} + \mathbf{P}$.

Finally, a linear function with softmax activation is used to compute the probability of next word x_t via:

$$p\left(x_t|x_{\leq t-1}\right) = \mathrm{softmax}\left(g\left(h_t\right)\right) \tag{4}$$

We calculate negative log-likelihood loss for model training:

$$\mathcal{L}_{\mathrm{lm}} = -\frac{1}{m}\sum_t \log p\left(x_t|x_{\leq t-1}\right) \tag{5}$$

2.2 Two-Stage Generation

Outline Preparation

In order to regard the outline generation task as a supervised learning problem, we must construct a high-quality training dataset including sufficient prompt-outline pairs. In this work, we investigate two forms of the outline: keyword and abstract. These two forms retain the important information of the story and ignore some details and commonly used in two-stage based methods [5,10,35]. Our motivation is to use two-stage generation to improve performance of one-stage Transformer-based language models, so we do not design a new middle form. Specifically, we use the RAKE algorithm [29][1] to extract keywords of stories. According to [35] and the average length of stories in our corpus, we extract 10 keywords for each story. We use a variation of the TextRank algorithm [3][2] to extract abstracts of stories. In order to retain important information and ignore some detail information, we extract 30% sentences of each story as abstract in TextRank. Thus, we can get (prompt, outline, story) triples automatically to train the two-stage model.

Prompt to Outline Generation

A Transformer-based language model based decoder is used to generate outlines. Specifically, we concatenate prompt $\mathbf{X}$ and outline $\mathbf{Z}$ with `<SEP>` token to get a sequence $\mathbf{X}'$. For training, we compute cross entropy of all tokens in $\mathbf{X}'$ as normal language model. When testing, given the prompt tokens as context, the decoder generates outline tokens.

Prompt and Outline to Story Generation

Another decoder with the same architecture is used to generate stories. We concatenate prompt $\mathbf{X}$, outline $\mathbf{Z}$ and story $\mathbf{Y}$ with `<S>` and `<SEP>` token to get a

Table 2. Example pairs from Books 8 dataset.

S1	Marker	S2
Her eyes flew up to his face.	and	Suddenly she realized why he looked so different.
The concept is simple.	but	The execution will be incredibly dangerous.
You used to feel pride.	because	You defended innocent people.
Belter was still hard at work.	when	Drade and barney strolled in.
I’ ll tell you about it.	if	You give me your number.
We plugged bulky headsets into the dashboard.	so	We could hear each other when we spoke into the microphones.
It was mere minutes or hours.	before	He finally fell into unconsciousness.
And then the cloudy darkness lifted.	though	The lifeboat did not slow down.

[1] https://pypi.org/project/rake-nltk/.

[2] https://radimrehurek.com/gensim/.

sequence $\mathbf{X}''$. For training, we compute cross entropy of prompt and story tokens in $\mathbf{X}''$. Note that we don't calculate the loss of the outline tokens. Because the outline tokens come from the story and we avoid computing loss of these tokens twice. When testing, given the prompt and the outline tokens as context, the decoder generates story tokens. Next, two components are incorporated in this stage to enhance discourse coherency and coreference consistency.

2.3 Discourse Coherency Enhancement

In order to improve discourse representation of Transformer-based language model, we design a discourse relation classification task as an auxiliary task. Discourse relations describe how two segments (e.g., clauses, sentences, and larger multi-clause groupings) of discourse are logically connected. These relations can be used to describe the high-level organization of the text. Thus, discourse relation is an important aspect of story coherence. In this work, we only consider shallow discourse relations between adjacent sentences as many research on discourse relation classification do [2,6,14].

Discourse Information Preparation
In order to get discourse label of adjacent sentences in stories, we need to train a golden discourse relation classification model. However, there is limited annotation corpus of implicit discourse relations and explicit discourse relations. For example, the commonly used dataset Penn Discourse Treebank 2.0 [26] contains about 10k pairs. Following [22], we use discourse markers as replacement of discourse relations. Because we are able to automatically curate a sizable training set of sentence pairs with discourse markers. We use the discourse marker dataset Book 8 from [22], which contains 3.6M sentence pairs and each pair is labeled with one connective of 8 connectives as discourse label. Several sentence pairs and corresponding discourse markers are shown in Table 2.

We fine tune BERT [8][3] in this dataset to get a golden discourse marker prediction model. Then we use this model to tag discourse relation label of sentence pairs in our story corpus. Considering that this automatic tagging may produce large errors, we only keep labels with high classification probability (the threshold is 0.8), and labels with lower probability are replaced with the ninth label *unknown* and are discarded. The sentence pairs with labels belonging to 8 connectives are used to train our discourse relation classification component.

Discourse-aware Story Generation
The discourse relation classification component contains a sentence encoder and a two-layer MLP. The encoder is used to extract semantic feature of sentences and the MLP is used to convert feature into classification probability. The sentence encoder shares parameters with the story decoder excluding the output layer. For a story $\mathbf{Y}$ contains several sentence $\{\mathbf{S_1}, \mathbf{S_i}, \mathbf{S_p}\}$ and each sentence contains several words $\mathbf{S_i} = \{y_{i1}, y_{ij}, y_{iq}\}$, we get output h_{ij}^{w} of encoder as word

[3] https://github.com/huggingface/transformers.

representation and use max pooling operation on words of this sentence to get sentence representation h_i^s:

$$\mathbf{H}_i^s = \text{encoder}(\mathbf{S_i}) \tag{6}$$

$$h_i^s = \max(\mathbf{H}_i^s) \tag{7}$$

Then the MLP is used to classify adjacent sentences as follows:

$$f = \tanh(\mathbf{W_f}[h_i^s, h_{i+1}^s] + b_f) \tag{8}$$

$$p(dis|\mathbf{S_i}, \mathbf{S_j}) = \quad \text{softmax}(\mathbf{W_o} f + b_o) \tag{9}$$

The loss function $\mathcal{L}_{\text{dis}}$ of this component is the cross-entropy of discourse label. Then a joint loss function is applied to train the second stage model:

$$\mathcal{L} = \mathcal{L}_{\text{lm}} + \lambda_1 \mathcal{L}_{\text{dis}} \tag{10}$$

where λ_1 is a hyperparameter to balance two tasks.

2.4 Coreference Consistency Enhancement

Although Transformer-based language model has the ability of long-distance dependence, there are still some coreference errors in the generated stories. In order to encourage the model to attend correct entities, we add a supervision on attention weight of entity mention tokens. We use Stanford's CoreNLP tool[4] to extract coreference annotation of stories.

Specifically, for a story $\mathbf{Y}$ we get p coreference clusters and each cluster contains q entity mentions. We assign each token y_i^c in entity mention subsequence $\mathbf{Y}^c = \{y_1^c, y_i^c, y_{pq}^c\}$ a cluster label c_i to get cluster label sequence $\mathbf{C} = \{c_1, c_i, c_{pq}\}$. During training, for an entity mention token y_i^c, we get attention weights between current token and previous tokens $\{y^c \leq i - 1\}$ in last self-attention layer of the decoder, the sum of which is 1:

$$\sum_{k=1}^{i-1} \alpha_{ik} = 1 \tag{11}$$

We design a coreference loss to maximize attention weights of tokens in the same cluster as follows:

$$\mathcal{L}_{\text{coref}} = -\frac{1}{pq} \sum_{i=1}^{pq} \sum_{k=1}^{i-1} \mathbb{1}(c_k = c_i) \log \alpha_{ik} \tag{12}$$

Considering these two components, the loss function for the second stage model is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{lm}} + \lambda_1 \mathcal{L}_{\text{dis}} + \lambda_2 \mathcal{L}_{\text{coref}} \tag{13}$$

[4] https://stanfordnlp.github.io/CoreNLP/.

3 Experimental Setup

3.1 Settings and Data Set

For two Transformer decoders, we apply the same model size as GPT2-117M [27]. Thus we can analyze the effect of pre-trained weights of GPT2. Specifically, the dimension of word embedding and the dimension of hidden vectors are set to 768. The number of self-attention block is set to 12 and 12 heads are used in self multi-head attention. We train the model using Adam [13] with learning rate 0.0005. The dropout rate is set to 0.3 for regularization. λ_1 and λ_2 are set to 0.1 and 0.3 according to the performance in valid set. Following [9] we generate stories with random top k sampling, where next words are sampling from the top $k = 20$ candidates rather than the entire vocabulary distribution.

We use writing prompts dataset from [9], which is collected from Reddit's WRITINGPROMPTS forum[5]. WRITINGPROMPTS is a community where online users inspire each other to write by submitting story prompts. Each prompt can have multiple story responses. The prompts have a large diversity of topic, length, and detail. There are 300k stories and the dataset is split into TRAIN, VAL and TEST (90%/5%/5%). For our experiments, we limit the length of the stories to 500 words maximum. We use GPT2's BPE vocabulary with size of 50,527 in our model.

3.2 Evaluation Metrics

Automatic Evaluation. Many commonly used metrics based on n-gram overlap between the generated text and the human text, such as BLEU [23], are not useful in story generation, which is also observed by previous works [9,20]. Because we do not aim to generate a specific story; we want to generate viable and novel stories.

In order to evaluate different aspects of stories, we use four types of metrics. We use **Perplexity** to evaluate the fluency of stories. Perplexity is commonly used to evaluate the quality of language models, and it reflects how fluently the model can produce the correct next word given the preceding words. What's more, in order to evaluate the diversity of stories we compute **Distinct-1/2** [17], which is the percentage of distinct n-grams in all generated stories and is widely used in conversation generation.

In order to evaluate the discourse coherency of the stories, we reuse the fine-tuned BERT for evaluation. Specifically, we use BERT to tag discourse label for sentence pairs in generated stories in the same way as the tagging process of training set in Sect. 2.3. We compute the percentage of sentence pairs with **Unknown** labels in generated stories. The fewer sentence pairs with unknown labels the model generates, the better the coherency of stories are. In order to evaluate the coreference coherence, we compute the averaged number of **Coreference Chains** in each story. Specifically, we use Stanford's CoreNLP tool[6] to extract coreference chains of generated stories.

[5] https://www.reddit.com/r/WritingPrompts/.

[6] https://stanfordnlp.github.io/CoreNLP/.

Human Evaluation. To further evaluate the quality of generated stories, we conduct pair-wise comparisons with two strong baseline models (FConvS2S and GPT2P). We evaluate the models from the following three perspectives: **Relevance** to indicate whether a story is relevant to the given prompt, **Grammaticality** to indicate whether a story is natural and fluent, and **Logicality** to indicate whether a story is consistent and coherent in terms of causal dependencies in the context. Note that the three aspects are independently evaluated. We randomly sample 100 prompts from the test set and obtain 300 stories from three models. For each pair of stories (one by our model and the other by a baseline, along with the prompt), three graduate students as annotators are asked to give a preference (win, lose, or tie) in terms of three metrics respectively. We adopt majority voting to make final decisions among the three annotators.

3.3 Comparison Methods

Conv Seq2Seq with self-attention (ConvS2S). We replicate the model proposed by [9] using their source code, which applies a convolutional sequence-to-sequence model with gated self-attention to generate stories from prompts.
Fusion of Conv Seq2Seq with self-attention (FConvS2S). The model is also proposed by [9], which utilizes a fusion mechanism to integrate two **ConvS2S**.
GPT2. The model only contains a Transformer-based decoder and has the same model size as GPT2-117M [27]. We train the model from scratch.
GPT2 with Pre-trained (GPT2P). We first load pre-trained weights of GPT2-117M and then fine tune the model on the used dataset.
Ours. Our overall model contains two-stage generation, discourse relation classification and coreference supervision. In order to evaluate the upper bound of two-stage generation, we use different percentages of tokens of ground truth outlines as contexts to generate stories. Ours(0%) means using own generated outlines as contexts in the second stage to generate stories. It is our final model. Ours(100%) means all tokens of ground truth outlines are used as contexts. It is the upper bound model.

4 Results and Discussions

Automatic Evaluation and Human Evaluation
As shown in Table 3, we compute four types of metrics for these methods. We can see that GPT2 outperforms FConvS2S and ConvS2S in all metrics. This indicates that the self-attention based model is superior to the convolutional based model in story generation. Although FConvS2S and ConvS2S are enhanced with a self-attention mechanism, their ability to capture long-distance dependence is still weaker than GPT2. Compared to GPT2, GPT2P improves the perplexity and distinct significantly. GPT2P also generates the least sentence pairs with *unknown* discourse relation. This shows that pre-trained weights contribute to generating more fluent, diverse and coherent stories. Compared to these methods,

Table 3. Automatic evaluation results on TEST set.

Method	Perplexity↓	Dis-1(%)↑	Dis-2(%)↑	Unknown(%)↓	Coref Chains↑
ConvS2S	34.61	0.400	5.191	76.01	5.52
FConvS2S	33.97	0.482	6.271	75.60	5.43
GPT2	29.50	0.474	6.796	74.95	5.67
GPT2P	25.64	0.493	7.333	73.61	5.61
Ours (0% ground truth outline)	30.84	0.531	7.379	75.19	5.98
Ours (50% ground truth outline)	19.21	1.311	13.253	75.15	5.97
Ours (100% ground truth outline)	10.32	1.509	15.266	74.97	5.80

Table 4. Human evaluation results on TEST set.

Method	Relevance			Grammaticality			Logicality		
	Win(%)	Tie(%)	Lose(%)	Win(%)	Tie(%)	Lose(%)	Win(%)	Tie(%)	Lose(%)
Ours vs. FConvS2S	**23**	66	11	**28**	53	19	**40**	33	27
Ours vs. GPT2P	**21**	60	19	**17**	69	14	**31**	47	22

our model (Ours(0%)) achieves the best diversity and coreference performance. This demonstrates the effectiveness of our overall model. Ours performs worse than GPT2 and GPT2P in perplexity score. This indicates that our model sacrifices part of fluency for the plot control. What's more, we can see that all two-stage models perform worse in *unknown* score compared with GPT2 and GPT2P. We claim that two-stage generation and discourse relation component may repel each other.

Table 4 reports human evaluation results. Our method achieves the best scores in three metrics. Specifically, our method mainly improves scores on Logicality. This shows that our method can generate more coherent stories by utilizing discourse and coreference supervision. Our method performs similarly to GPT2P in terms of Relevance and Grammaticality. Because both methods use Transformer as the decoder and our model does not design a component to improve the relevance to the prompt. Next, we conduct ablation experiment to evaluate each component of our method.

Outline Analysis

Table 5. Comparison of different outlines.

Method	Perplexity↓	Dis-1(%)↑	Dis-2(%)↑	Unknown(%)↓	Coref Chains↑
First stage					
keyword	74.46	0.964	7.132	/	/
abstract	35.53	0.776	10.060	/	/
Second stage					
story with keyword	17.82	0.461	6.188	74.26	5.67
story with abstract	10.65	0.512	7.358	74.54	5.81

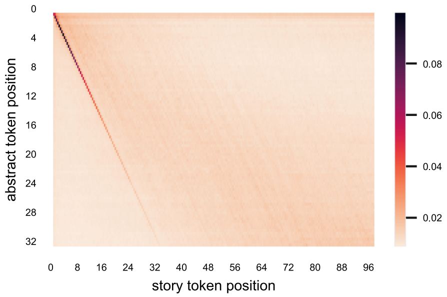


Fig. 2. The attention weight distribution of story tokens in different positions.

We compare the performance of keyword and abstract as outlines. As shown in Table 5, in the first stage keyword is more difficult to generate than abstract, for that keyword gets a higher perplexity. From the second stage, we can see that stories using abstract as outline get better scores in four metrics. This indicates that the abstract contributes to generating stories with better diversity and consistency. Therefore, we take abstract as outline in our model. In order to evaluate whether the stories are generated following the plot order of abstract, we plot story tokens' attention weight distributions on abstract tokens. The attention weight distributions are computed by averaging 2,000 generated stories. Because of the limited space, we only list tokens of the abstract and the story in the front positions. The result is shown in Fig. 2. There are several lines with darker colors in the diagonal direction of the figure. This demonstrates that the story's focus follows the plot order of the abstract and our two-stage model can control the plots of the story well.

Discourse Relation Classification

Table 6. The percentages of discourse relations with different λ_1.

TLM+Discourse	And(%)↑	When(%)↑	Unknown(%)↓
0.1	11.43	2.90	72.94
0.3	11.38	2.80	73.60
0.5	10.91	2.72	73.78

Table 7. The percentages of discourse relations of different methods.

Method	And(%)↑	When(%)↑	Unknown(%)↓
ConvS2S	8.52	2.45	76.01
FConvS2S	8.67	2.41	75.59
GPT2	10.14	2.50	74.96
GPT2P	10.96	2.60	73.61
TLM+Discourse	11.43	2.90	72.94

We evaluate the effect of discourse relation classification with **TLM+Discourse**, which enhances Transformer based language model with our proposed discourse relation classification component. We show the percentages of *and*, *when* and *unknown* for that the percentages of other relations are small. We first try several different values for λ_1 in loss function and the result is shown in Table 6. When λ_1 is high, there is no gain in three discourse metrics. When λ_1 is 0.1, we get the most gain in three discourse metrics. This indicates that large weight affects the main task and damages the performance of language model. Therefore, we use 0.1 in our final model. Then we compare **TLM+Discourse** with other baselines. The percentages of discourse relations are shown in Table 7. We can see that GPT2 and GPT2P perform better than ConvS2S and FConvS2S. TLM+Discourse generates least sentence pairs with *unknown* relation and achieves the best score among all methods. This shows that discourse relation classification component is helpful to generate more sentence pairs with certain discourse relations and further improve the coherency of generated stories.

Case Study

We analyze some generated stories to evaluate our overall model. An example is shown in Table 8. Stories generated by GPT2P have poor plot consistency and have some coreference errors, such as blue sentences and red words in Table 8. Compared with GPT2P, our model can effectively control the plot consistency of the story through the abstract. Therefore, stories generated by our model have better plot consistency. In addition, our model has fewer coreference errors than GPT2P and generates stories with better coreference consistency. What's more, the coherency between sentences is also better than GPT2P.

Table 8. Comparison of different methods.

Prompt: The zombie apocalypse is upon us but all humans are immune to the virus . All other mammals are susceptible.
GPT2P: As I sat there I tried to remember how many days it had been since the last one died . [My wife and daughter were dead, I knew they would n't be alive any time soon.]$_{1a}$ The food ran out quickly , they had to be taken away to stay in the house , but that was no reason to be angry if there were a few people left living there . After a month of food being gone a little girl sat next to me . She was the only other survivor who was n't bitten , as it was hard for us to eat them , we were all sickly . That would be another month to go and if the other animals were still alive in the food, they would n't be able to eat her . " Hey little one . " I looked up and saw her . [I had a big smile on my face at this point .]$_{2a}$ [My wife was the only one who noticed , but her smile was all that was visible of me , and not of her .]$_{1b}$ [I looked at her , and smiled sadly .]$_{2b}$ She looked sad , but did n't care . She never cared for me .
Our (generated abstract): He was the last to see the end of this world . The only reason John had survived was by his own hand , and it took much more than he could bear with a few months to heal his wounds . It took a few months for him to be able to see the end and get his arm back from the dead , and he did n't care for a few more years without having to go back . **Our (generated story)**: When he had been first laid to rest the first few days he did n't care , and he had thought it was just a small thing, but it started getting worse . He was the last to see the end of this world . The only reason John had survived was by his own hand, and it took much more than he could bear with a few months to heal his wounds . John had lived the rest of his life on Earth without anything else but the clothes on his back , and the rifle he kept at home when he was alone . The first few days were easy . A little bit of food , a bit of protection . He spent a few days walking around the city and drinking a beer , thinking he was safe , but the next few months were hell . It took a few months for him to be able to see the end and get his arm back from the dead , and he did n't care for a few more years without having to go back. It was better to go back , to be safe , so he would be safe for a while , and so he would n't get infected .

5 Conclusion

In this paper, we propose a two-stage generation model to improve the consistency and coherency of generated stories. The first stage is to build the story outline, and the second stage is to expand the outline into a complete story. What's more, we design a supplementary task of discourse relation classification to improve the discourse representation ability of the model. In addition, we enhance the model with coreference supervision to improve coreference consistency in generated stories. Experimental results on a story dataset show that our method is superior to baseline methods.

Acknowledgements. This research is supported by National Natural Science Foundation of China (Grant No. 61773229 and 6201101015), Tencent AI Lab Rhino-Bird Focused Research Program (No. JR202032), Shenzhen Giiso Information Technology Co. Ltd., the Basic Research Fund of Shenzhen City (Grand No. JCYJ20190813165003837), and Overseas Cooperation Research Fund of Graduate School at Shenzhen, Tsinghua University (Grant No. HW2018002).

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
2. Bai, H., Zhao, H.: Deep enhanced representation for implicit discourse relation recognition. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 571–583. Association for Computational Linguistics, Santa Fe (2018)
3. Barrios, F., López, F., Argerich, L., Wachenchauzer, R.: Variations of the similarity function of textrank for automated summarization. arXiv preprint arXiv:1602.03606 (2016)
4. Brown, T.B., et al.: Language models are few-shot learners. arXiv preprint arXiv:2003.04195 (2020)
5. Chen, G., Liu, Y., Luan, H., Zhang, M., Liu, Q., Sun, M.: Learning to predict explainable plots for neural story generation. arXiv preprint arXiv:1912.02395 (2019)
6. Chen, J., Zhang, Q., Liu, P., Qiu, X., Huang, X.: Implicit discourse relation detection via a deep architecture with gated relevance network. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1726–1735. Association for Computational Linguistics, Berlin (2016)
7. Clark, E., Ji, Y., Smith, N.A.: Neural text generation in stories using entity representations as context. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 2250–2260 (2018)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
9. Fan, A., Lewis, M., Dauphin, Y.: Hierarchical neural story generation. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 889–898 (2018)
10. Fan, A., Lewis, M., Dauphin, Y.: Strategies for structuring story generation. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2650–2660. Association for Computational Linguistics, Florence (2019)
11. Guan, J., Huang, F., Zhao, Z., Zhu, X., Huang, M.: A knowledge-enhanced pretraining model for commonsense story generation. Trans. Assoc. Computat. Linguist. **8**, 93–108 (2020)
12. Jain, P., Agrawal, P., Mishra, A., Sukhwani, M., Laha, A., Sankaranarayanan, K.: Story generation from sequence of independent short descriptions. arXiv preprint arXiv:1707.05501 (2017)
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

14. Lan, M., Wang, J., Wu, Y., Niu, Z.Y., Wang, H.: Multi-task attention-based neural networks for implicit discourse relationship representation and identification. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1299–1308. Association for Computational Linguistics, Copenhagen (2017)
15. Lebowitz, M.: Planning stories. In: Proceedings of the 9th Annual Conference of the Cognitive Science Society, pp. 234–242 (1987)
16. Li, B., Lee-Urban, S., Johnston, G., Riedl, M.: Story generation with crowdsourced plot graphs. In: Twenty-Seventh AAAI Conference on Artificial Intelligence (2013)
17. Li, J., Galley, M., Brockett, C., Spithourakis, G., Gao, J., Dolan, B.: A persona-based neural conversation model. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 994–1003. Association for Computational Linguistics (2016)
18. Li, P.: An empirical investigation of pre-trained transformer language models for open-domain dialogue generation. arXiv preprint arXiv:2003.04195 (2020)
19. Li, X., Li, P., Bi, W., Liu, X., Lam, W.: Relevance-promoting language model for short-text conversation. In: AAAI, pp. 8253–8260 (2020)
20. Martin, L.J., et al.: Event representations for automated story generation with deep neural nets. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
21. Mostafazadeh, N., et al.: A corpus and cloze evaluation for deeper understanding of commonsense stories. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 839–849. Association for Computational Linguistics, San Diego (2016)
22. Nie, A., Bennett, E., Goodman, N.: DisSent: Learning sentence representations from explicit discourse relations. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4497–4510. Association for Computational Linguistics, Florence (2019)
23. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics (2002)
24. PÉrez, R.P.Ý., Sharples, M.: Mexica: a computer model of a cognitive account of creative writing. J. Exp. Theor. Artif. Intelli. 13(2), 119–139 (2001)
25. Porteous, J., Cavazza, M.: Controlling narrative generation with planning trajectories: the role of constraints. In: Iurgel, I.A., Zagalo, N., Petta, P. (eds.) ICIDS 2009. LNCS, vol. 5915, pp. 234–245. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10643-9_28
26. Prasad, R., et al.: The Penn discourse TreeBank 2.0. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008). European Language Resources Association (ELRA), Marrakech (2008)
27. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8), 9 (2019)
28. Riedl, M.O., Young, R.M.: Narrative planning: balancing plot and character. J. Artif. Intell. Res. **39**, 217–268 (2010)
29. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. Text Min. Appl. Theory **1**, 1–20 (2010)

30. See, A., Pappu, A., Saxena, R., Yerukola, A., Manning, C.D.: Do massively pretrained language models make better storytellers? In: Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pp. 843–861. Association for Computational Linguistics, Hong Kong (2019)
31. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
32. Wang, Y., et al.: A large-scale chinese short-text conversation dataset. In: Zhu, X., Zhang, M., Hong, Yu., He, R. (eds.) NLPCC 2020. LNCS (LNAI), vol. 12430, pp. 91–103. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-60450-9_8
33. Xu, J., Ren, X., Zhang, Y., Zeng, Q., Cai, X., Sun, X.: A skeleton-based model for promoting coherence among sentences in narrative story generation. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4306–4315 (2018)
34. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V.: XLNet: Generalized autoregressive pretraining for language understanding. arXiv preprint arXiv:1906.08237 (2019)
35. Yao, L., Peng, N., Weischedel, R., Knight, K., Zhao, D., Yan, R.: Plan-and-write: towards better automatic storytelling. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 7378–7385 (2019)

A Hierarchical Approach for Joint Extraction of Entities and Relations

Siqi Xiao[1(✉)], Qi Zhang[2], Jinquan Sun[2], Yu Wang[2], and Lei Zhang[2]

[1] Unicom Big Data Company, Beijing, China
xiaosq15@chinaunicom.cn
[2] Alibaba Group Inc., Hangzhou, China
{mickey.zq,jinquan.sjq,tonggou.wangyu,lei.zhang.lz}@alibaba-inc.com

Abstract. Most existing approaches for the extraction of entities and relations face two main challenges: extracting overlapping relations and capturing the interactions between entity and relation extractions. In this paper, we present a novel sequence-to-sequence model with a hierarchical decoder to solve both issues elegantly and efficiently. Specifically, we use the low-level decoder to predict multi-relations and produce a relation vector for each triple. Given this relation vector, the high-level decoder generates two entities associated with the triple. In this manner, we can directly capture the interactions between entity and relation extractions. Moreover, by decomposing two tasks into two decoding phases, the overlapping multi-relations extraction can be naturally separated. Experiments on popular public datasets demonstrate that our model can effectively extract overlapping triples.

Keywords: Entity recognition · Relation extraction · Hierarchical architecture

1 Introduction

Entity and relation extraction are two fundamental tasks in natural language processing field. Detecting and extracting the structured triples from the unstructured document can support many other tasks such as knowledge base construction, information retrieval and question answering. The concrete goal is to extract triples like ⟨*rel, e1, e2*⟩ from unstructured text, which means there is a relation *rel* between entity *e1* (also called source entity) and entity *e2* (also called target entity).

Some problems still remain in the previous methodology for entity and relation extraction. First, most previous approaches can not capture the interaction between entity recognition and relation extraction. The pipeline method [5,27] regards named entity recognition (NER) and relation classification (RC) as two separate tasks, and it classifies relations based on the NER results. This kind of approach ignores the correlation between the two sub-tasks and may cause error propagation. Second, overlapping multi-relations in sentence are not handled well in many prior studies. Most of the existing methods assume that there is only one single relation in the sentence. Those approaches ignore the multi-relations scenarios, so their performance will be affected by the existence of multi-relations. Finally, prior studies about joint extraction of entities and relations are

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 710–723, 2021.
https://doi.org/10.1007/978-3-030-72113-8_47

expensive to train. These models are task-specific and cannot utilize the transfer learning method to initialize the parameter and save the computing resources.

In this paper, we propose an end-to-end joint learning method without any additional feature engineering. The proposed model with encoder-decoder architecture could capture the interaction between entity and relation extraction by introducing a hierarchical decoder in which the low-level decoder is used to generate one or more relations contained in the sentence and the high-level decoder are designed to detect entity pairs by fusing the information of relations generated from low-level decoder. In such way, extracting each triple can be separated in each low-level decoding step. Besides, our approach is configurable. We can apply different types of encoder including popular Bidirectional Encoder Representations from Transformers (BERT) proposed by [7].

Moreover, our model is equipped with hierarchical attention mechanism. One attention module in low-level decoder is used to improve the performance of relation prediction and the other in high-level decoder is designed to integrate the extracted relation to obtain more accurate entity label sequence corresponding to the current relation. Based on this unique design, our model could generate multiple triples in one sentence. Since each relation and entity pair is independently generated, our model will not be affected by the overlapping multi-relations situation.

The contributions of our work are concluded as follows:

- We introduce a novel entity and relation extractor with hierarchical attention modules, in which the low-level decoder serves as a relation generator, and the high-level decoder combines the predicted relation information to complete the entity recognition process. Through this framework, our model can handle complex overlapping multi-relations extraction.
- We design a configurable sequence-to-sequence approach for entity and relation extraction in which we can utilize any contextual feature extractor including some popular pre-trained models like BERT.
- Experiments on widely used dataset including NYT10 and NYT11 show that this method achieves the best results and further outperforms state-of-the-art with 7.0% and 7.6% improvement by introducing BERT, respectively.

2 Related Work

Recent years, many works have been contributed to entity and relation extraction, which can be roughly classified into two categories. The first class is pipeline-based methods. These methods regard name entity recognition [14,31] and relation extraction [17,30] as two independent tasks, which means relation extraction is performed based on the output of name entity recognition. Since the powerful representation and feature extraction capability of neural network, these methods prefer adopting recurrent neural network (RNN) and convolution neural network (CNN) as backbone. Given the marked target entity pairs, [22] first proposed using RNN for relation classification, and [28] used CNN for this task. [4,8,24] improved the previous works and achieved better performance on relation classification. Unfortunately, there are several shortcomings of pipeline-based method: error propagation, neglecting the correlation between two subtasks and generating redundant information.

The second class is joint learning method [6,25], which have received considerable attentions in recent years. Joint learning methods use a single model to extract entities and relations simultaneously, which can utilize the close interaction between entity extraction and relation extraction. Joint learning methods can be divided into two categories: feature-based [15,19,20] and deep learning-based. [18] used neural networks to extract entities and relations jointly. They shared the underlying encoded information and presented a sequence-based LSTM for NER, and a tree-based dependency LSTM for RC. [13] introduced attention mechanism to extract entities and relations in combination with Bi-LSTMs without using any dependency features. [32] proposed an entity relation extraction method based on a novel tagging schema, which combines entity and relation information in the tags. This method completely transforms the joint learning model into a sequence labeling problem. Although [32] can extract multiple relations, it cannot handle the overlapping relation extraction well, because each entity can only be assigned with a single label in this method.

In the past two years, several studies have been carried out on overlapping multi-relations extraction, such as [2,3,9,11]. [29] proposed an end-to-end model based on sequence-to-sequence method with copy mechanism, which can classify relation types from predefined relational tables or copy entities from original text at the specified time, but it cannot extract complete multi-word entities. [23] introduced a hierarchical reinforcement learning (HRL) framework to solve the overlapping relation extraction problem, which decomposes the task into using high-level RL for relation detection and low-level RL for entity extraction. However, the training process for HRL is time-consuming, and it is hard to converge.

From the perspective of model architecture, hierarchical recurrent network has been widely used to simulate the hierarchy of the language directly [16]. Early work applied hierarchical recurrent networks to simple algorithmic problems [10]. In recent years, it has been successfully used in image captioning [12] and video captioning [26]. In this paper, we creatively introduce hierarchical architecture for joint extraction of entities and relations.

3 Method

In this section, we will describe our hierarchical sequence-to-sequence framework with hierarchical attention for overlapping multi-relations extraction in detail. The overall structure of our model is shown in Fig. 1. In the encoding phase, the model encodes the variable-length word sequence into a fixed-length vector representation. Then the model decodes the encoding vector representation by hierarchical decoder. The low-level decoder decodes the relation at each step. The high-level decoder combined CRF layer calculates the corresponding label sequence according to the relation generated by the low-level decoder to extract entity pair. Based on this design, our model can extract multiple triples in multiple decoding steps.

3.1 Encoder

As mentioned, our model is configurable. In the encoder phase, we can utilize different contextual feature extractors. Here in this paper, we use two types of encoder. One

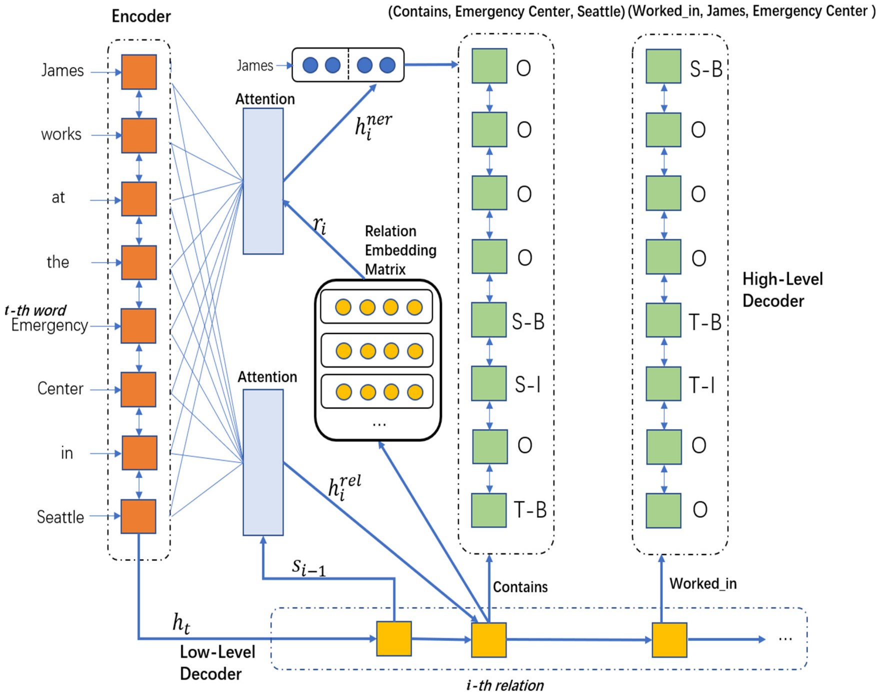


Fig. 1. The overall structure of our model. The orange blocks on the left represent the encoder, the yellow blocks represent the low-level decoder cells, and the green blocks represent the high-level decoder cells. For the i-th decoding step, the low-level decoder generates a relation *contains*, and the high-level decoder predicts an entity label sequence according to the relation.

is based on Bi-LSTMs, and the other is based on BERT, which is a pre-trained language representation model that has shown marvellous improvements across various NLP tasks. We will describe the use of two encoders, respectively.

Bi-LSTMs. Giving a text sequence $S = [w_1, w_2, ..., w_n]$, where w_t represents the t-th word in the sentence of length n. First, we transform one-hot vector of each word into embedding matrix through embedding layer, and get $E = [x_1, x_2, ..., x_n]$, where x_t means the word vector of t-th word in the sequence. We use Glove [1] to initialize our embedding layer and the parameters are updated during the training of model.

Then we use Bi-LSTMs to encode the input sequence. Bi-LSTMs consists of a forward LSTM (encoding from left to right) and a backward LSTM (encoding from right to left). LSTM network is a variant of RNN, which has the ability to model long-term dependency in the sequence. Then we concatenate $\overrightarrow{h_t}$ which is obtained by forward LSTM and $\overleftarrow{h_t}$ which is obtained by backward LSTM as the final encoding vector of the

[1] https://nlp.stanford.edu/projects/glove/.

t-th word, denoted as h_t = $[\overrightarrow{h_t}, \overleftarrow{h_t}]$. In this way, the encoding vector of each word can obtain the semantic information of its surrounding context.

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C_t} &= tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \\ C_t &= f_t * C_{t-1} + i_t * \tilde{C_t} \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{1}$$

The calculation formula of LSTM cell is shown in Eq. (1), where x_t represents the word vector of t-th word, h_t represents hidden states vector at the time t, W_f, W_i, W_c, W_o are learnable weight matrices, b_f, b_i, b_c, b_o are learnable bias vectors, σ and $tanh$ are nonlinear activation function.

BERT. BERT is a bidirectional transformer-based language representation model that is pre-trained on a large-scale corpus. Usually, for a downstream NLP task, a task-specific layer is added on top of a pre-trained BERT model. In this paper, since the following work of the model involves NER (case-sensitive), we choose the BERT-Base-Cased version as the encoder and fine-tune along with the training of the whole model. The BERT-Base-Cased structure contains 12 layers of transformer blocks.

For a given token, the input representation is constructed by summing the corresponding token, segment, and position embeddings. The first word of each input sequence is always a special classification token ([CLS]). Similar to LSTM, to obtain a fixed-dimensional pooled representation of the input sequence, we use the hidden state of the last layer of the first token as the sentence embedding to initialize the initial state of the decoder. Meanwhile, we select h_t, which represents the hidden state of the last layer of BERT at time t, as the final vector representation of the t-th word in the sentence.

3.2 Decoder

The model uses the hierarchical decoder structure with a hierarchical attention mechanism to decode relations and corresponding entity pairs.

Low-Level Decoder. A unidirectional LSTM is used to generate one or more relations contained in the input sequence.

$$o_i, s_i = cell([r_{i-1}, h_i^{rel}], s_{i-1}) \tag{2}$$

In Eq. (2), $cell$ represents an LSTM cell. In the training phase, r_{i-1} is the embedding of the target relation at the previous step in the decode sequence. In the test phase, r_{i-1} is the embedding vector of the relation predicted at the previous step, s_{i-1} is the hidden state of the previous step.

Meanwhile, the low-level attention aims to score how well the t-th input word and the output i-th relation match. h_i^{rel} represents the weighted sum of the hidden state of the encoder. We use the attention calculation method proposed by [1] as follows:

$$\begin{aligned} e_t^i &= v^{rel} tanh(W_h^{rel} h_t + W_s^{rel} s_{i-1} + b_{attn}^{rel}) \\ a^i &= softmax(e^i) \\ h_i^{rel} &= \sum_t a_t^i h_t \end{aligned} \tag{3}$$

where W_h^{rel}, W_s^{rel} and b_{attn}^{rel} are learnable parameters, h_t means the t-th hidden state of encoder when the input is the word vector x_t of the t-th word, s_i is the hidden state of low-level LSTM in decoder at step i. We get h_i^{rel} and concatenate it with relation embedding as input for the next decoding step. The output of LSTM o_i is fed through a linear layer and then we calculate the probability distribution over relation vocabulary:

$$P_{rel} = softmax(W_r o_i + b_r) \tag{4}$$

where W_r, b_r are learnable parameters. We select the relation with the highest probability as the predicted relation at the current time.

High-Level Decoder. This hierarchy is designed to generate a NER tag sequence according to the current relation. We adopt Bi-LSTMs with CRF structure, CRF can learn the transition probability between tags, which has a certain constraint on the generation of adjacent tags. The target tag set includes {O, S-B, S-I, T-B, T-I}, where S-B denotes the beginning of a source entity in a triple, S-I denotes the rest of the source entity, T-B denotes the beginning of the target entity, T-I denotes the rest of the target entity, and O denotes non-entity.

$$o_i^t, s_i^t = cell([x_t, h_i^{ner}], s_i^{t-1}) \tag{5}$$

In Eq. (5), *cell* represents LSTM cell. For two types of encoder structures, x_t has two forms. When the encoder is Bi-LSTMs, x_t is the word vector of each word. When the encoder is BERT, x_t is the hidden vector representation of the last layer of the BERT model. h_i^{ner} is the weighted sum of the hidden states of the encoder calculated by attention mechanism.

High-level attention is used to optimize NER tag prediction by making full use of the relation information. In detail, we use the relation embedding r_i of current low-level decoding step i to attend over the encoder representation, thus relation information is incorporated into h_i^{ner} to help high-level NER tag decoding. This can be formally defined as follows:

$$\begin{aligned} m_t^i &= v^{ner} tanh(W_h^{ner} h_t + W_s^{ner} r_i + b_{attn}^{ner}) \\ n^i &= softmax(m^i) \\ h_i^{ner} &= \sum_t n_t^i h_t \end{aligned} \tag{6}$$

Similarly, we concatenate x_t and h_i^{ner} as the input of high-level LSTM cell. The output of Bi-LSTMs h^{label} is transformed by a linear layer and fed to a CRF layer, and then we get the final NER label sequence $y = [y_1, y_2, ..., y_n]$ decoded by CRF as Eq. (7):

$$P(y|S) = \frac{exp(\sum_t (W_{crf}^{y_t} h_t^{label} + T^{(y_{t-1}, y_t)}))}{\sum_{y' \in Y} exp(\sum_t (W_{crf}^{y'_t} h_t^{label} + T^{(y'_{t-1}, y'_t)}))} \tag{7}$$

where T is a square transition matrix in which each entry represents transition score from one tag to another, W_{crf} is a learnable parameter, Y represents a set of all possible label sequences. We apply the Viterbi algorithm to find the label sequence with the highest probability.

3.3 Training

During training, we minimize the cross-entropy loss for relation extraction which is using low-level LSTM, denoted as L_{rel}, and CRF loss for entity pair extraction, which is high-level LSTM, denoted as L_{ner}. These two loss functions are defined as follows:

$$\begin{aligned} L_{rel} &= \frac{1}{B * I} \sum_{b=1}^{B} \sum_{i=1}^{I} -log(P_i^b(rel)) \\ L_{ner} &= \frac{1}{B} \sum_{b=1}^{B} -log(P(y|S_b)) \end{aligned} \tag{8}$$

where B is the batch size and I is the maximum step of low-level decoder, S_b is the b-th sentence in the batch.

The total loss is a linear combination of L_{rel} and L_{ner} controlled by the hyperparameter λ.

$$L_{total} = L_{rel} + \lambda * L_{ner} \tag{9}$$

The complete training process is shown in Algorithm 1.

4 Experiment

4.1 Experimental Setting

Dataset. Two widely used public datasets, NYT10 and NYT11 are used to evaluate the performance of the different model. NYT (New York Times) dataset is generated by aligning Freebase relations with the New York Times (NYT) corpus, NYT10 [21] and NYY11 [20] are two versions of this dataset. For NYT10 dataset, we use the original train and test set, in which sentences from the years 2005–2006 are for training while those from 2017 for testing. For NYT11 dataset, this dataset consists of 1.18 M sentences sampled from 294 k 1987–2007 New York Times news articles. We use the segmentation method same as [29] which randomly selects sentences as the training set,

Algorithm 1: Training procedure of our model

```
Calculate $h_t$ for each word in the sentence using encoder by Eq. (1);
Use $h_n$ (where $n$ represents the length of sentence) to initialize decoder states;
for $i = 1 \rightarrow max\_decode\_step$ do
    Calculate attention weight and context vector for relation extraction by Eq. (3);
    Get hidden state $s_i$ of low-level LSTM cell at step $i$ by Eq. (2);
    Sample relation r from $o_i$ by Eq. (4);
    Use $s_i$ to initialize high-level LSTM states;
    Calculate attention weight and context vector $h_i^{ner}$ for sequence labeling using
    relation embedding of $r_i$ by Eq. (6);
    Obtain hidden state of high-level LSTM by Eq. (5);
    Calculate label sequence using CRF layer by Eq. (6);
    if r = $\langle end \rangle$ then
        break;
    end
end
Calculate Low-level LSTM loss and High-level CRF loss by Eq. (8) and Eq. (9);
Optimize the model by Adam;
```

validation set, and test set, respectively. For both datasets, we filter sentences that do not contain valid triples. The statistical information of these two pre-processed datasets is shown in Table 1.

Table 1. Statistics of NYT10 and NYT11 datasets.

	NYT10	NYT11
Relation types	29	24
Entity tag types	5	5
Vocab size	78,350	83,002
Train set	66,823	60,000
Train triples	83,353	101,044
Test set	4006	3335
Test triples	5859	5574

Evaluation Metrics. We adopt standard micro Precision, Recall, and F1 score to evaluate the model. We do not use the label of entity types to train the model, so entity types are not considered when computing F1 score [32]. A triple is regarded as correct only when its relation type and two corresponding entities are all correct, where an entity is considered correct if the head and tail offsets are both correct.

Hyperparameters Setting. In our experiments, for LSTM cell, we set 256 as the hidden state dimension and for BERT, we use its pre-trained model version BERT-Base-Cased, following its hidden state with dimension of 768. We set the dimension of word vector as 100. Both relation type vectors and entity tag vectors are initialized randomly. The maximum low-level decoding step of our model is set to be 10, which means the model can generate up to 10 triples. And we set λ as 5. We update all model parameters by backpropagation using Adam with a learning rate of 0.001 and the batch size is set to 64. These hyperparameters are tuned on the validation set.

Baselines. Our model is compared with several influential extraction methods as following:

- **NovelTagging** [32]: a method converts the joint extraction task to a sequence labeling problem based on a novel tagging scheme where each tag contains entity and relation type information.
- **CopyR** [29]: an end-to-end model based on sequence-to-sequence learning with copy mechanism in which the decoder can generate multi-relations.
- **HRL** [23]: a hierarchical reinforcement learning framework to handle overlapping relations where high-level RL process for relation detection and low-level RL process for entity extraction.

Table 2. Comparison of results of our model and baselines on NYT10 dataset.

Model	NYT10		
	Precison	Recall	F1 score
Noveltagging	0.564	0.377	0.452
CopyR	0.510	0.426	0.465
HRL	0.716	0.578	0.639
Our LSTM-based	0.694	0.601	**0.649**
Our BERT-based	0.734	0.640	**0.684**

Table 3. Comparison of results of our model and baselines on NYT11 dataset.

Model	NYT11		
	Precison	Recall	F1
Noveltagging	0.594	0.380	0.463
CopyR	0.586	0.574	0.580
HRL	0.797	0.664	0.725
Our LSTM-based	0.775	0.742	**0.758**
Our BERT-based	0.777	0.783	**0.780**

4.2 Experimental Results

The results of different methods on NYT10 and NYT11 dataset are shown in Table 2 and Table 3. It is noticeable that our LSTM-based model (LSTM as the encoder) outperforms other methods in F1 score and achieves 1.6% and 4.6% improvements in F1 score over the state-of-the-art (HRL) [23]. After introducing BERT, our model has been further improved and exceeds HRL 7.0%, 7.6% on two datasets, respectively. From the two tables above, we can see our LSTM-based model is slightly lower than HRL in precision, but gains about 4% and 11% improvement in recall, which shows that our model is better at extracting multiple relations from sentences than HRL.

4.3 Effect on Overlapping Relation Extraction

In our experiment, we divide overlapping triple containing in a sentence into two classes: *EntityPairOverlap* and *SingleEntityOverlap*. *EntityPairOverlap* refers to a pair of entities having multiple relations. As shown in S1 in Fig. 2, entity pair ⟨*Poland, Warsaw*⟩ is shared by two relations in the sentence. *SingleEntityOverlap* means multiple triples share a single entity. Such as S2 in Fig. 2, entity *Venice* belongs to two relations in the sentence.

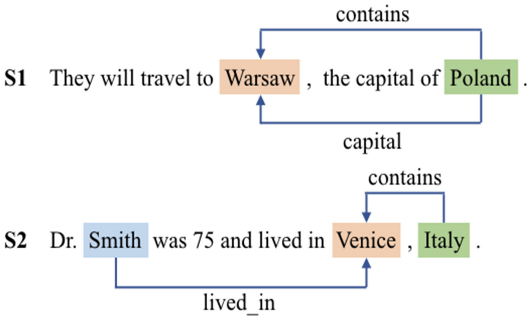

contains
S1 They will travel to Warsaw , the capital of Poland .
capital
contains
S2 Dr. Smith was 75 and lived in Venice , Italy .
lived_in

Fig. 2. Two sentences with different degrees of overlap. S1 belongs to *EntityPairOverlap* class and S2 belongs to *SingleEntityOverlap* class.

We divide the test set of NYT10 into two subsets. One subset contains sentences in *SingleEntityOverlap* category, and the other contains sentences in *EntityPairOverlap* category. Note that if a sentence contains both cases, it will appear in both subsets. Finally, we get (1) 865 sentences containing 2421 triples, which fall into *EntityPairOverlap* class (2) 245 sentences containing 723 triples, which fall into *SingleEntityOverlap* class, respectively. The experimental results are shown in Table 4 and Table 5.

As shown in the two following tables, our model performs better in extracting overlapping multiple relations in both *EntityPairOverlap* class and *SingleEntityOverlap* class. Again, after the introduction of BERT, the performance has been further improved. Compared with HRL, our model can extract more triples from data, which we believe is due to the nature of the RNN structure. We use LSTM as the decoder

Table 4. Comparison of effect of our model and baselines on *EntityPairOverlap* extraction.

Model	EntityPairOverlap		
	Precison	Recall	F1
Noveltagging	0.335	0.316	0.325
CopyR	0.524	0.332	0.406
HRL	0.820	0.487	0.611
Our LSTM-based	0.808	0.539	**0.647**
Our BERT-based	0.811	0.552	**0.657**

Table 5. Comparison of effect of our model and baselines on *SingleEntityOverlap* extraction.

Model	SingleEntityOverlap		
	Precison	Recall	F1
Noveltagging	0.279	0.268	0.273
CopyR	0.398	0.209	0.274
HRL	0.646	0.299	0.408
Our LSTM-based	0.600	0.320	**0.417**
Our BERT-based	0.643	0.356	**0.458**

and set a special identifier $\langle END \rangle$ to indicate the end of decoding. When a sentence contains multiple relations, it is more likely to generate valid relations than generating $\langle END \rangle$.

4.4 Effect on Relation Classification

To further analyze the extraction ability of the model, we compare the performance of our proposed model and baselines in relation classification on NYT11 dataset. The results are shown in Table 6.

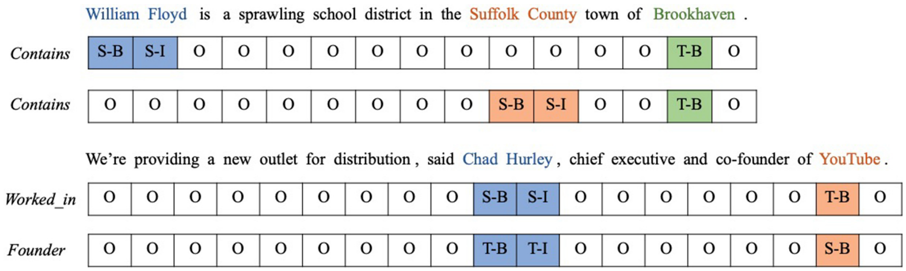
Table 6. Comparison of effect of our model and baselines on relation classification.

Model	Relation classification		
	Precison	Recall	F1
CopyR	0.863	0.846	0.855
HRL	0.909	0.774	0.836
Our LSTM-based	0.893	0.855	**0.873**
Our BERT-based	0.899	0.906	**0.903**

From Table 6, we can see that our proposed method performs better than baselines in relation classification. The performance of relation classification is improved after introducing BERT, which also proves that the vector representation of each word can obtain sufficient semantic information when using the pre-trained language model, as it is better in capturing the relations between words.

4.5 Case Study

To show the superiority of our method more explicitly, we show the prediction results of our method and select two representative examples to illustrate as Fig. 3. The first sentence comes from the test set of NYT11, which is in *SingleEntityOverlap* class. In particular, it contains two triples with the same relation *contains* and our model can still extract the two triples correctly. The second sentence is selected from the test set of NYT10 which is in *EntityPairOverlap* class. *Chad Hurley* works in *YouTube* and is also the founder of *YouTube*. When the encoder obtains the semantic encoding information, the low-level decoder decodes two relations including *Worked_in* and *Founder*, and the high-level decoder detects the corresponding entity pair ⟨*Chad Hurley, Youtube*⟩. Each triple can be generated independently by using the hierarchical decoder structure without being affected by overlapping entities.



William Floyd is a sprawling school district in the Suffolk County town of Brookhaven .

Contains	S-B	S-I	O	O	O	O	O	O	O	O	O	O	O	T-B	O
Contains	O	O	O	O	O	O	O	O	O	S-B	S-I	O	O	T-B	O

We're providing a new outlet for distribution , said Chad Hurley , chief executive and co-founder of YouTube .

Worked_in	O	O	O	O	O	O	O	O	O	S-B	S-I	O	O	O	O	O	O	T-B	O
Founder	O	O	O	O	O	O	O	O	O	T-B	T-I	O	O	O	O	O	O	S-B	O

Fig. 3. Extraction examples by our model. Words on the left below the sentence represent relations, and tables on the right represent entity label sequences based on the left relations. We use different colors to mark out source entity and target entity.

5 Conclusion

In this paper, we propose a novel sequence-to-sequence model based on hierarchical attention to extract entities and relations jointly, and effectively solve the problem of overlapping multi-relations extraction. In this hierarchical structure, the low-level decoder is used to extract one or more relations, and the high-level decoder performs the corresponding entity recognition process. Additionally, a hierarchical attention mechanism is introduced in two places to generate relations and label sequences better. The experimental results demonstrate the effectiveness of our model and outperform the state-of-the-art method. In the future, we will continue to explore how to integrate relation information into the entity recognition process better and improve the existing performance.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
2. Bekoulis, G., Deleu, J., Demeester, T., Develder, C.: Adversarial training for multi-context joint entity and relation extraction. arXiv preprint arXiv:1808.06876 (2018)
3. Bekoulis, G., Deleu, J., Demeester, T., Develder, C.: Joint entity recognition and relation extraction as a multi-head selection problem. Expert Syst. Appl. **114**, 34–45 (2018)
4. Cai, R., Zhang, X., Wang, H.: Bidirectional recurrent convolutional neural network for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, pp. 756–765 (2016)
5. Chan, Y.S., Roth, D.: Exploiting syntactico-semantic structures for relation extraction. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 551–560. Association for Computational Linguistics (2011)
6. Chen, X., Lei, X., Liu, Z., Sun, M., Luan, H.: Joint learning of character and word embeddings. In: International Conference on Artificial Intelligence (2015)
7. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
8. dos Santos, C., Xiang, B., Zhou, B.: Classifying relations by ranking with convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 626–634 (2015)
9. Eberts, M., Ulges, A.: Span-based joint entity and relation extraction with transformer pre-training. arXiv preprint arXiv:1909.07755 (2019)
10. El Hihi, S., Bengio, Y.: Hierarchical recurrent neural networks for long-term dependencies. In: Advances in Neural Information Processing Systems, pp. 493–499 (1996)
11. Fu, T.-J., Li, P.-H., Ma, W.-Y.: Graphrel: modeling text as relational graphs for joint entity and relation extraction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1409–1418 (2019)
12. Jing, B., Xie, P., Xing, E.: On the automatic generation of medical imaging reports. arXiv preprint arXiv:1711.08195 (2017)
13. Katiyar, A., Cardie, C.: Investigating LSTMs for joint extraction of opinion entities and relations. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, pp. 919–929 (2016)
14. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360 (2016)
15. Li, Q., Ji, H.: Incremental joint extraction of entity mentions and relations. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 402–412 (2014)
16. Liang, X., Hu, Z., Zhang, H., Gan, C., Xing, E.P.: Recurrent topic-transition GAN for visual paragraph generation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3362–3371 (2017)
17. Liu, Y., Wei, F., Li, S., Ji, H., Zhou, M., Wang, H.: A dependency-based neural network for relation classification. arXiv preprint arXiv:1507.04646 (2015)
18. Miwa, M., Bansal, M.: End-to-end relation extraction using LSTMs on sequences and tree structures. arXiv preprint arXiv:1601.00770 (2016)
19. Miwa, M., Sasaki, Y.: Modeling joint entity and relation extraction with table representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1858–1869 (2014)

20. Ren, X., et al.: Cotype: joint extraction of typed entities and relations with knowledge bases. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1015–1024. International World Wide Web Conferences Steering Committee (2017)
21. Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.): ECML PKDD 2010. LNCS (LNAI), vol. 6323. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15939-8
22. Socher, R., Huval, B., Manning, C.D., Ng., A.Y.: Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1201–1211. Association for Computational Linguistics (2012)
23. Takanobu, R., Zhang, T., Liu, J., Huang, M.: A hierarchical framework for relation extraction with reinforcement learning. Proc. AAAI Conf. Artif. Intell. **33**, 7072–7079 (2019)
24. Wang, L., Cao, Z., De Melo, G., Liu, Z.: Relation classification via multi-level attention CNNs (2016)
25. Yamada, I., Shindo, H., Takeda, H., Takefuji, Y.: Joint learning of the embedding of words and entities for named entity disambiguation. arXiv preprint arXiv:1601.01343 (2016)
26. Yu, H., Wang, J., Huang, Z., Yang, Y., Xu, W.: Video paragraph captioning using hierarchical recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4584–4593 (2016)
27. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. J. Mach. Learn. Res. **3**, 1083–1106 (2003)
28. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J.: Relation classification via convolutional deep neural network. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pp. 2335–2344 (2014)
29. Zeng, X., Zeng, D., He, S., Liu, K., Zhao, J.: Extracting relational facts by an end-to-end neural model with copy mechanism. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, pp. 506–514 (2018)
30. Zhang, D., Wang, D.: Relation classification via recurrent neural network. arXiv preprint arXiv:1508.01006 (2015)
31. Zhang, Y., Yang, J.: Chinese NER using lattice LSTM. arXiv preprint arXiv:1805.02023 (2018)
32. Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., Xu, B.: Joint extraction of entities and relations based on a novel tagging scheme. arXiv preprint arXiv:1706.05075 (2017)

A Zero Attentive Relevance Matching Network for Review Modeling in Recommendation System

Hansi Zeng(✉), Zhichao Xu, and Qingyao Ai

School of Computing, University of Utah, Salt Lake City, UT 84112, USA
{hanszeng,brutusxu,aiqy}@cs.utah.edu

Abstract. User and item reviews are valuable for the construction of recommender systems. In general, existing review-based methods for recommendation can be broadly categorized into two groups: the siamese models that build static user and item representations from their reviews respectively, and the interaction-based models that encode user and item dynamically according to the similarity or relationships of their reviews. Although the interaction-based models have more model capacity and fit human purchasing behavior better, several problematic model designs and assumptions of the existing interaction-based models lead to its suboptimal performance compared to existing siamese models. In this paper, we identify three problems of the existing interaction-based recommendation models and propose a couple of solutions as well as a new interaction-based model to incorporate review data for rating prediction. Our model implements a relevance matching model with regularized training losses to discover user relevant information from long item reviews, and it also adapts a zero attention strategy to dynamically balance the item-dependent and item-independent information extracted from user reviews. Empirical experiments and case studies on Amazon Product Benchmark datasets show that our model can extract effective and interpretable user/item representations from their reviews and outperforms multiple types of state-of-the-art review-based recommendation models.

Keywords: Review modeling · Interaction-based model · Relevance matching

1 Introduction

Review text is considered to be valuable for effectively learning user and item representations for recommender systems. Previous studies show that the incorporation of reviews into the optimization of recommender systems can significantly improve the performance of rating prediction by alleviating data sparsity problems with user preferences and item properties expressed in review text [5,19,20,41]. In general, existing review based recommender systems for rating prediction can be roughly categorized into two groups: 1) The siamese models that independently encode static user and item representations from reviews and use the static representations to predict the rating [5,41]; 2) The

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 724–739, 2021.
https://doi.org/10.1007/978-3-030-72113-8_48

interaction-based models that dynamically learn the user and item representations based on their context [10,31]. In particular, the interaction-based models assume that, given different target items, different user reviews might play different roles in determining the utility of the items. For example, when the target item is about an album from the Led Zeppelin, the user review that reflects her interest on Rock & Roll music might be more useful than the rest of her reviews.

Although the interaction-based models have more model capacity and fit the human purchasing behavior better [31], several problematic model designs and assumptions lead to its lower performance than siamese models as shown in recent studies [25]. First, most existing interaction-based models exploit co-attention mechanism [6,26,30,36] to distill textual similarity information between user and item reviews, but such information might be diluted when there is a vast amount of text in user and item reviews. Second, because the number of reviews to profile each user in the training set is limited, it is common that the target item's characteristics are beyond the interest of a user expressed in her limited reviews. Interaction-based models that force the user representations to extract valuable information from user reviews for the target item might introduce irrelevant aspects of the user and cause serious overfitting. Third, existing interaction-based models extract user-item relationships mostly by modeling the textual similarity between user and item reviews. High textual similarities between user and item reviews, however, not necessarily reflect the user's true opinion on the target item. For example, an item review "the taste of cappuccino is really good" might have higher textual similarity with a user review "I really enjoy the taste of beef pho" than with "I am a big coffee fan", but the latter review that reflects the user opinion on coffee could be more informative when predicting her rating on the target item.

Based on these observations, in this paper, we propose a new interaction-based rating prediction model to mitigate the weakness of the existing interaction-based recommendation models. First, we implement a relevance matching model [11] instead of a semantic matching model [36] to search the relevant review from the user to the target item. Our relevance matching model treats each user review as a query to search and extracts relevant information from all the reviews of the target item. It is capable of discovering relevance information from a large amount of review text with thousands or more words. Second, to better capture the semantic relationships instead of the textual similarity between user and item reviews, we use the *ground-truth* review (available in the training stage) written from the user to the target item and the corresponding item reviews as a pair of positive "query-document" to train our relevance matching module and plug it as the auxiliary loss in the training objective, since the *ground-truth* review expresses the user true opinion to the target item. After the relevance matching function is well-trained, other user reviews that have high relevance matching scores to the target item would share similar characteristics to the *ground-truth* review and also reflect the user true interest to the target. Last but not least, when there is not relevant review from the user to the target item, we exploit a zero-attention network [1] to avoid using irrelevant reviews to build user representations. Our zero-attention network not only builds dynamic user representations when there are high informative reviews from the user to the target item, but also allows the model to degenerate to a siamese model with static user representations when all user reviews are not relevant to the target item. Specifically, separated from the interaction module, we build static user and item embeddings using a multi-layer convolutional self-attention network to extract information hierarchically from words, sentences and reviews. We then construct the final user representations using both the dynamic user representations extracted by the interaction module and the static user embeddings created by the self-attention network. When there is no user review relevant to the target item, the dynamic user representation created by the interaction module with zero attention networks would be downgraded to a zero vector and the final rating prediction of the user-item pair would purely depend on the static user and item embeddings. Empirical experiments and case studies on four datasets from Amazon Product Benchmark show that our proposed model the Zero Attentive Relevance Matching Network (ZARM) can extract effective and interpretable user/item representations from review data and outperforms multiple types of state-of-the-art review-based recommendation models.

2 Related Works

Review Based Recommendation. Using review text to enhance user and item representations for recommender system has been widely studied in recent years [9,14,19,20,23,29,39,40]. Many works are focus on topic modeling from review text for users and items. For example, the HFT [20] uses LDA-like topic modeling to learn user and item parameters from reviews. The likelihood from the topic distribution is used as a regularization for rating prediction by matrxi factorization (MF). The RMR [19] uses the same LDA-like model on item reviews but fit the ratings using Guassian mixtures but not MF-like models. Recently, with the advance of deep learning, many recommendation models start to combine neural network with review data to learn user and item representations, including DeepCoNN [41], TransNets [4] D-Att [27], NARRE [5], HUITA [33], HANN [7], MPCN [31], AHN [10], HSACN [38]. Despite their differences, existing work using deep learning models for review modeling can be roughly divided into two styles – siamese networks and interaction-based networks. For example, DeepCoNN [41] uses two convolution neural networks to learn user and item representations from reviews statically; NARRE [5] extends CNN with an attention network over review-level to select reviews with more informativeness. In addition, the MPCN [31], a interaction-based network, uses co-attention mechanism to select the most informative and matching reviews from the user and item respectively, then another attention mechanism is applied to learn the fixed dimensional representation, by modeling the word-level interaction on the matched reviews. However, both of these two styles have their own weaknesses. The siamese models lack the dynamic target-dependent modeling and neglect the interaction between the user and target. But the interaction-based models forcely require the dynamic matching between each user and item, neglecting the fact that not every user exists the informative review to the target. Even the informative review exists, the matching information might be diluted considering thousands of words within tens of review for profiling the user and item.

Interaction Based Text Matching. The review based dynamic user-item modeling is closely related to query-document representation learning in the QA [26] task or premise-hypothesis encoding in the NLI [6,30,36] task that exploit the co-attention mechanism. The co-attention mechanism computes the pairwise similarity between two sequences, builds the pair-wise attention weights, and integrates them with other feautres of the sequences for effective text semantic matching learning. Besides the text semantic matching in the NLP tasks, several works on the IR tasks [2,11,12,34] also utilize the interaction-based approaches for text relevance matching learning. For example, DRMM [11] proposes a pooling pyramid technique that converts the pair-wise similarity matrix into the histogram, and use it as feature for final text matching prediction. Based on the DRMM, K-NRM [34] introduces the kernel-based differential pooling technique that can learn the matching signal in different level. Recent work [22] further investigate using the semantic matching and relevance matching together or alone in the NLP and IR tasks. It finds that using relevance matching alone performs reasonable well in many NLP tasks but the semantic matching is not effective for IR tasks.

Rethinking the Progress of Deep Recommender System. While we have witnessed the rapid advancements of deep learning methodology and its applications on the field of recommender systems, there are worries about the progress we made. Dacrema et al. [8] investigated the performance of several recent algorithms proposed in top conferences and found most of them can not compete with traditional methods, like Matrix Factorization and its derivative models [15,18,35], BPR [24] or Item-KNN. Furthermore, Sachdeva et al. [25] focused on the usefulness of reviews. He examined several review-based recommendation algorithms and found that applying complex structures to extract semantic information in reviews, not necessarily improve the system's performance. Our goal is to try to tackle these existing problems in this field and propose an interpretable method to effectively utilize the review information.

3 Proposed Method

3.1 Overview

The goal of the proposed model is to predict the rating from the user to the target item based on their review text. The architecture of our model is shown in Fig. 1. Our model contains two parallel encoders that use multi-layer convolution self-attention network to hierarchically encode user and item static representations from their reviews respectively. Besides, the model has a interaction module that encodes the user dynamic representation according to her current interacted item where we first compute the relevant level of each user review to the target item by the relevance matching function. Then the zero-attention network is applied

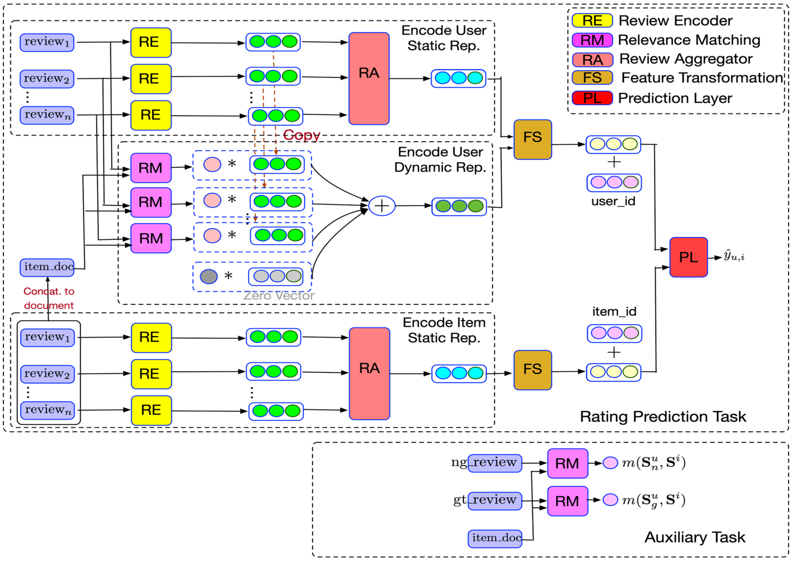


Fig. 1. Overview of our model structure

to allow the dynamic user representation degrade to a zero vector when there is no user review relevant to the target, in which case the final user representation would purely depend on the static user representation. The encoded user static and dynamic representations will be concatenated and be taken as the input to the feature transformation layer to encode the final user representation. On the rightmost of the model, the prediction layer is added to let the learned user and item final representations interact with each other and compute the final rating prediction. In the training stage, the auxiliary loss is plugged to guide the training of relevance matching function. In the following sections, we will introduce the static user/item encoder (Sect. 3.2), dynamic user encoder composed of the relevance matching function and zero-attention network (Sect. 3.3), prediction layer(Sect. 3.4), and the training objective (Sect. 3.5) in details.

3.2 Static User/Item Encoder

Since the static user and item encoder only differ in their inputs, we introduce the process of encoding user static representation in the following in details. And the same process is applied to static item encoder in the similar way. Assume the input of the user encoder is $\{r^u_1, \dots r^u_N\}$, where N is the number of reviews written by the user. We learn each review representation hierarchically from word-level to sentence-level. More specifically, a user review $r^u = \{s_1, \dots s_T\}$ consists of T sentences, and each sentence s_i is composed of a sequence of L words $\{w^i_1, \dots w^i_L\}$. To learn the sentence representation $\boldsymbol{s_i}$, we apply the word-level self-attentive convolution network to encode the contextual representation of each word in the sentence and use the attention network to aggregate the learned contextual embeddings in to a single vector. Mathematically, we first apply the word embedding layer to map each word w_j^i into a vector $\boldsymbol{w}_j^i \in \mathbb{R}^{d_w}$ to form a sequence of word embeddings $\boldsymbol{W}^i \in \mathbb{R}^{d_w \times L}$, then we apply the word-level convolution neural network to learn the local semantic representation of each words:

$$\boldsymbol{Q}_w^i = \mathrm{CNN}_w^Q(\boldsymbol{W}^i),\ \boldsymbol{K}_w^i = \mathrm{CNN}_w^K(\boldsymbol{W}^i),\ \boldsymbol{V}_w^i = \mathrm{CNN}_w^V(\boldsymbol{W}^i) \tag{1}$$

where $\boldsymbol{Q}_w^i,\ \boldsymbol{K}_w^i,\ \boldsymbol{V}_w^i \in \mathbb{R}^{d_w \times L}$. To enrich each word semantic representation and capture long-range dependencies between words, we apply the multihead-self-attention network [32] on top of the learned word local representation from CNN(·). Finally, a 1 layer feed-forward network is sequentially plugged to learn more flexible representations:

$$\boldsymbol{Z}_w^i = \mathrm{FFN}_w\Big(\text{Multihead-Self-Attention}_w(\boldsymbol{Q}_w^i, \boldsymbol{K}_w^i, \boldsymbol{V}_w^i)\Big) \tag{2}$$

where $\boldsymbol{Z}_w^i \in \mathbb{R}^{d_s \times L}$. Then we use addictive-attention network [37] to aggregate the contextual representations into a single vector $\boldsymbol{s_i} \in \mathbb{R}^{d_s}$ for sentence modeling:

$$\boldsymbol{s_i} = \text{Addictive-Attention}_w(\boldsymbol{Z}_w^i) \tag{3}$$

We apply the same procedure on each sentence of the review r^u to form a sequence of sentence representations $\boldsymbol{S} = [\boldsymbol{s}_1, \ldots \boldsymbol{s}_T] \in \mathbb{R}^{d_s \times T}$. Then we take the sentence sequences as an input to sentence-level self-attentive convolution network with addictive-attention network to form the review representation $\boldsymbol{r}^u \in \mathbb{R}^{d_r}$:

$$\boldsymbol{Z}_s = \text{Multihead-Self-Attention}_s\Big(\mathrm{CNN}_s^Q(\boldsymbol{S}), \mathrm{CNN}_s^K(\boldsymbol{S}), \mathrm{CNN}_s^V(\boldsymbol{S})\Big) \tag{4}$$

$$\boldsymbol{r}^u = \text{Addictive-Attention}_s(\boldsymbol{Z}_s) \tag{5}$$

We apply the same hierarchical network on each review written by the user, then form a sequence of review representations $\boldsymbol{R} = [\boldsymbol{r}_1^u, \ldots \boldsymbol{r}_N^u] \in \mathbb{R}^{d_r \times N}$. Finally, we apply the user-level addictive-attention network to aggregate the information of these reviews and form a single vector $\boldsymbol{u}^{static} \in \mathbb{R}^{d_r}$ to form the user static representation:

$$\boldsymbol{u}^{static} = \text{Addictive-Attention}_u(\boldsymbol{R}^u) \tag{6}$$

The item static representation $\boldsymbol{i}^{static}$ can be obtained using a similar procedure.

3.3 Dynamic User Encoder

To learn the dynamic user representation, we first compute the relevant scores of the reviews of the user to the target item using relevance matching function. In other words, given N reviews written by the user $\{r_1^u, \dots r_N^u\}$, we want to compute their corresponding relevance scores $\{\alpha_1, \dots \alpha_N\}$ to the target. The detailed introduction of the relevance matching function is in the following.

Relevance Matching Function: The input of the function is a query-document pair where we treat the user review as a query and the target item reviews as a document, and we denote the function as $m(\cdot, \cdot)$. Formally, each user review r_k^u can be alternatively represented as a sequence of word embeddings $[\boldsymbol{e}_1^u, \dots, \boldsymbol{e}_M^u] := \boldsymbol{S}_u^k$, and the item document is a concatenation of a sequence of word embeddings of its each review $[\boldsymbol{e}_1^i, \dots, \boldsymbol{e}_M^i, \dots, \boldsymbol{e}_{(N-1)M+1}^i, \dots, \boldsymbol{e}_{NM}^i] := \boldsymbol{S}_i$, where $\boldsymbol{e}_k^u$, $\boldsymbol{e}_k^i \in \mathbb{R}^{d_w}$, $\boldsymbol{S}_u^k \in \mathbb{R}^{d_w \times M}$, $\boldsymbol{S}^i \in \mathbb{R}^{d_w \times MN}$, d_w is the dimension of the word embedding, M is the review length, and N is the number of review from the target item. To get the relevant matching score from the k-th user review to its target item, we first compute the word similarity matrix S:

$$\boldsymbol{M} = \boldsymbol{S}_u^{k^T} \boldsymbol{S}_i \in \mathbb{R}^{M \times MN} \tag{7}$$

where $\mathbf{M}_{i,j}$ can be considered as cosine similarity score (we normalize it into cosine space) by matching i-th word of user review with j-th word of item document. We apply mean pooling and max pooling on every row of similarity matrix to obtain discriminate features:

$$mean(\mathbf{M}) = \begin{bmatrix} mean(\mathbf{M_{1:}}) \\ \dots \\ mean(\mathbf{M_{n:}}) \end{bmatrix} \in \mathbb{R}^M \;, max(\mathbf{M}) = \begin{bmatrix} max(\mathbf{M}_{1:}) \\ \dots \\ max(\mathbf{M}_{n:}) \end{bmatrix} \in \mathbb{R}^M \tag{8}$$

Also, we consider the relative important score for each word in the user review $\boldsymbol{S}_u^k$ by applying a function $imp(\cdot)$:

$$imp(\mathbf{S}_u^k) = \begin{bmatrix} imp(\mathbf{e}_l^u) \\ \dots \\ imp(\mathbf{e}_l^u) \end{bmatrix} \in \mathbb{R}^M \quad \text{where,} \quad imp(\boldsymbol{e}_j^u) = \frac{\exp(\boldsymbol{w}_p^T \boldsymbol{e}_k^u)}{\sum_{o=1}^n \exp(\boldsymbol{w}_p^T \boldsymbol{e}_o^u)} \tag{9}$$

where $\boldsymbol{w}_p \in \mathbb{R}^{d_w}$, then the input feature for scoring function parameterized by a 2 layer feed-forward neural network is:

$$\boldsymbol{I}^{rel} = \begin{bmatrix} imp(\boldsymbol{S}_u^k) \odot mean(\mathbf{M}) \\ imp(\boldsymbol{S}_u^k) \odot max(\mathbf{M}) \end{bmatrix} \in \mathbb{R}^{2M} \tag{10}$$

Hence the relevant score between the k-th user review $\boldsymbol{S}_k^u$ and item document $\boldsymbol{S}^i$ is:

$$m(\boldsymbol{S}_k^u, \boldsymbol{S}^i) = \text{FFN}\big(\text{FFN}(\boldsymbol{I}^{rel})\big) = \alpha_k \in [-\infty, \infty] \tag{11}$$

When there is no user review relevant to the target item, we can expect each relevant score $\alpha_k \ll 0$. However, if we naively normalized the relevant scores, and use them as weights to measure the importance of each user review, the final dynamic user representation we get by weighted sum of the user review representations would be a non-zero vector. It is due to the fact that after the normalized process, every relevant score will be assigned as a probability measure, and summation of these probabilities being 1 makes the situation that every normalized relevant weight $a_k \approx 0$ become impossible, hence the weighted sum of the user reviews cannot be a zero vector. For example, suppose that the relevant scores of all user reviews are $\alpha_k = -100$, where $k = 1, \dots N$, then the normalized relevant score would be $\alpha_k = \frac{1}{N}$. The dynamic item representation will become $\boldsymbol{u}^{dynamic} = \sum_{k=1}^{N} \frac{1}{N} \boldsymbol{r}_k^u$, which is not a zero vector even when all user reviews are not relevant to the target item. To resolve the problem, we use the zero-attention network motivated by [1].

Zero-Attention Network: we introduce a zero score $\alpha_0 = 0$, and re-normalize the relevant scores by taking the zero score in to account. Formally, $\hat{\alpha}_k = \frac{\exp(0)+\exp(\alpha_k)}{\exp(0)+\exp(\alpha_1)+\dots\exp(\alpha_m)} = \frac{1+\exp(\alpha_k)}{1+\exp(\alpha_1)+\dots\exp(\alpha_m)}$, $k = 1, \dots N$, and $\hat{\alpha}_0 = \frac{1}{1+\exp(\alpha_1)+\dots\exp(\alpha_m)}$, then the user dynamic representation is,

$$\mathbf{u}^{dynamic} = \sum_{k=1}^{N} \hat{\alpha}_k \mathbf{r}_k^u + \hat{\alpha_0} \mathbf{0} \tag{12}$$

Intuitively, when $\alpha_k \ll 0$, the normalized score $\hat{\alpha}_k \approx 0$ for all $k = 1, \dots, N$, and the $\boldsymbol{u}^{dynamic} \approx \mathbf{0}$, which is close to a zero vector. In the other hand, if there exist a large relevant score, for example $\alpha_k = 10$ for a certain k, the effect of $\alpha_0 = 0$ will be very low, and the normalized score will be $\hat{\alpha}_k \approx 1$, and $\boldsymbol{u}^{dynamic} \approx \boldsymbol{r}_k^u$

3.4 Prediction Layer

This layer combine the static and dynamic user representations to form a final user representation learned from reviews. Also, it learns a final item static representation from reviews by 1-layer feed-forward neural network:

$$\boldsymbol{u}^r = \text{Relu}\Big(\Big[\mathbf{W}_u^{static}, \mathbf{W}_u^{dynamic} \Big] \begin{bmatrix} \boldsymbol{u}^{static} \\ \boldsymbol{u}^{dynamic} \end{bmatrix} + \mathbf{b}_u \Big) \tag{13}$$

$$\boldsymbol{i}^r = \text{Relu}\Big(\boldsymbol{W}_i^{static} \boldsymbol{i}^{static} + \mathbf{b}_i \Big) \tag{14}$$

where $\boldsymbol{W}_u^{static}$, $\boldsymbol{W}_u^{static}$, $\boldsymbol{W}_i^{dynamic}$, $\boldsymbol{W}_i^{static} \in \mathbb{R}^{d_h \times d_r}$, $\boldsymbol{b}_u$, $\boldsymbol{b}_i \in \mathbb{R}^{d_h}$. Finally, we combine the user and item id embeddings $\boldsymbol{u}^{id}$, $\boldsymbol{i}^{id} \in \mathbb{R}^{d_h}$, with user and item embeddings learned from reviews $\boldsymbol{u}^r$, $\boldsymbol{i}^r$, to form their final representations, which are $\boldsymbol{u} = \boldsymbol{u}^r + \boldsymbol{u}^{id},\ \boldsymbol{i} = \boldsymbol{i}^r + \boldsymbol{i}^{id}$. We take the user and item embeddings as input to get the final rating prediction:

$$\hat{y}_{u,i} = \boldsymbol{w}_f^T(\boldsymbol{u} \odot \boldsymbol{i}) + b_u + b_i + b_g \tag{15}$$

where $\boldsymbol{w}_f \in \mathbb{R}^{d_h},\ b_u,\ b_i,\ b_g \in \mathbb{R}$

3.5 Training Objective

Besides a regression loss for the rating prediction, an auxiliary loss is utilized for better training the relevance matching function. Specifically, we assumed there is a user-item pair (u, i) with the *ground-truth* rating $y_{u,i}$ in the training stage, and the *ground-truth* review $r^g_{u,i}$ written from the user to the target item is treated as a "positive query" to the target item. Also we randomly sample a review from the different user different item as a "negative query" to the target item which is $r^n_{u,i}$. The corresponding word sequence representation of the *ground-truth* review, negative review and target item document is $\boldsymbol{S}^u_g,\ \boldsymbol{S}^u_n,\ \boldsymbol{S}^i$. Ideally, a good relevance matching function $m(\cdot, \cdot)$ can distinguish the positive query-document pair from the negative one, in other words, we wish $m(\boldsymbol{S}^u_g, \boldsymbol{S}^i) > m(\boldsymbol{S}^u_n, \boldsymbol{S}^i)$. In the same time, we want to minimize the regression loss between ground-truth rating $y_{u,i}$ and predicted rating $\hat{y}_{u,i}$ computed from Eq. 15. To achieve the above two goals, we write the objective function as followed,

$$\mathit{loss} = \sum_{\{(u,i)\} \in \mathcal{S}} \underbrace{\Big(y_{u,i} - \hat{y}_{u,i}\Big)^2}_{\text{regression loss}} - \underbrace{\Big(\log\big(m(\boldsymbol{S}^u_g, \boldsymbol{S}^i)\big) + \log\big(1 - m(\boldsymbol{S}^u_n, \boldsymbol{S}^i)\big)\Big)}_{\text{auxiliary loss}}$$

4 Experimental Setup

Datasets and Evluation Metrics. We conduct our experiment on four different categories of 5-core Amazon product review datasets [13]. The statistics of these four categories are shown in the first and second columns of the Table 1. For each dataset, we randomly split user-item pairs into training, validation, and testing sets with ratio 8:1:1. We use NLTK [3] to tokenize sentences and words of reviews. We let the number of reviews be the same for profiling user and item where the number of reviews is set to cover 90% of users for the balance of efficiency and performance. We adopt Mean Square Error (MSE) as the main metric to evaluate the performance of our model. The source code can be found here[1].

Compared Methods. To evaluate the performance of our method, we compare it to several state-of-the-art baseline models: (1) **MF** [17]: a basic but well-known CF model that predict the rating using inner product between user, item hidden

[1] https://github.com/HansiZeng/ZARM.

representations plus user, item and global bias; (2) **NeurMF** [15]: the CF based model combines linearity of GMF and non-linearity of MLPs for modeling user and item latent representations; (3) **HFT** [20]: the topic modeling based model combines the ratings with reviews via LDA; (4) **DeepCoNN** [41]: the CNN based model uses two convolution neural network to learn user and item representation; (5) **NARRE** [5]: the CNN based model modifies the DeepCoNN by using the attention network over review-level to select reviews with more informativeness. (6) **MPCN** [31]: the model that selects informative reviews from user and item by review-level pointers using the co-attention technique, and selects informative word-level representations for the rating prediction by applying word-level pointers over the selected reviews; (7) **AHN** [10]: a dynamic model using co-attention mechansim but treats user and item asymmetrically; (8) **ZARM-static**: the variant of the ZARM that only user static representations; And (9) **ZARM-dynamic**: the variant of the ZARM that only uses user dynamic representations.

Parameter Settings. We use 300-dimensional pretrained word embeddings from Google News [21], and employ the Adam [16] for optimization with an inital learning rate 0.001. We set the dimension of sentence hidden vector and review hidden vector as 100, and the latent dimension of the prediction layer as 32. Also, the convolution kernel size is 1 or 3 based on the performance in each dataset, and number of head for each self-attention layer is 2. We apply dropout after the word embedding layer, after each feed forward layer in sequence encoding modules, and before the prediction layer with rate $[0.2, 0.3, 0.5]$. The hidden dimension of the two layer neural network in the Relevance Matching Module is set to 16. The hyper-parameters of baselines are set following the settings of their original papers.

5 Results and Analysis

The MSE results of compared models are shown in Table 1. Based on the results, we can make several observations. Firstly, the siamese models outperform the interaction-based models significantly. As discussed previously, due to the fact that not every user exists informative review to the target, interaction-based models that force to extract informative reviews from user data will suffer from heavily over-fitting. Among siamese networks, we observe that the ZARM-static outperforms the other siameses models. This demonstrates that ZARM-static can capture the review hierarchical structure and use attention neural network to select the important information in each level. Among interaction-based models, ZARM-dynamic outperforms the other baselines such as MPCN and AHN. This demonstrates the effectiveness of the relevance matching component in discovering relevant information from vast review text and the utility of the auxiliary training loss that makes the found relevant review more aligned with the *ground-truth* that reflect the user true opinion on the target item. Finally, our model (i.e., ZARM) shows consistently improvement over siamese and interaction-based models across all datasets. Our model uses the zero-attention network that can

Table 1. Experiment results on benchmark datasets. † and ‡ represents the best performance among siamese and interaction-based models, respectively. The bold value is the best performance among all models in each dataset.

Dataset #Reviews / #Users / #Items		Toys & games 167k/19k/12k	Video games 232k/24k/11k	Kindle store 983k/68k/62k	Office products 53k/2k/4k
Non-text-based	MF	0.8010	1.0979	0.6231	0.6954
	NeuMF	0.8012	1.0931	0.6255	0.6941
Siamese	HFT	0.7947†	1.0837	0.6172	0.6881
	DeepCoNN	0.8273	1.1241	0.6437	0.7102
	NARRE	0.7982	1.0881	0.6199	0.6794
	ZARM-static	0.7952	1.0774†	0.6159†	0.6757†
Interaction-based	MPCN	0.8199	1.1062	0.6337	0.7101
	AHN	0.8233	1.1137	0.6341	0.7341
	ZARM-dynamic	0.8054‡	1.1054‡	0.6279‡	0.7024‡
Hybrid	ZARM	**0.7881**	**1.0632**	**0.6083**	**0.6695**

Table 2. Ablation study (validation MSE) on four datasets

Architecture	Toys-and-games	Video-games	Kindle-store	Office-product
Default	0.7897	**1.0611**	**0.5961**	**0.6731**
(1) max pooling	0.7922	1.0645	0.6075	0.6795
(2) avg embedding	**0.7854**	1.0641	0.6043	0.6742
(3) Remove pos. vec	0.7913	1.0654	0.5985	0.6755
(4) Remove u/i bias	0.8021	1.0713	0.6022	0.6761
(5) Remove aux. loss	0.8147	1.0944	0.6189	0.6893
(6) Add item dyn	0.7938	1.0695	0.6053	0.6800

build dynamic user representations from reviews when there are high informative reviews and can easily degrade to user static representations when there is not. This strategy combines the advantages of both the siamese and interaction-based models.

5.1 Ablation Studies

We conduct the ablation study on the validation sets of the four benchmark datasets. We report the performance of 6 variant models from the defualt model setting: (1) we change the static review aggregator in Eq. 6 to max pooling; (2) we encode each review using average embedding of words; (3) We use the relative position representations [28] to encode the relative position between entities in the self-attention network, now we remove the position encoding vectors to conduct ablation study; (4) we remove the user and item bias in Eq. 15; (5) we remove the auxiliary loss in the training objective; And (6) we make our user and item representations symmetrically by adding the dynamic item encoding to represent the item.

As shown in Table 2, the performance of ZARM would drop when we use max pooling in the aggregator, remove the position encoding vectors in self-attention network, or remove the u/i bias. Using average word embeddings for review embeddings achieves suboptimal performance on most datasets, but it also outperforms the default ZARM on Toys-and-Games, which indicates that such simple aggregators may have some value on specific data types. Interestingly, in our experiments, the variant architecture that encodes item using its dynamic and static representation underperforms the default ZARM which only use the item static encoding. This indicates that building interaction-based representations on the item side may not as profitable as they are on the user side, or the current interaction module is not suitable for the construction of dynamic item representations.

5.2 Behavior of the Dynamic Interaction Matching

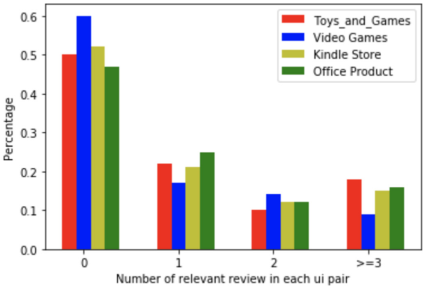


Fig. 2. The distribution of user-item pairs with different numbers of relevant review ($\alpha_k > 0$).

We conduct several experiments on investigating the behaviors of the dynamic interaction matching Firstly, we investigate the number of relevant reviews ($\alpha_k > 0$ in Eq. (11)) each user have to the target item as shown in Fig. 2. Although the number of relevant review from the user to the target is dataset dependent, there are around 50% of user-item pairs do not have the relevant review in each dataset where the type of pairs in Video Games dataset account for 60% the most, and in Office Product account for 48% the least. On the other hand, some users have more than one relevant reviews to their target items. For example, in the Toys & Games dataset, 15% of the user-item pairs have relevant review more than 3. Such observation implies that some users have consistent interests and tend to buy items with similar characteristics, which lead to their target item matched to her multiple history items in high possibility.

To further analyze the interaction module in our model, we randomly sample 40 users and their corresponding target items in the validation set for case studies. For each user we visualize the zero score and the relevant score of each review to the target item (from r1 to r11) as shown in Fig. 3. We observe that there are roughly half of the user-item pairs having large zero score which is larger than 0.5. On the other hand, there are some users containing reviews with high relevant scores like the pair (user5, item5), (user36, item36) with review id r6, r9. We then take a closer look into the two high relevant review r6, r9 and their corresponding target item documents as shown in Table 3. We observe that the high relevant reviews and their target item documents share multiple similar keywords, and these keywords are highly informative that can describe the item

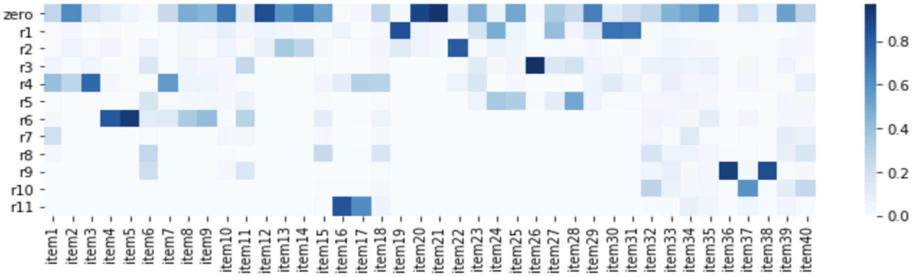


Fig. 3. The distribution of relevance matching scores of each user review given a user-item pair.

characteristics to a large extent. For example, the keyword "Gyro Hercules" in the r6 and "Gyro Hercules helicopter" in its corresponding target item document have high textual similarity and describe the general characteristics of the two items that that r6 and target item belong to. Moreover, The user true opinion on the target item can be reflected in the high relevant reviews from the user to target. For example, the first target item (item5) which has the advantage of "keep on going and not falls down" meets the user interest that is shown in r6 that mentions she likes a helicopter that is "truly withstand a hard fall". And the second target item (item 36) which is suitable for kid Christmas gift conforms to the user interest reflected in r9 in which she mention that she needs a Christmas gift for her 3-year-old granddaughter.

Table 3. Examples of high relevant reviews r6, r9 and their corresponding target item documents. The first column is their complete user review, and the second column are sampled text selected from the item documents.

User review	Target item document
I have bought other remote **control helicopters** only to take them outside and have a little breeze of wind **knock it down and break**. With the **Gyro Hercules** it can **truly withstand a hard fall** so you can fly it nearly anywhere	My kids demolish other helicopters/**keeps on going and not falls down**/helicopter/**Gyro Hercules helicopter**/ it is durable enough I can't even break it with my terrible skills.
Bought for our **Granddaughter(she is 3)** for **Christmas**. She just loves the write on wipe off A, B, C's and 1, 2, 3's. The **art projects** that were included and quality of the items for the project, TERRIFIC! Would recommend for all 3 year olds	This is perfect for a rainy day **Christmas Vacation**/**my three yr old LOVES crafts**/Filled with all the supplies to make 16 high quality crafts

6 Conclusion

We propose a new model ZARM for the review based rating prediction task. In our model, the interaction module based on relevance matching function with zero-attention network is utilized to learn user dynamic representation in more flexible way. And the auxiliary loss plugged into the training object make the relevance matching function better trained. Experiments on the four Amazon benchmark datasets show our model can outperform the state-of-art models based on the siamese network and interaction-based network. By conducting case studies, we take a deeper look into the behavior of our interaction module, and investigate the several statistical and semantic characteristics of the relevant reviews for users to targets extracted by the interaction module.

Acknowledgements. This work was supported in part by the School of Computing, University of Utah and in part by NSF IIS-2007398. 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. Ai, Q., Hill, D.N., Vishwanathan, S., Croft, W.: A zero attention model for personalized product search. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (2019)
2. Bi, K., Ai, Q., Croft, W.B.: A transformer-based embedding model for personalized product search. In: Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3397271.3401192
3. Bird, S.: NLTK: the natural language toolkit. ArXiv cs.CL/0205028 (2006)
4. Catherine, R., Cohen, W.: Transnets: learning to transform for recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems (2017)
5. Chen, C., Zhang, M., Liu, Y., Ma, S.: Neural attentional rating regression with review-level explanations. In: Proceedings of the 2018 World Wide Web Conference (2018)
6. Chen, Q., Zhu, X.D., Ling, Z., Wei, S., Jiang, H., Inkpen, D.: Enhanced LSTM for natural language inference. In: ACL (2017)
7. Cong, D., et al.: Hierarchical attention based neural network for explainable recommendation. In: Proceedings of the 2019 on International Conference on Multimedia Retrieval (2019)
8. Dacrema, M.F., Cremonesi, P., Jannach, D.: Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: Proceedings of the 13th ACM Conference on Recommender Systems (2019)
9. Diao, Q., Qiu, M., Wu, C.Y., Smola, A., Jiang, J., Wang, C.: Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In: KDD 2014 (2014)
10. Dong, X., et al.: Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation. In: AAAI (2020)
11. Guo, J., Fan, Y., Ai, Q., Croft, W.: A deep relevance matching model for adhoc retrieval. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (2016)

12. Guo, J., et al.: A deep look into neural ranking models for information retrieval. Inf. Process. Manage. **57**(6), 102067 (2020)
13. He, R., McAuley, J.: Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering. ArXiv abs/1602.01585 (2016)
14. He, X., Chen, T., Kan, M., Chen, X.: Trirank: review-aware explainable recommendation by modeling aspects. In: CIKM 2015 (2015)
15. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web (2017)
16. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2015)
17. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42** (2009)
18. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
19. Ling, G., Lyu, M.R., King, I.: Ratings meet reviews, a combined approach to recommend. In: RecSys 2014 (2014)
20. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. In: RecSys 2013 (2013)
21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. ArXiv abs/1310.4546 (2013)
22. Rao, J., Liu, L., Tay, Y., Yang, W., Shi, P., Lin, J.: Bridging the gap between relevance matching and semantic matching for short text similarity modeling. In: EMNLP/IJCNLP (2019)
23. Ren, Z., Liang, S., Li, P., Wang, S., Rijke, M.: Social collaborative viewpoint regression with explainable recommendations. In: WSDM 2017 (2017)
24. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. arXiv preprint arXiv:1205.2618 (2012)
25. Sachdeva, N., McAuley, J.: How useful are reviews for recommendation? A critical review and potential improvements. arXiv preprint arXiv:2005.12210 (2020)
26. Santos, C.D., Tan, M., Xiang, B., Zhou, B.: Attentive pooling networks. ArXiv abs/1602.03609 (2016)
27. Seo, S., Huang, J., Yang, H., Liu, Y.: Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In: Proceedings of the Eleventh ACM Conference on Recommender Systems (2017)
28. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. ArXiv abs/1803.02155 (2018)
29. Tan, Y., Zhang, M., Liu, Y., Ma, S.: Rating-boosted latent topics: understanding users and items with ratings and reviews. In: IJCAI (2016)
30. Tay, Y., Luu, A.T., Hui, S.: Compare, compress and propagate: Enhancing neural architectures with alignment factorization for natural language inference. In: EMNLP (2018)
31. Tay, Y., Tuan, L.A., Hui, S.C.: Multi-pointer co-attention networks for recommendation. arXiv preprint arXiv:1801.09251 (2018)
32. Vaswani, A., et al.: Attention is all you need. ArXiv abs/1706.03762 (2017)
33. Wu, C., Wu, F., Liu, J., Huang, Y.: Hierarchical user and item representation with three-tier attention for recommendation. In: NAACL-HLT (2019)
34. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling. arXiv preprint arXiv:1706.06613 (2017)

35. Xu, Z., Han, Y., Zhang, Y., Ai, Q.: E-commerce recommendation with weighted expected utility. arXiv preprint arXiv:2008.08302 (2020)
36. Yang, R., Zhang, J., Gao, X., Ji, F., Chen, H.: Simple and effective text matching with richer alignment features. ArXiv abs/1908.00300 (2019)
37. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: HLT-NAACL (2016)
38. Zeng, H., Ai, Q.: A hierarchical self-attentive convolution network for review modeling in recommendation systems. arXiv preprint arXiv:2011.13436 (2020)
39. Zhang, W., Yuan, Q., Han, J., Wang, J.: Collaborative multi-level embedding learning from reviews for rating prediction. In: IJCAI (2016)
40. Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., Ma, S.: Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In: SIGIR (2014)
41. Zheng, L., Noroozi, V., Yu, P.S.: Joint deep modeling of users and items using reviews for recommendation. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (2017)

Utilizing Local Tangent Information for Word Re-embedding

Wenyu Zhao[1,3], Dong Zhou[1](✉), Lin Li[2], and Jinjun Chen[3]

[1] School of Computer Science and Engineering, Hunan University of Science and Technology,
Xiangtan 411201, Hunan, China
wenyuzhao1993@hotmail.com, dongzhou1979@hotmail.com

[2] School of Computer Science and Technology,
Wuhan University of Technology, Wuhan 430070, Hubei, China
cathylilin@whut.edu.cn

[3] Department of Computer Science and Software Engineering, Swinburne University of
Technology, Hawthorn, Melbourne, VIC 3122, Australia
jinjun.chen@gmail.com

Abstract. Word embedding models typically learn dense and fixed-length vectors based on local word collocation patterns in a text corpus. Recent studies have discovered that these models often underestimate similarities between similar words and overestimate similarities between distant words. This leads to word similarity results obtained from word embedding models inconsistent with human judgment. A number of manifold learning-based word re-embedding methods are proposed to address this problem by re-embedding word vectors from the original embedding space to a new embedding space. However, these methods perform a weighted locally linear combination of embeddings of words and their neighbors twice. Besides, the reconstruction weights are easily influenced by the selection of word neighbors and the whole combination process is very time-consuming. In this paper, we introduce a novel word re-embedding method based on local tangent information to re-embed word vectors into a refined new space. Unlike previous approaches, our method re-embeds word vectors by aligning original and new embedding spaces based on the tangent information instead of performing weighted locally linear combination twice. To validate the proposed method, experiments were conducted on two standard evaluation tasks. The experimental results show that our method achieves better performance than state-of-the-art methods for word re-embedding.

Keywords: Word re-embedding · Local tangent information · Manifold learning

1 Introduction

Word embedding models represent words as dense and fixed-length vectors by mapping them from high-dimensional space to low-dimensional space. As the common knowledge, the distance between these dense vectors reflects the semantic relatedness of their

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 740–754, 2021.
https://doi.org/10.1007/978-3-030-72113-8_49

corresponding words. Furthermore, vectors generated by these models contain semantic and syntactic features, which are beneficial to mine the semantic relationships of words. Due to the ability of vector-space representations, word embedding models play an important role in a lot of Information Retrieval (IR) and Natural Language Processing (NLP) tasks, such as question answering [1], ad-hoc retrieval [2] and machine translation [3], part-of-speech tagging [4], named entity recognition [5], text classification [6]. Obviously, the discovery of semantic information is closely linked to the quality of word vectors. The representation quality of word vectors can directly affect the performance of a large amount of IR and NLP tasks as well.

Recently, a variety of word embedding models has been proposed to generate word embeddings, such as BERT [7], C&W [8], Continuous Bag-of-Words (CBOW) [9], Skip-Gram [9], GloVe [10] and other variants [11, 12]. BERT [7] and its variants [13, 14] can effectively produce contextual word embeddings with better support for different IR and NLP tasks. However, the computational cost is very high due to the huge amount of parameters. The refinement of contextual word embeddings will be studied in the future. In comparison with contextual models, static word embedding models are generally simple and efficient with a much lower computational cost. Although these static word embedding models can easily learn word vectors with linear structure data distribution, they fail to estimate similarities between words when the data distribution of words shows strong non-linear characteristics. They may underestimate similarities between nearby words and overestimate similarities between distant words, causing the problem about word similarity results obtained by word embedding models inconsistent with human judgment [15, 16].

As an example given in previous studies [15, 16], an example of the ground truth similarities between words obtained by human experience in a typical semantic similarity task is shown in Fig. 1. Another example of cosine similarity results of the same word pairs obtained by GloVe is shown in Fig. 2. As shown in these two Figures, the similarity result between “physics” and “proton” is more similar than that of “shore” and “woodland” based on human experience in Fig. 1. However, it achieves the opposite result in Fig. 2. The phenomenon fully reflects that similarity results between word pairs obtained by word embedding models may be inconsistent with human judgment.

$$sim("shore","woodland") = 3.08$$
$$< sim("physics","proton") = 8.12$$

Fig. 1. Standard word similarity results judged by human beings

$$sim("shore","woodland") = 0.36$$
$$> sim("physics","proton") = 0.33$$

Fig. 2. Word similarity results obtained by GloVe word embedding models

To address the similarity inconsistency problem, the existing studies show that re-embedding can rectify this problem by using manifold learning-based methods [15, 16]. Several approaches were proposed to re-embed word vectors into a new embedding space by using manifold learning-based methods for this purpose. For example, Locally Linear Embedding (LLE) [15] and Modified Locally Linear Embedding (MLLE) algorithms [16] were proposed to re-embed pre-trained GloVe word vectors into a new embedding space. The above two methods both consider the local geometric information between words and their local neighboring words. They re-embed word vectors based on the weighted locally linear combination of words and their neighbors in both original and refined semantic spaces. Although they achieve good performance on word re-embedding, there exist certain demerits in both methods. On the one hand, the reconstruction weights can be easily affected by various options of word neighbors because these weights are generated by a linear combination of nearby words. On the other hand, these two methods need to perform the weighted locally linear combination twice in both two embedding spaces, which is time-consuming with high computation cost.

Unlike LLE and MLLE methods, in this paper, we introduce a novel word re-embedding method based on Local Tangent Information (denoted as **LTI**) to re-embed word vectors into a refined new space. Our method firstly applies Principal Components Analysis (PCA) on word neighbors to construct a locally linear plane, which can be regarded as an approximation of the tangent information of these local words [17, 18]. Our **LTI** method then re-embeds word vectors by aligning original and refined new embedding space based on the local tangent information (containing different local geometric information). The proposed method can be more effective and efficient by directly aligning two embedding spaces based on local tangent information in comparison with LLE and MLLE methods, which perform combination operation twice. To verify the proposed **LTI** method, we conduct several experiments on standard semantic relatedness and semantic similarity tasks. The experimental results show that our method achieves better performance than the state-of-the-art baseline methods for word re-embedding.

The contributions of our work are summarized as follows:

- We introduce a novel word re-embedding method based on local tangent information. Our method re-embeds word vectors by aligning original and refined semantic spaces based on the tangent information of words, which contains more geometric information and directly captures the relationships between original and refined embedding spaces.
- We are the first to demonstrate that local tangent information can be used to improve the performance of word re-embedding.
- We conduct several experiments to validate our proposed method in this paper. Compared with the state-of-the-art baseline methods of word re-embedding, the results show that our proposed method can achieve better performance by utilizing local tangent information of words and their neighbors.

The rest of our paper is organized as follows: Sect. 2 describes the related work. Our method is presented in Sect. 3. Section 4 shows the details of experimental settings. In Sect. 5, we provide and analyze the experimental results. Finally, Sect. 6 concludes the paper and discusses future research.

2 Related Work

2.1 Count-Based Word Embedding Methods

Count-based word embedding methods only focus on word co-occurrence probability or word counts. Vector space model is the early idea to use vectors to express words [19].

This method constructed a word-document co-occurrence matrix and used it to represent words and documents as vectors by using TF-IDF. However, this method does not consider the true semantic information of words. Latent Semantic Analysis (LSA) [20] can also generate word embeddings by applying Singular Value Decomposition (SVD) to a word-document matrix. Subsequently, Lund and Burgess [21] proposed a Hyperspace Analogue to Language (HAL) model that constructed a word-context word matrix based on a corpus to form vector representations. Dhillon et al. [22] introduced an alternative method leveraging Canonical Correlation Analysis (CCA) between left and right contexts to generate word embeddings. Lebret and Collobert [23] used Hellinger PCA to the word-context matrix to obtain word embeddings. In summary, these methods globally utilize word-context co-occurrence or counts to produce word embeddings based on word-context matrices in a corpus. Though the aforementioned methods are simple and effective, these count-based methods only consider the co-occurrence probability or word counts between words and their context words rather than the real semantic relationships between them.

2.2 Prediction-Based Word Embedding Methods

Prediction-based word embedding methods generate word embeddings by using the contexts of words. In the early time, Hinton proposed a word distributed representation hypothesis [24]. Most of the subsequent methods are inspired by this hypothesis. They represent words as distributional dense, fixed-length and low-dimensional word vectors. Bengio et al. [25] proposed an N-Gram neural network language model and used it to generate word embeddings. In this method, embeddings are a by-product during training a neural network language model (NNLM). Bengio and Senecal [26] improved NNLM by using a Menote Carlo method and hierarchical softmax layer to speed up word embedding generation. Similarly, Mnih and Hinton [27] proposed a slightly modified log-bilinear model to produce word embeddings. As word embeddings are by-products of previous models, Collobert and Weston [28] designed a model solely aimed at generating word embeddings by using unlabeled data. Following these mentioned works, Collobert et al. proposed a unified neural network architecture C&W and a learning algorithm to discover internal representations of words [8]. Mikolov et al. presented two famous model architectures for learning high-quality continuous vector representations for words [9]. One model (CBOW) predicts the current word by utilizing the context of this word. Another model (Skip-gram) predicts the surrounding words based on the current word. Inspired by Skip-gram and CBOW, Qiu et al. proposed two variants of the CBOW model and the Skip-gram model to produce high-quality distributed representations for words by considering both word proximity and ambiguity [11]. Similar to these studies, Pennington et al. [10] proposed a GloVe model that combines the global features of a corpus and the local contextual features of words for generating word representations.

Apart from the static word embedding models described above, several contextual embedding models have been proven to be effective for word embedding generation these days, such as BERT [7] and its variants [13, 14]. Though word embeddings generated from such models can provide good support for different IR and NLP tasks, the computational cost is very high due to the huge amount of parameters. On the contrary, static word embedding models are simpler and more efficient with a much lower computational cost. In this paper, we mainly focus on static word embeddings and leave the study of refining contextual word embeddings as the future work.

2.3 Word Vector Re-embedding Methods

Many studies are focusing on re-embedding word vectors for improving the quality of word vectors. For example, Chaudhary et al. adapted continuous word representations by using morphological and phonological subword representations for low-resourced languages [29]. Kolyvakis et al. utilized a novel entity alignment method called DeepAlignment to refine pre-trained word vectors for generating ontological entity descriptions in the ontology matching task [30]. Seyeditabari et al. incorporated emotional information of words into pre-trained word vectors for generating emotional embeddings, which can capture the emotional contents of words [31]. Utsumi proposed a simple method to re-embed pre-trained word embeddings by using layer-wise relevance propagation [32]. Yu et al. presented an improved word vector model to refine existing pre-trained word vectors by leveraging real-valued sentiment intensity scores provided by sentiment lexicons [33].

However, this paper mainly focuses on studies about word vector re-embedding by re-mapping word vectors from the original embedding space to a new refined embedding space. Mu et al. projected word embeddings by removing the common mean vectors of pre-trained word vectors [34]. Some methods focus on exploring the geometric structure of word embeddings by using manifold-learning based algorithms and they show that reconstruction of word embeddings can capture the underlying manifold of the data [15, 16, 35]. Hasan and Curry utilized word neighbors in the original embedding space to re-embed pre-trained GloVe vectors into a new embedding space based on LLE [15]. The re-embedded word vectors could learn rich semantic information of word embeddings from a new embedding space for addressing the word similarity inconsistency issue. Furthermore, Chu et al. used a Modified Locally Linear Embedding (MLLE) algorithm to refine word representations in the aspect of geometric information of words and their neighbors [16].

Although the aforementioned manifold learning algorithms for word re-embedding have been proven to be effective, these methods need to perform the weighted locally linear combinations twice in both original and refined embedding spaces. Unlike these methods, we approach the problem of word re-embedding by utilizing local tangent information of words. This information can directly capture the relationships between the original and new embedding space instead of relying on local weights. Our method also avoids performing a locally linear combination of nearby words twice.

3 A Novel Word Re-embedding Method

3.1 Overall Framework

The overall framework of our proposed method based on Local Tangent Information (**LTI**) is shown in Fig. 3. There are four main steps in our method. In step (a), we choose

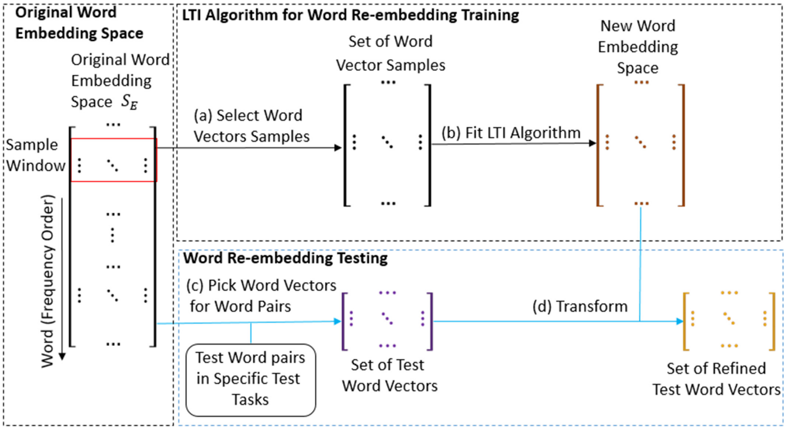

Original Word Embedding Space
Original Word Embedding Space S_E
Sample Window
Word (Frequency Order)
LTI Algorithm for Word Re-embedding Training
Set of Word Vector Samples
(a) Select Word Vectors Samples
(b) Fit LTI Algorithm
New Word Embedding Space
Word Re-embedding Testing
(c) Pick Word Vectors for Word Pairs
Test Word pairs in Specific Test Tasks
Set of Test Word Vectors
(d) Transform
Set of Refined Test Word Vectors

Fig. 3. The framework of our proposed method

a subset of word vector samples from the original embedding space by using a sample window. Word vectors are ordered according to their correspondent word frequencies (frequent word co-occurrences) in this corpus. Note that as in previous studies [15, 16], ordering word vectors and selecting samples instead of using all vectors can avoid a high computational cost. In our work, the original embedding space we used is trained by GloVe, because the pre-trained word vectors from this model can effectively represent words by considering contextual features of words and global features of a corpus in comparison with other static word embedding generation models. In step (b), we train a Local Tangent Information algorithm (**LTI**) on these selected samples in step (a) and this fitted manifold learning algorithm will be used to transform word vectors from original embedding space to a new refined embedding space. In this process, we just transform between two equally-dimensional coordinate systems and keep the dimension of word vectors unchanged. In step (c), we obtain word vectors of test word pairs (test word pairs from specific tasks to validate the effect of word re-embedding) from the original embedding space. In step (d), we re-embed these test word vectors into a new re-embedding space to obtain new vectors by using the fitted **LTI** obtained in step (b).

3.2 Word Re-embedding Based on Local Tangent Information

LLE [15] and MLLE [16] methods aim at addressing the problem that word similarity results of word pairs obtained by word embedding models are inconsistent with that determined by human beings through word re-embedding. These two methods re-embed word vectors by preserving local geometric information of words and their neighbors. However, their research has certain limitations that the reconstruction weights are easily influenced and these two methods need to perform the weighted locally linear combination twice in both two embedding spaces.

Algorithm 1. Word Re-embedding Algorithm based on LTI

```
Input:  original word embedding space $\boldsymbol{S}$, test words $\{w_1, w_2, \cdots, w_m\}$
Output: refined word representations set $\boldsymbol{Z}$ of test words
choose word vector samples $\boldsymbol{X} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_N}]$ from $\boldsymbol{S}$
for each $\boldsymbol{X} \in \boldsymbol{S}$ do
    according to Eq. (1), (5) and (6), fit $\boldsymbol{X}$ to obtain new word embedding space $\boldsymbol{Y}$
end for
for all $w \in \{w_1, w_2, \cdots, w_m\}$ do
    obtain word vectors of $w$ from $\boldsymbol{S}$
    re-embed vector of $w$ to obtain refined vector set $\boldsymbol{Z}$ based on $\boldsymbol{Y}$
end for
return refined word representations set $\boldsymbol{Z}$ of test words
```

Unlike LLE and MLLE methods, our proposed method uses local geometric information different from those of the above two methods. To address the limitations brought by their methods, in this paper, we introduce a novel word re-embedding method based on Local Tangent Information (denoted as **LTI**) to re-embed word vectors into a refined new space. To be specific, a locally linear plane is constructed by leveraging PCA on word neighbors. It is considered as an approximation of the tangent information at each word point [17, 18]. Since both the original and new embedding spaces exist a linear mapping of each word from their spaces to the local tangent information, our method aligns these linear mappings based on local tangent information to re-embed word representations.

As we mentioned in the last subsection, word vector samples are firstly chosen from pre-trained GloVe word vector corpus (original embedding space $\boldsymbol{S}$) through a simple window and Local Tangent Information (**LTI**) is trained on these samples. The set of selected samples is defined as a word vector set $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N]$, where $\boldsymbol{X} \in R^{d \times N}$, N is the number of words and d represents the dimension of word vectors. In our proposed method, for each word vector $\boldsymbol{x_i}$, $(i = 1, 2, \cdots N)$, we find its k nearest neighborhoods (including $\boldsymbol{x_i}$ itself) and denote the adjacent neighborhood set as $\boldsymbol{X_i} = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \cdots, \boldsymbol{x_{ik}}]$. Subsequently, for each word vector $\boldsymbol{x_i}$, we apply PCA to each neighborhood set $\boldsymbol{X_i}$ to approximate the local tangent information of the word corresponding to a word vector $\boldsymbol{x_i}$ for preserving the local structure of the neighborhood set $\boldsymbol{X_i}$ of $\boldsymbol{x_i}$. The objective function is

$$\arg\min_{\boldsymbol{Q}_i,\theta_i}\sum_{j=1}^{k}\left|\left|\left(\boldsymbol{x_{ij}}-\boldsymbol{x}\right)-\boldsymbol{Q}\theta_{ij}\right|\right|^2=\arg\min_{\boldsymbol{Q_i},\boldsymbol{\Omega_i}}||\boldsymbol{X_i}\boldsymbol{H_k}-\mathbf{Q}\boldsymbol{\Omega_i}||^2 \tag{1}$$

where $\boldsymbol{H_k} = \boldsymbol{I} - \frac{\boldsymbol{ee}^T}{k}$ is centralization matrix, $\boldsymbol{I}$ is an identity matrix, $\boldsymbol{e}$ means the vector of all 1's, $\mathbf{Q}$ is an orthonormal basis matrix of the tangent information, $\boldsymbol{\Omega_i} = [\theta_{i1}, \theta_{i2}, \cdots, \theta_{ik}]$ represents a local linear approximation of $\boldsymbol{X_i}$, i.e. θ_{ij} is the tangent coordinate corresponding to the orthonormal basis matrix $\boldsymbol{Q_i}$. Apparently, the optimal $\boldsymbol{x}$ is the mean value of all neighborhood words vectors $\boldsymbol{x_{ij}}$, $(j = 1, 2, \cdots k)$ of the sample point $\boldsymbol{x_i}$, $(i = 1, 2, \cdots N)$. The optimal $\mathbf{Q}$ is given by $\mathbf{Q}_i$ and it is made up of t left singular vectors of $\boldsymbol{X_i H_k}$ corresponding to its t largest singular values (t is equal to d, as the embedding dimension is the same in both two embedding spaces.) The tangent coordinates $\boldsymbol{\Omega_i}$ can be computed as

$$\boldsymbol{\Omega_i} = \mathbf{Q}_i^T \boldsymbol{X_i H_k} \tag{2}$$

After obtaining the local tangent coordinates, we have to construct the global coordinates in a new embedding space. The purpose of the global arrangement of local tangent information is to find a group of new space coordinates $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N]$, which are called global coordinates in a new embedding space. Therefore, we assume that there is a projection matrix, which re-embeds tangent coordinates $\boldsymbol{\Omega_i}$ to new space coordinates $\boldsymbol{Y_i} = \{\boldsymbol{y}_{i1}, \boldsymbol{y}_{i2}, \cdots, \boldsymbol{y}_{iN}\}$, then we have

$$\boldsymbol{Y_i H_k} = \boldsymbol{L_i \Omega_i} + \boldsymbol{E_i} \tag{3}$$

where $\boldsymbol{L_i}$ is the projection matrix mapping $\boldsymbol{\Omega_i}$ to $\boldsymbol{Y_i}$ and $\boldsymbol{E_i}$ is the local reconstruction error term. To preserve as much of the local geometry in a new embedding space as possible, we intend to find $\boldsymbol{Y_i}$ and $\boldsymbol{L_i}$ by minimizing the reconstruction error $\boldsymbol{E_i}$

$$\arg\min_{\boldsymbol{Y}} \sum_{i=1}^{N} ||\boldsymbol{E_i}||^2 = \arg\min_{\boldsymbol{Y}} \sum_{i=1}^{N} ||\boldsymbol{Y_i H_k} - \boldsymbol{L_i \Omega_i}||^2 \tag{4}$$

Obviously, the mapping error is minimal when $\boldsymbol{L_i} = \boldsymbol{Y_i H_k \Omega_i^+}$, where $\boldsymbol{\Omega_i^+}$ is Moore-Penrose generalized inverse of $\boldsymbol{\Omega_i}$. Let refined word vector set $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N]$ be the d dimensional global coordinates of all words in $\boldsymbol{X}$ ($\boldsymbol{Y}$ also be refined new embedding space) and ϕ_i be the 0-1 selection matrix such that $\boldsymbol{Y}\phi_i = \boldsymbol{Y_i}$. The optimal $\boldsymbol{Y}$ can be achieved by minimizing the overall reconstruction error of all neighborhoods and the Formula (4) can be rewritten as:

$$\begin{aligned} \arg\min_{\boldsymbol{Y}} \sum_{i=1}^{N} ||\boldsymbol{E_i}||^2 &= \arg\min_{\boldsymbol{Y}} \sum_{i=1}^{N} ||\boldsymbol{Y_i \phi_i W_i}||^2 \\ &= mintrace\left(\boldsymbol{Y\phi W W^T \phi^T Y^T}\right) \\ &= \min trace\left(\boldsymbol{YBY^T}\right) \end{aligned} \tag{5}$$

where $\phi = [\phi_1, \phi_2, \cdots, \phi_N]$, $\boldsymbol{W} = diag(W_1, W_2, \cdots, W_N)$ with $\boldsymbol{W_i} = \boldsymbol{H_k}\left(\boldsymbol{I} - \boldsymbol{\Omega_i^+ \Omega_i}\right)$ and $\boldsymbol{B} = \boldsymbol{\phi W W^T \phi^T}$. In order to uniquely obtain $\boldsymbol{Y}$, we will impose the constraint $\boldsymbol{YY^T} = \boldsymbol{I}$. The refined new word vector set $\boldsymbol{Y}$ is composed of the t eigenvectors of the matrix $\boldsymbol{B}$, and these eigenvectors correspond to the 2nd to $(t+1)th$ smallest eigenvalues of $\boldsymbol{B}$. Then the eigenvector matrix picked from $\boldsymbol{B}$ is $[\boldsymbol{u}_2, \cdots, \boldsymbol{u}_{t+1}]$, where $\boldsymbol{u_i}$ is an eigenvector of $\boldsymbol{B}$. Thus, d dimensional refined new embedding set $\boldsymbol{Y}$ should be:

$$\boldsymbol{Y} = [\boldsymbol{u}_2, \cdots, \boldsymbol{u}_{t+1}] \tag{6}$$

In our work, we firstly use word vectors samples from the original embedding space to train the **LTI** algorithm by Eq. (1), Eq. (5) and Eq. (6) to obtain a new embedding space $\boldsymbol{Y}$. Then we can obtain the refined new embedding set of test word vectors in specific tasks by using the new embedding space $\boldsymbol{Y}$. The overall procedure of our Word Re-embedding Algorithm based on **LTI** is described in Algorithm 1.

4 Experimental Setup

4.1 Data Description

As we mentioned before, we use the original word vectors trained by GloVe [10]. Moreover, we use two sets of GloVe word vectors[1]. One is trained from Wikipedia 2014+Gigaword 5 (consists of 6 Billion tokens, 400,000 vocabularies, word vectors with 50, 100, 200, and 300 dimensions, denoted as 6B50/100/200/300d). Another set is trained from Common Crawl (consists of 42 Billion tokens, 1.9 Million vocabularies, word vectors with 300 dimensions, denoted as 42B300d). To demonstrate the effectiveness of our proposed method, we conduct experiments on semantic relatedness and semantic similarity tasks. The semantic relatedness task focuses on the degree of semantic relatedness between words. It contains three datasets, including MEN dataset (3000 word pairs) [36], WordRel (WordRel) dataset (252 word pairs) [37], MTurk (MTurk) dataset (287 word pairs) [38]. The semantic similarity task pays attention to the degree of semantic similarity between words. It includes four datasets, which are RG65 (RG) dataset (65 word pairs) [39], WordSim-353 (WS353) dataset (353 word pairs) [40], SimLex-999 (SimLex) dataset (999 word pairs) [41], and WordSim-203 (WS203) dataset (203 word pairs) [42] respectively.

4.2 Baselines

We validate our proposed method for word re-embedding by comparing it with the following representative baseline methods.

GloVe. It is the original GloVe method [10]. This distributed word representation method is general and quite effective. The word vectors trained by this method consider local features of contextual words and global features of a corpus.

LLE. Hasan and Curry utilized local word neighbors to re-embed pre-trained word vectors (also trained by GloVe) based on the LLE manifold learning algorithm [15].

RoM. Mu et al. removed the common mean vectors of the pre-trained word vectors and the top principal components of all words for post-processing word vectors [34].

MLLE. Similar to [15], Chu et al. used the MLLE manifold learning algorithm to re-embed word vectors trained by GloVe [16].

LTI. The method proposed in the current paper. We use a manifold learning method that utilizing local tangent information of words and their neighbors to re-embed word vectors by aligning the original and new embedding space based on the tangent information of words.

4.3 Evaluation Metrics

To evaluate the performance of our proposed method and baseline methods, we adopt Spearman's method to compute the Spearman Rank Correlation coefficient between word similarity scores (similarity scores of word pairs obtained from word re-embedding

[1] http://nlp.stanford.edu/projects/glove.

methods) with human judgments (original similarity scores of word pairs in each dataset). The Spearman Rank Correlation is defined as:

$$\cos(u_1, u_2) = \frac{u_1 \cdot u_2}{\|u_1\| \cdot \|u_2\|} \tag{7}$$

$$r = p_{x,y} = \frac{cov(x, y)}{\sigma_x \cdot \sigma_y} \tag{8}$$

Equation (7) is used to calculate the similarity results of each pair of words in specific tasks, where u_1 and u_2 represent two word vectors. Equation (8) represents the Spearman Rank Correlation coefficient between word similarity scores and human ratings, $cov(x, y)$ represents the covariance between the score ranking list x and y, which denote the score list of word similarity scores obtained by word re-embedding methods and the score list of human judgments respectively, σ_x and σ_y represent the corresponding standard deviations of these two score lists. The more consistent similarities of word pairs obtained by word re-embedding methods with human judgments, the higher the Spearman score is.

4.4 Implementation Details

Firstly, we select word vector samples from a pre-trained word vector corpus by using a sample window and use the **LTI** algorithm to train these samples. Then for each specific task, we obtain word vectors of test word pairs and transform these word vectors into a new embedding space by using the fitted **LTI** algorithm. Finally, we compute cosine similarity scores of word pairs in each specific task and compute the Spearman scores. In our method, the range value of number of neighbors chosen was set as [300, 1500] and the step is 100. The range value of the training sample window size was set as [300, 2000] and the step is 50. Previous experiments show that the best sample size should be as close as possible to the number of neighbors because a wider range has no significant difference in results and has high time and computation cost.

5 Results and Discussion

5.1 Performance on Word Vectors with Different Embedding Dimensions

In order to evaluate the performance of our proposed method and other word re-embedding methods on word vectors with different embedding dimensions, we conduct experiments on WS353 and RG dataset as in previous studies [15, 16]. The experimental results are shown in Table 1. As shown in this table, LLE, MLLE and our proposed **LTI** method perform better than GloVe in most cases. This demonstrates that using a manifold-learning based algorithm is beneficial to generate word embeddings with high quality. Furthermore, we can observe that our proposed method achieves better performance than LLE and MLLE methods in 5 out of 10 experimental runs. In terms of dataset, the MLLE method achieves good performance on RG dataset than that of WS353 dataset. We can observe that our proposed method achieves the best result in 4 out of 5 experimental runs on RG dataset. However, this proposed method only obtains the highest scores in 2 out of 5 experimental runs on WS353 dataset. This is probably due to some noises existing in word vectors in the due dataset. Another reason is that the distance of words and their neighbors in RG dataset may be closer than that of words and their neighbors in WS353 dataset, so the geometric information of RG dataset is more beneficial to the manifold-learning based methods for word re-embedding than that of WS353 dataset.

Table 1. Spearman correlations scores of various methods on two evaluation datasets. Bold values represent that our method achieves the best results than baseline methods. Note that baseline results are taken from [16].

Space	Task	GloVe	LLE	MLLE	LTI
6B50d	WS353	61.2	56.6	63.2	61.2
6B100d	WS353	64.5	64.3	64.6	**66.4**
6B200d	WS353	68.5	69.7	67.0	68.2
6B300d	WS353	65.8	70.3	67.9	69.3
42B300d	WS353	75.2	78.4	78.6	**78.6**
Space	Task	GloVe	LLE	MLLE	LTI
6B50d	RG	60.2	53.0	64.4	62.6
6B100d	RG	65.3	67.3	68.8	**73.3**
6B200d	RG	75.5	76.0	79.4	**81.5**
6B300d	RG	75.5	80.5	81.1	**83.1**
42B300d	RG	80.0	83.4	83.5	**86.5**

In addition, with regard to embedding dimensions, our proposed method outperforms the MLLE method on both datasets with embedding dimensions more than 50. However, when RG and WS353 datasets containing 6B tokens and the embedding dimension is 50, the MLLE shows better performance than our method. The reason may be that multiple weights are more suitable to describe the relationships between words and their neighbors than the tangent information when the embedding dimension is very low. However, when RG dataset contains 6B tokens and the embedding dimension increases, our proposed method shows better performance than all baseline methods. It is obvious that the higher dimension of word vectors is, the better performance of our proposed method can get because word vectors with high dimensions can capture more semantic information.

Moreover, we can observe that when the size of datasets increases (from 6B to 42B) and the embedding dimension reaches 300, our proposed method can greatly improve word similarity performance on both datasets. This indicates that the larger training size and larger dimension are beneficial for word re-embedding. Then, we conduct experiments on seven datasets with a size of 42B and use the embedding dimension of 300 to further validate the effectiveness of our proposed **LTI** method.

5.2 Performance on Two Evaluation Tasks

In addition to these experiments, more experiments are conducted on seven datasets to further validate the performance of our proposed **LTI** method. Table 2 displays the results of all methods in two evaluation tasks. From this table, an observation is that almost all word re-embedding methods (**LTI**, MLLE, LLE and RoM) perform better than Glove. These results are in-line with previous findings so that these two tasks are quite suitable to evaluate the word re-embedding methods. This further suggests that word re-embedding can improve the performance of word representations.

Table 2. Spearman correlations between scores predicted by our method and scores obtained from human judgment on two evaluation tasks. Bold values represent that our method achieves the best results than baseline methods. Note that baseline results are taken from [16].

	Semantic similarity task				Semantic relatedness task		
Method	RG	WS353	SimLex	WS203	MTurk	WordRel	MEN
GloVe	76.90	71.25	40.83	80.15	69.29	64.43	80.49
LLE	74.71	77.14	48.14	81.40	71.92	72.90	83.37
RoM	74.36	76.79	44.97	–	70.85	–	81.78
MLLE	77.19	78.40	49.40	82.32	72.78	73.69	84.19
LTI	**86.48**	**78.58**	**50.46**	81.92	**73.15**	**74.65**	83.50

We notice that our proposed **LTI** method is the best performing method on 5 out of 7 datasets in comparison with the MLLE method. This is because the MLLE method may be strongly influenced by the local weights. Our method aligns the original and refined semantic space based on the local tangent information rather than the multiple local weights. Furthermore, our method does not calculate the weighted combination of embedding of words and their neighbors twice, which is more efficient. Our **LTI** method performs slightly worse than the MLLE method on WS203 and MEN datasets. This is likely caused by the better effect of the local weights in the MLLE method. However, the differences are quite small (0.49% and 0.83%). In summary, our method can achieve better performance than all other baseline methods and it is more computationally efficient than all previously proposed word re-embedding methods that are included in the comparison.

6 Conclusion

Word re-embedding can address the problem that the similarity scores of word pairs obtained by word embedding models are inconsistent with human ratings. In this paper, we introduce a novel word re-embedding method based on Local Tangent Information (**LTI**) to re-embed word vectors. Our **LTI** method tries to re-embed vectors by aligning the original and new embedding spaces based on local tangent information.

We conduct several experiments on semantic relatedness and semantic similarity tasks. The results demonstrate that our proposed method achieves better performance than the existing word re-embedding methods. In future work, our method can be advanced in two directions. On the one hand, we will try to discover the key factors that influence the effectiveness of the word re-embedding process. On the other hand, we will explore the contextual word embedding refinement by using manifold learning methods.

Acknowledgements. We would like to thank anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Natural Science Foundation of China under Project No. 61876062 and General Key Laboratory for Complex System Simulation under Project No. XM2020XT1004.

References

1. Esposito, M., Damiano, E., Minutolo, A., De Pietro, G., Fujita, H.: Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering. Inf. Sci. **514**, 88–105 (2020)
2. Bagheri, E., Ensan, F., Al-Obeidat, F.: Neural word and entity embeddings for ad hoc retrieval. Inf. Process. Manag. **54**(4), 657–673 (2018)
3. Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R. M., Makhoul, J.: Fast and robust neural network joint models for statistical machine translation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL), vol. 1, pp. 1370–1380 (2014)
4. Collobert, R.: Deep learning for efficient discriminative parsing. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 224–232 (2011)
5. Turian, J., Lev, R., Yoshua, B.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 384–394. Association for Computational Linguistics (ACL) (2010)
6. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), vol. 1631, pp. 1631–1642. Citeseer (2013b)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Minneapolis, MN, USA, pp. 4171–4186 (2018)
8. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**(1), 2493–2537 (2011)
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of Workshop at International Conference on Learning Representations (ICLR), Scottsdale, Arizona, pp. 3111–3119 (2013a)
10. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
11. Qiu, L., Cao, Y., Nie, Z., Yu, Y., Rui, Y.: Learning word representation considering proximity and ambiguity. In: Proceedings of Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 1572–1578 (2014)

12. Peng, X., Zhou, D.: A framework for learning cross-lingual word embedding with topics. In: Wang, X., Zhang, R., Lee, Y.-K., Sun, L., Moon, Y.-S. (eds.) APWeb-WAIM 2020. LNCS, vol. 12318, pp. 285–293. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-60290-1_22
13. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: a lite BERT for self-supervised learning of language representations. In: Proceedings of the 8th International Conference on Learning Representations (ICLR), Addis Ababa, Ethiopia, pp. 1–17 (2020)
14. Liu, W., Zhou, P., Wang, Z., Zhao, Z., Deng, H., Ju, Q.: FastBERT: a self-distilling BERT with adaptive inference time. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 6035–6044 (2020)
15. Hasan, S., Curry, E.: Word re-embedding via manifold dimensionality retention. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 321–326 (2017)
16. Chu, Y., Lin, H., Yang, L., Diao, Y., Zhang, S., Fan, X.: Refining word representations by manifold learning. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), pp. 5394–5400 (2019)
17. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimension reduction via local tangent space alignment. J. Shanghai Univ. **8**(4), 406–424 (2002)
18. Zhang, Z., Zha, H.: Nonlinear dimension reduction via local tangent space alignment. In: Liu, J., Cheung, Y., Yin, H. (eds.) IDEAL 2003. LNCS, vol. 2690, pp. 477–481. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45080-1_66
19. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM **18**(11), 613–620 (1975)
20. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. J. Am. Soc. Inf. Sci. **41**(6), 391–407 (1990)
21. Lund, K., Burgess, C.: Producing high-dimensional semantic spaces from lexical co-occurrence. Behav. Res. Methods Instrum. Comput. **28**(2), 203–208 (1996)
22. Dhillon, P., Foster, D.P., Ungar, L.H.: Multi-view learning of word embeddings via CCA. Advances in Neural Information Processing Systems, pp. 199–207 (2011)
23. Lebret, R., Collobert, R.: Word embeddings through hellinger PCA. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. Idiap, pp. 482–490 (2013)
24. Hinton, G.E.: Learning distributed representations of concepts. In: Proceedings of the Eighth Annual Conference of the Cognitive Science Society, vol. 1, pp. 1–12 (1986)
25. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**(1), 1137–1155 (2003)
26. Bengio, Y., Senécal, J.S.: Quick training of probabilistic neural nets by importance sampling. In: Proceedings of the Conference on Artificial Intelligence and Statistics (AISTATS), pp. 1–9 (2003)
27. Mnih, A., Hinton, G.: Three new graphical models for statistical language modelling. In: Proceedings of the 24th International Conference on Machine Learning, pp. 641–648 (2007)
28. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning, pp. 160–167 (2008)
29. Chaudhary, A., Zhou, C., Levin, L., Neubig, G., Mortensen, D.R., Carbonell, J.G.: Adapting word embeddings to new languages with morphological and phonological subword representations. arXiv preprint arXiv:1808.09500 (2018)
30. Kolyvakis, P., Kalousis, A., Kiritsis, D.: Deepalignment: Unsupervised ontology matching with refined word vectors. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 787–798 (2018)

31. Seyeditabari, A., Tabari, N., Gholizadeh, S., Zadrozny, W.: Emotional embeddings: refining word embeddings to capture emotional content of words. arXiv preprint arXiv:1906.00112 (2019)
32. Utsumi. A.: Refining pretrained word embeddings using layer-wise relevance propagation. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 4840–4846 (2018)
33. Yu, L.C., Wang, J., Lai, K.R., Zhang, X.: Refining word embeddings using intensity scores for sentiment analysis. IEEE Trans. Audio Speech Lang. Process. **26**(3), 671–681 (2017)
34. Mu, J., Bhat, S., Viswanath, P.: All-but-the-top: simple and effective postprocessing for word representations. In: Proceedings of Poster at 6th International Conference on Learning Representations (ICLR), 1–25 (2018)
35. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
36. Bruni, E., Tran, N.K., Baroni, M.: Multimodal distributional semantics. J. Artif. Intell. Res. **49**(1), 1–47 (2014)
37. Agirre, E., Alfonseca, E., Hall, K.B., Kravalova, J., Pasca, M., Soroa, A.: A study on similarity and relatedness using distributional and wordnet-based approaches. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL), pp. 19–27 (2009)
38. Kira, R., Agichtein, E., Gabrilovich, E.: A word at a time: computing word relatedness using temporal semantic analysis. In: Proceedings of the 20th International Conference on World Wide Web (WWW), pp. 337–346 (2011)
39. Rubenstein, H., Goodenough, J.B.: Contextual correlates of synonymy. Commun. ACM **8**(10), 627–633 (1965)
40. Finkelstein, L., Gabrilovich, E., Matias, Y.: Placing search in context: the concept revisited. In: Proceedings of the 10th International Conference on World Wide Web (WWW), pp. 406–414 (2001)
41. Hill, F., Korhonen, A.: Simlex-999: evaluating semantic models with (genuine) similarity estimation. Comput. Linguist. **41**(4), 665–695 (2015)
42. Gerz, D., Vulić, I., Hill, F., Reichart, R., Korhonen, A.: Simverb-3500: a large-scale evaluation set of verb similarity. arXiv preprint arXiv:1608.00869 (2016)

Content Selection Network for Document-Grounded Retrieval-Based Chatbots

Yutao Zhu[1](✉), Jian-Yun Nie[1], Kun Zhou[2], Pan Du[1], and Zhicheng Dou[3]

[1] Université de Montréal, Québec, Canada
yutao.zhu@umontreal.ca, {nie,pandu}@iro.umontreal.ca
[2] School of Information, Renmin University of China, Beijing, China
francis_kun_zhou@163.com
[3] Gaoling School of Artificial Intelligence,
Renmin University of China, Beijing, China
dou@ruc.edu.cn

Abstract. Grounding human-machine conversation in a document is an effective way to improve the performance of retrieval-based chatbots. However, only a part of the document content may be relevant to help select the appropriate response at a round. It is thus crucial to select the part of document content relevant to the current conversation context. In this paper, we propose a document content selection network (CSN) to perform explicit selection of relevant document contents, and filter out the irrelevant parts. We show in experiments on two public document-grounded conversation datasets that CSN can effectively help select the relevant document contents to the conversation context, and it produces better results than the state-of-the-art approaches. Our code and datasets are available at https://github.com/DaoD/CSN.

Keywords: Content selection · Document-grounded dialogue · Retrieval-based chatbots

1 Introduction

Retrieval-based chatbots such as Microsoft XiaoIce [19] and Amazon Alexa [16] are widely used in real-world applications. Given a user input, an upstream retrieval system can provide a set of response candidates, and the retrieval-based chatbot should choose the appropriate one. This mainly relies on a matching score between the context and each candidate response. It has been found that the conversation context alone is insufficient in many cases for response selection [22,28]. In fact, human conversations are usually also grounded in external knowledge or documents: our responses are strongly related to our knowledge or information contained in the documents at hand. On Reddit, for example, people usually discuss about a document posted at the beginning of a thread, which provides the background topics and basic facts for the following conversations.

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, pp. 755–769, 2021.
https://doi.org/10.1007/978-3-030-72113-8_50

On Twitter, people may also exchange opinions related to a news article. In these cases, in addition to the conversation context, the document or news article also provides useful background information to guide response selection. A conversation that does not take into account the background information may lead to off-topic responses. This paper deals with the problem of document-grounded conversation - conversation based on a given document [1,4,15,28,29].

The task of document-grounded response selection is formulated as selecting a good response from a candidate pool that is consistent with the context and relevant to the document. Several existing studies have shown that leveraging the background document can significantly improve response selection [4,28,29]. Generally, the common strategy is selecting the response based on a combination of context-response matching and document-response matching. The latter can boost the responses that are related to the document content. However, a good response does not need to be related to the whole content of the document, but to a small part of it. The selection of the relevant part of the document is crucial.

The problem can be illustrated by an example from CMUDoG [30] in Fig. 1. In this dataset, a movie-related wiki article is used as the grounding document. We can see that the conversation is highly related to the document. R1, R2, and R3 are three candidate responses for U6, and R3 is the desired response. The wrong response R1 could be highly scored because it shares several key words with the document (*i.e.*, document-response matching score is high). However, R1 is not an appropriate response in the current context, which asks about

Document			
Name	The inception	Year	2009
Director	Christopher Nolan	Genre	Scientific
Cast	Leonardo DiCaprio as Dom Cobb, a professional thief who specializes in conning secrets from his victims by infiltrating their dreams. Tom Hardy as Eames, a sharp-tongued associate of Cobb. ···		
Critical Resp.	Response DiCaprio, who has never been better as the tortured hero, draws you in with a love story that will appeal even to non-scifi fans. The movie is a metaphor for the power of delusional hype for itself.		
Intro.	··· Dominick Cobb and Arthur are extractors, who perform corporate espionage using an experimental military technology to infiltrate the subconscious of their targets and extract valuable information through a shared dream world. Their latest target, Japanese businessman Saito, reveals that he arranged the mission himself to test Cobb for a seemingly impossible job: planting an idea in a person's subconscious, or inception.		
Rating	Rotten Tomatoes: 86% and average: 8.1/10; IMDB: 8.8/10		
Conversation			
U1	Have you seen the **inception**?		
U2	No, I have not but have heard of it. What is it about?		
U3	It's about **extractors that perform experiments using military technology on people to retrieve info about their targets.**		
U4	Sounds interesting. Do you know which actors are in it?		
U5	I haven't watched it either or seen a preview. But it's scifi so it might be good. Ugh **Leonardo DiCaprio is the main character**. He plays as Don Cobb.		
U6	I'm not a big scifi fan but there are a few movies I still enjoy in that genre. Is it a long movie?		
R1	Many long shots are used to show the beautiful scene. Besides, it is really a good **story that will appeal even to non-scifi fans**!		
R2	Well, not really. The **extractors** come out with the **military technology** and **infiltrate the subconscious**.		
R3 ✓	Doesn't say how long it is. The **Rotten Tomatoes** score is **86%.**		

Fig. 1. An example in CMUDoG dataset. The words in color correspond to those in the document. R3 is the ground-truth response.

the length of the movie. This example shows that a correct response is well grounded in the document not because it corresponds to the document content, but because it corresponds to the part relevant to the conversation context. Therefore, a first challenge is to select the part of the document content relevant to the current conversation context. R2 looks like a proper response to U6, yet it conveys similar information as U3, which makes the dialogue less informative. This response could be selected if we use the whole conversation history as conversation context - the response could have a high context-response matching score. In fact, the current context in this example is about the length of the movie. The previous utterances in the history are less relevant. This case illustrates the need to well calibrate and model the current conversation context.

The two key problems illustrated by the above example (R1 and R2) are not well addressed in previous studies: (1) They usually perform a soft selection of document content by assigning attention weights to them [4,29]. Even though the less relevant parts could be assigned lower weights, the cumulative weight of many irrelevant parts could be large, so that they collectively influence the response selection in a wrong direction. We believe that a key missing element is a proper (hard) selection of the document content that fits the current conversation context, instead of a (soft) weighting. The hard selection of document content is motivated by the following observation: although the whole conversation can cover many aspects described in the grounding document, each of the step is related to only a small part of the document content. For example, in our conversation about a movie, we could discuss about an actor in one step. The selection of such a small part of the content is crucial. This observation advocates a hard selection rather than a soft weighting used in the previous studies. (2) The existing studies usually use the entire context to determine the weights of parts (sentences) of the document content. This strategy fails to distinguish the current conversation context from the ones in the history. As a result, a past round of conversation could be mistaken as the current one, leading to a redundant response as illustrated by the R2 example.

In this paper, we propose a **Content Selection Network** (CSN) to tackle these problems. **First**, we use a modified *gate mechanism* to implement the document content selection according to the conversation context, before using it to match with the response candidate. The content relevant to the current conversation step will be assigned a higher weight and pass the selection gate, while the irrelevant parts will be blocked. We use the gate mechanism to select sentences or words. **Second**, as the topic usually evolves during the conversation, we determine the current conversation context by focusing on the most recent utterances, rather than on the whole conversation history. To this end, we design a decay mechanism for the history to force the model focusing more on the current dialogue topic. The selected document contents and the conversation context are finally combined to select the candidate response.

The main contributions of this paper are: (1) We propose a content selection network to explicitly select the relevant sentences/words from the document to complement the conversation context. Our experiments show that this is a much more effective way to leverage the grounding document than a soft weighting. (2) We show that document-grounded conversation should focus on the topics in the recent state rather than using the whole conversation context. On two public datasets for document-grounded conversation, our method outperforms the existing state-of-the-art approaches significantly.

2 Related Work

Retrieval-Based Chatbots. Existing methods for open-domain dialogue can be categorized into two groups: retrieval-based and generation-based. Generation-based methods are mainly based on the sequence-to-sequence (Seq2seq) architecture with attention mechanism and aim at generating a new response for conversation context [2,8,17,18,26]. On the other hand, retrieval-based methods try to find the most reasonable response from a large repository of conversational data [10,21,25,27]. We focus on retrieval-based methods in this paper. Early studies use single-turn response selection where the context is a single message [6,7], while recent work considers all previous utterances as context for multi-turn response selection [21,25,27,31]. In our work, we also consider the whole conversation history (but with decaying weights).

Document-Grounded Conversation. Multiple studies have shown that being grounded in knowledge or document can effectively enhance human-machine conversation [3,11,28,29]. For example, a Seq2seq model is first applied to generate responses based on both conversation history and external knowledge [3]. An approach using a dually interactive matching network has been proposed, in which context-response matching and document-response matching are performed separately using a shared structure [4]. This model achieved state-of-the-art performance on persona-related conversation [28]. Recently, Zhao et al. [29] proposed a document-grounded matching network that lets the document and the context to attend to each other so as to generate better representations for response selection. Through the attention mechanism, different parts (sentences) of the document are assigned different weights and will participate in response selection to different extents. However, even though one may expect the noise contents (for the current step) be assigned with lower weights, they can still participate in response selection.

Our work differs from the existing studies in that we explicitly model the document content selection process and prevent the irrelevant contents from participating in response selection. In addition, we also define the current conversation context by focusing more on recent utterances in the history rather than taking the whole history indistinctly. These ideas will bring significant improvements compared to the existing methods.

3 Content Selection Network

3.1 Problem Formalization

Suppose that we have a dataset $\mathcal{D}$, in which each sample is represented as (c, d, r, y), where $c = \{u_1, \ldots, u_n\}$ represents a conversation context with $\{u_i\}_{i=1}^{n}$

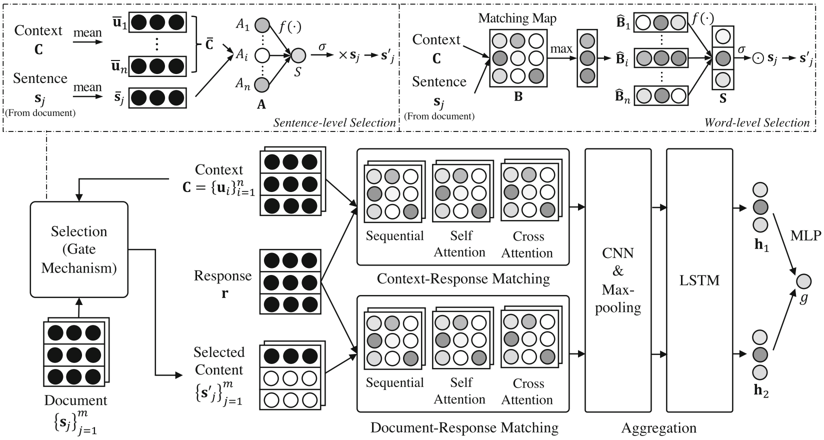


Fig. 2. The structure of CSN.

as utterances; $d = \{s_1, \ldots, s_m\}$ represents a document with $\{s_i\}_{i=1}^m$ as sentences; r is a response candidate; $y \in \{0, 1\}$ is a binary label, indicating whether r is a proper response. Our goal is to learn a matching model g from $\mathcal{D}$, such that for a new context-document-response triplet (c, d, r), $g(c, d, r)$ measures the degree of suitability of a response r to the given context c and the document d.

3.2 Model Overview

We propose a content selection network (CSN) to model $g(\cdot, \cdot, \cdot)$, which is shown in Fig. 2. Different from the previous work [4,21,25] which uses the whole document contents, we propose a selection module with a gate mechanism to select the relevant parts of document content based on the context. Then, the context-response matching and the document-response matching are modeled based on the sequential, self-attention, and cross-attention representations. Finally, CNNs and RNNs are applied to extract, distill, and aggregate the matching features, based on which the response matching score is calculated.

3.3 Representation

Consider the i-th utterance $u_i = (w_1^u, \cdots, w_L^u)$ in the context, the j-th sentence $s_j = (w_1^s, \cdots, w_L^s)$ in the document, and the response $r = (w_1^r, \cdots, w_L^r)$, where L is the number of words[1]. CSN first uses a pre-trained embedding table to map each word w to a d_e-dimension embedding $\mathbf{e}$, *i.e.*, $w \Rightarrow \mathbf{e}$. Thus the utterance u_i, the sentence s_j, and the response r are represented by matrices $\mathbf{E}^{u_i} = (\mathbf{e}_1^{u_i}, \cdots, \mathbf{e}_L^{u_i})$, $\mathbf{E}^{s_j} = (\mathbf{e}_1^{s_j}, \cdots, \mathbf{e}_L^{s_j})$, and $\mathbf{E}^r = (\mathbf{e}_1^r, \cdots, \mathbf{e}_L^r)$, respectively. Then, CSN encodes the utterances, sentences and responses by bi-directional

[1] To simplify the notation, we assume their lengths are the same.

long short-term memories (BiLSTM) [5] to obtain their sequential representations: $\mathbf{u}_i = \text{BiLSTM}(\mathbf{E}^{u_i})$, $\mathbf{s}_j = \text{BiLSTM}(\mathbf{E}^{s_j})$, $\mathbf{r} = \text{BiLSTM}(\mathbf{E}^{r})$. Note that the parameters of these BiLSTMs are shared in our implementation. The whole context is thus represented as $\mathbf{C} = [\mathbf{u}_1, \cdots, \mathbf{u}_n]$. With the BiLSTM, the sequential relationship and dependency among words in both directions are expected to be encoded into hidden vectors.

3.4 Content Selection

In document-grounded conversation, the document usually contains a large amount of diverse information, but only a part of it is related to the current step of the conversation. To select the relevant part of document contents, we propose a content selection phase by a *gate mechanism*, which is based on the relevance between the document and the context. We design the gate mechanism at two different levels, *i.e.*, sentence-level and word-level, to capture relevant information at different granularities. If the sentences/words in the document are irrelevant to the current conversation, they will be filtered out. This is an important difference from the traditional gating mechanism, in which elements are assigned different attention weights, but no element is filtered out. We use the conversation context to control the gate, which contains several previous turns of conversation. Along the turns, the conversation topic gradually changes. The most important topic is that of the most recent turn, while more distant turns are less important. To reflect this fact, we design a decay mechanism on the history to assign a higher importance to the recent context than to the more distant ones. The selection process is automatically trained with the whole model in an end-to-end manner.

Sentence-Level Selection. Let us first explain how document sentences are selected according to conversation context. Considering the context $c = (u_1, \cdots, u_n)$ and the j-th sentence s_j in the document, CSN computes a score for the sentence s_j by measuring its matching degree with the current dialogue context. In particular, CSN first obtains the sentence representations of the context c and the sentence s_j by mean-pooling over the word dimension of their sequential representations:

$$\bar{\mathbf{C}} = \underset{dim=2}{\text{mean}}(\mathbf{C}), \quad \bar{\mathbf{s}}_j = \underset{dim=1}{\text{mean}}(\mathbf{s}_j), \tag{1}$$

where $\bar{\mathbf{C}} \in \mathbb{R}^{n \times 2d}$ and $\bar{\mathbf{s}}_j \in \mathbb{R}^{2d}$. Then CSN computes a sentence-level matching vector $\mathbf{A}$ by cosine similarities:

$$\mathbf{A} = \cos(\bar{\mathbf{C}}, \bar{\mathbf{s}}_j). \tag{2}$$

We can treat $\mathbf{A} \in \mathbb{R}^n$ as a similarity array $\mathbf{A} = [A_1, \cdots, A_n]$ and compute a matching score S for the sentence s_j by fusing the similarity scores:

$$S = f(A_1, A_2, \cdots, A_n). \tag{3}$$

The fusion function $f(\cdot)$ can be designed in different ways, which will be discussed later. After obtaining the matching scores for sentences, we select the relevant sentences and update their representations as follows:

$$S' = S \times (\sigma(S) \geq \gamma), \quad \mathbf{s}'_j = S' \times \mathbf{s}_j, \tag{4}$$

where $\sigma(\cdot)$ is the Sigmoid function and γ is a hyperparameter of the gate threshold. By this means, we will filter out a sentence s_j if its relevance score is below γ. The filtering is intended to remove the impact of clearly irrelevant parts of document content.

Word-Level Selection. In the sentence-level selection, all words in a sentence are assigned the same weights. We can further perform a selection of words by computing a score for each word in the sentence. Specifically, CSN constructs a word-level matching map through the attention mechanism as follows:

$$\mathbf{B} = \mathbf{v}^\top \tanh(\mathbf{s}_j^\top \mathbf{W}_1 \mathbf{C} + \mathbf{b}_1), \tag{5}$$

where $\mathbf{W}_1 \in \mathbb{R}^{2d \times 2d \times h}$, $\mathbf{b}_1 \in \mathbb{R}^h$ and $\mathbf{v} \in \mathbb{R}^{h \times 1}$ are parameters. $\mathbf{B} \in \mathbb{R}^{n \times L \times L}$ is the word-alignment matrix between the context and the document sentence. Then, to obtain the most important matching features between s_j and each utterance in the context, CSN conducts a max-pooling operation as follows:

$$\hat{\mathbf{B}} = \max_{dim=3} \mathbf{B}, \tag{6}$$

where $\hat{\mathbf{B}} \in \mathbb{R}^{n \times L}$, and it can be represented in an array form as $\hat{\mathbf{B}} = [\hat{\mathbf{B}}_1, \cdots, \hat{\mathbf{B}}_n]$. The element $\hat{\mathbf{B}}_i \in \mathbb{R}^L$ contains L local matching signals for all words in the document sentence s_j with respect to the utterance u_i. Thereafter, CSN applies a fusion function to combine these local matching signals and obtains a global matching vector:

$$\mathbf{S} = f(\hat{\mathbf{B}}_1, \hat{\mathbf{B}}_2, \cdots, \hat{\mathbf{B}}_n). \tag{7}$$

$\mathbf{S} \in \mathbb{R}^L$ thus contains L global matching scores for all words in $\mathbf{s}_j$ to the whole context. In the next step, CSN selects the relevant words in the document and updates the document representation as follows:

$$\mathbf{S}' = \mathbf{S} \odot (\sigma(\mathbf{S}) \geq \gamma), \quad \mathbf{s}'_j = \mathbf{S}' \odot \mathbf{s}_j, \tag{8}$$

where $\odot$ is the element-wise product. Different from the sentence-level matching score S' in Eq. 4, the word-level matching score $\mathbf{S}'$ is a vector containing weights for different words.

Fusion Function. The fusion function $f(\cdot)$ in Eq. (3) and (7) is used to aggregate the matching signals with each utterance in the context. Our fusion strategies attribute different weights to the utterances in the conversation history. Two different functions are considered: (1) Linear combination – the weight of each matching signal is learned during the model training. Ideally, an utterance containing more information about the conversation topic will contribute more to the selection of document content. (2) Linear combination with decay factors. This method assumes that the topic gradually changes along the conversation and the response is usually highly related to the most recent topic in the context. Therefore, we use a decay factor $\eta \in [0, 1]$ on the utterances in the context to decrease their importance when they are far away. The matching scores are then computed as:

$$A_i = A_i * \eta^{n-i}, \quad \text{(sentence-level)} \qquad \hat{\mathbf{B}}_i = \hat{\mathbf{B}}_i * \eta^{n-i}. \quad \text{(word-level)} \tag{9}$$

The decay factor η is a hyperparameter. Note that when $\eta = 1$, it degenerates to the normal linear combination.

3.5 Matching and Aggregation

The next problem is to select the appropriate response by leveraging the selected document parts. Following a recent study [4], CSN uses a dually interactive matching structure (as shown in Fig. 2) to determine context-response matching and document-response matching, where the two kinds of matching features are modeled by the same structure.

Based on the recent work [25,27,31] that constructs different matching feature maps, in addition to using the sequential representations of the sentences, CSN also uses matching on both self-attention and cross-attention representations. Given the sequential representations of the context $\mathbf{C} = [\mathbf{u}_1, \cdots, \mathbf{u}_n]$, the document $\mathbf{D} = [\mathbf{s}'_1, \cdots, \mathbf{s}'_m]$, and the response candidate $\mathbf{r}$, CSN first constructs a word-word similarity matrix $\mathbf{M}_1$ by dot product and cosine similarity:

$$\mathbf{M}_1^{cr} = \mathbf{C}\mathbf{H}_1\mathbf{r}^\top \oplus \cos(\mathbf{C}, \mathbf{r}), \qquad \mathbf{M}_1^{dr} = \mathbf{D}\mathbf{H}_1\mathbf{r}^\top \oplus \cos(\mathbf{D}, \mathbf{r}), \tag{10}$$

where $\mathbf{H}_1 \in \mathbb{R}^{2d \times 2d}$ is a parameter, and $\oplus$ is the concatenation operation.

To better handle the gap in words between two word sequences, CSN applies the attentive module, which is similar to that used in Transformer [23]. The input of an attentive module consists of three sequences, namely query ($\mathbf{Q}$), key ($\mathbf{K}$), and value ($\mathbf{V}$). The output is a new representation of the query and is denoted as $f_{\mathrm{ATT}}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ in the remaining description.

At first, CSN uses the attentive module over the word dimension to construct multi-grained representations, which is formulated as:

$$\hat{\mathbf{C}} = f_{\mathrm{ATT}}(\mathbf{C}, \mathbf{C}, \mathbf{C}), \qquad \hat{\mathbf{D}} = f_{\mathrm{ATT}}(\mathbf{D}, \mathbf{D}, \mathbf{D}), \qquad \hat{\mathbf{r}} = f_{\mathrm{ATT}}(\mathbf{r}, \mathbf{r}, \mathbf{r}). \tag{11}$$

The second similarity matrix $\mathbf{M}_2$ is computed based on these self-attention representations:

$$\mathbf{M}_2^{cr} = \hat{\mathbf{C}}\mathbf{H}_2\hat{\mathbf{r}}^\top \oplus \cos(\hat{\mathbf{C}}, \hat{\mathbf{r}}), \qquad \mathbf{M}_2^{dr} = \hat{\mathbf{D}}\mathbf{H}_2\hat{\mathbf{r}}^\top \oplus \cos(\hat{\mathbf{D}}, \hat{\mathbf{r}}). \tag{12}$$

Then, another group of attentive modules (cross-attention) is also applied to represent semantic dependency between the context, the document, and the response candidate:

$$\tilde{\mathbf{C}} = f_{\text{ATT}}(\mathbf{C}, \mathbf{r}, \mathbf{r}), \qquad \tilde{\mathbf{r}}^{c} = f_{\text{ATT}}(\mathbf{r}, \mathbf{C}, \mathbf{C}), \tag{13}$$

$$\tilde{\mathbf{D}} = f_{\text{ATT}}(\mathbf{D}, \mathbf{r}, \mathbf{r}), \qquad \tilde{\mathbf{r}}^{d} = f_{\text{ATT}}(\mathbf{r}, \mathbf{D}, \mathbf{D}). \tag{14}$$

Next, CSN also constructs a similarity matrix $\mathbf{M}_3$ as:

$$\mathbf{M}_3^{cr} = \tilde{\mathbf{C}}\mathbf{H}_3\tilde{\mathbf{r}}^{c\top} \oplus \cos(\tilde{\mathbf{C}}, \tilde{\mathbf{r}}^{c}), \qquad \mathbf{M}_3^{dr} = \tilde{\mathbf{D}}\mathbf{H}_3\tilde{\mathbf{r}}^{d\top} \oplus \cos(\tilde{\mathbf{D}}, \tilde{\mathbf{r}}^{d}). \tag{15}$$

The above matching matrices are concatenated into two matching cubes:

$$\mathbf{M}^{cr} = \mathbf{M}_1^{cr} \oplus \mathbf{M}_2^{cr} \oplus \mathbf{M}_3^{cr}, \qquad \mathbf{M}^{dr} = \mathbf{M}_1^{dr} \oplus \mathbf{M}_2^{dr} \oplus \mathbf{M}_3^{dr}. \tag{16}$$

Then CSN applies a CNN with max-pooling operation to extract matching features from $\mathbf{M}^{cr}$ and $\mathbf{M}^{dr}$. The output feature maps are flattened as matching vectors. As a result, we obtain two series of matching vectors: (1) between the context and the response $\mathbf{v}^{cr} = [\mathbf{v}^{u_1}, \cdots, \mathbf{v}^{u_n}]$; and (2) between the selected document and the response $\mathbf{v}^{dr} = [\mathbf{v}^{s_1}, \cdots, \mathbf{v}^{s_m}]$.

Finally, CSN applies LSTMs to aggregate these two series of matching vectors into two hidden vectors (the last hidden states of the LSTMs):

$$\mathbf{h}_1 = \text{LSTM}(\mathbf{v}^{cr}), \quad \mathbf{h}_2 = \text{LSTM}(\mathbf{v}^{dr}). \tag{17}$$

These vectors are concatenated together and used to compute the final matching score by an MLP with a Sigmoid activation function:

$$g(c, d, r) = \sigma\big(\text{MLP}(\mathbf{h}_1 \oplus \mathbf{h}_2)\big). \tag{18}$$

CSN learns $g(c, d, r)$ by minimizing the following cross-entropy loss with $\mathcal{D}$:

$$\mathcal{L}(\theta) = -\sum_{(y,c,d,r)\in\mathcal{D}} [y \log(g(c, d, r)) + (1 - y) \log(1 - g(c, d, r))]. \tag{19}$$

4 Experiments

4.1 Dataset

We conduct experiments on two public datasets.

PersonaChat. [28] contains multi-turn dialogues with user profiles. The goal is to generate/retrieve a response that corresponds to the user profile, which is used as a grounding document [28]. This dataset consists of 8,939 complete dialogues for training, 1,000 for validation, and 968 for testing. Response selection is conducted at every turn of a dialogue, and the ratio of the positive and the negative samples is 1:19 in training, validation, and testing sets, resulting in 1,314,380 samples for training, 156,020 for validation, and 150,240 for testing. Positive responses are real human responses while negative ones are randomly sampled from other dialogues. To prevent the model from taking advantage of trivial word overlap, the revised version of the dataset modified the persona profiles by rephrasing, generalizing, or specializing sentences, making the task much more challenging. We use "revised" and "original" to indicate the different versions of the dataset.

CMUDoG. [30] is designed specifically for document-grounded conversation. During the conversation, the speakers are provided with a movie-related wiki article. Two scenarios are considered: (1) Only one speaker has access to the article thus she should introduce the movie to the other; (2) Both speakers have access to the article thus they have a discussion. We use the dataset provided by [29], where the data of both scenarios are merged because the size of each dataset is relatively small. Notice that the model is only asked to select a response for the user who has access to the document. The ratio of the positive and the negative is 1:19 in training, validation, and testing sets. This results in 723,180 samples for training, 48,500 for validation, and 132,740 for testing.

Following previous work [29], we employ recall at position k as evaluation metrics (R@k), where $k = \{1, 2, 5\}$. For a single sample, if the only positive candidate is ranked within top k positions, then R@$k = 1$, otherwise, R@$k = 0$. The final value is the average over all test samples. Note that R@1 is equivalent to hits@1 that is used in related work [4,28].

4.2 Baseline Models

We compare CSN using sentence-level and word-level selection (denoted as CSN-sent and CSN-word respectively) with the following models:

(1) Starspace [24] concatenates the document with the context as a long sentence and learns its similarity with the response candidate by optimizing the embeddings using the margin ranking loss and k-negative sampling. Matching is done by cosine similarity of the sum of word embeddings.
(2) Profile Memory Network [28] uses a memory network with the context as input, then performs attention over the document to find relevant sentences. The combined representation is used to select the response. This model relies on the attention mechanism to weigh document contents.
(3) Key-value (KV) Profile Memory Network [28] uses dialogue histories as keys and the next dialogue utterances as values. In addition to the memory of the document, this model has a memory of past dialogues that can influence the response selection.
(4) Transformer [23] is used in [11] as an encoder for the context, document, and response. The obtained representations are input to a memory network to conduct matching in the same way as in Profile Memory Network.
(5) DGMN [29] is the state-of-the-art model on the CMUDoG dataset. It employs a cross attention mechanism between the context and document and obtains a context-aware document representation and a document-aware context representation. The two representations and the original context representation are all matched with the response representation. The three matching features are finally combined to output the matching score.

(6) DIM [4] is the state-of-the-art model on the PersonaChat dataset. It applies a dually interactive matching structure to model the context-response matching and document-response matching respectively. DIM conducts representation, matching, and aggregation by multiple BiLSTMs, and the final matching features are used to compute the matching score by an MLP.

4.3 Implementation Details

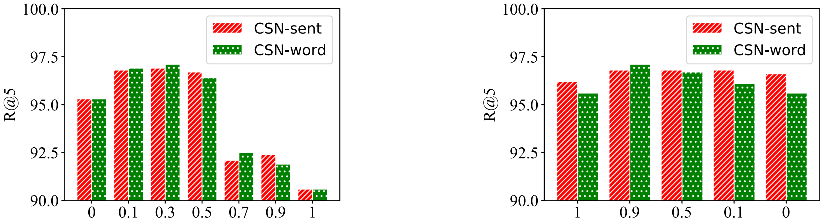
We use PyTorch [13] to implement the model. A 300-dimensional GloVe embedding [14] is used on all datasets. On PersonaChat, another 100-dimensional Word2Vec [12] embedding provided by [4] is used. Dropout [20] with a rate of 0.2 is applied to the word embeddings. All hidden sizes of the RNNs are set as 300. Two convolutional layers have 32 and 64 filters with the kernel sizes as [3, 3] and [2, 2]. AdamW [9] is employed for optimization with a batch size of 100. The initial learning rate is 0.001 and is decayed by 0.5 when the performance on the validation set is not increasing.

4.4 Experimental Results

The experimental results are shown in Table 1. The results on all three datasets indicate that our CSN outperforms all baselines, including DGMN and DIM, which are two state-of-the-art models. On the PersonaChat dataset, both CSN-word and CSN-sent achieve statistically significant improvements (p-value $\leq$ 0.05) compared with DIM, which is the best model on this dataset. In general, CSN-word performs better than CSN-sent, indicating the word-level selection is more able to select fine-grained document contents than the sentence-level selection. This comparison also confirms our intuition that it is advantageous for document-grounded conversation to rely on fine-grained information from the document. On CMUDoG, the two document content selection strategies work equally well. We explain this by the fact that the grounding document is longer in this dataset, and there is no obvious reason that one level of selection

Table 1. Experimental results on all datasets.

	PersonaChat-Original			PersonaChat-Revised			CMUDoG		
	R@1	R@2	R@5	R@1	R@2	R@5	R@1	R@2	R@5
Starspace	49.1	60.2	76.5	32.2	48.3	66.7	50.7	64.5	80.3
Profile	50.9	60.7	75.7	35.4	48.3	67.5	51.6	65.8	81.4
KV Profile	51.1	61.8	77.4	35.1	45.7	66.3	56.1	69.9	82.4
Transformer	54.2	68.3	83.8	42.1	56.5	75.0	60.3	74.4	87.4
DGMN	67.6	81.3	93.3	56.7	73.0	89.0	65.6	78.3	91.2
DIM	75.5	87.5	96.5	68.3	82.7	94.4	59.6	74.4	89.6
CSN-sent	77.5	88.8	96.8	70.1	83.4	95.1	**70.1**	82.5	**94.3**
CSN-word	**78.1**	**89.0**	**97.1**	**71.3**	**84.2**	**95.5**	69.8	**82.7**	94.0



(a) Effect of γ.

(b) Effect of η.

Fig. 3. Performance of different γ and η settings on original PersonaChat.

can determine more relevant parts than another. Nevertheless, both selection strategies show clear advantages over the baseline methods without selection.

Compared with other baselines that represent the whole document as a single vector, DGMN, DIM and our CSN consider fine-grained matching between parts of the document and response. We can see that these models achieve clearly better performances, confirming the necessity to use parts of the document rather than the whole document. However, DGMN and DIM only assign attention weights to sentences according to the context, without eliminating low-weighted ones. In contrast, our CSN model filters out all the irrelevant parts. In so doing, we expect the model not to be influenced by clearly irrelevant parts. As we can see in the experimental results, CSN achieves significantly higher performance than DGMN and DIM on all the datasets, confirming the usefulness of explicit selection (and filtering) of document contents.

Effect of Content Selection. The hyperparameter γ in Eq. (4) and (8) controls how much the document content is selected. We test the effect of this hyperparameter on the original PersonaChat dataset. Figure 3a shows that if γ is too small or too large, too much or too little information from the document may be selected. In particular, when $\gamma = 0$ – the whole document content is kept, the performance drops a lot. This strategy is comparable to that used in the existing models DIM and DGMN based on attention. We see again the usefulness of explicit document content filtering. On the other hand, when $\gamma = 1$, *i.e.*, no document content is selected, it degenerates to non document-grounded response selection and the performance also drops sharply. The best setting of γ is around 0.3 for both CSN-sent and CSN-word, which retains an appropriate amount of relevant document content for response matching.

Effect of Decaying Factor. The decay factor η works as prior knowledge to guide the model focusing more on the recent utterances. A lower η means the previous utterances have less contribution in the selection of the document. "$\eta = 1$" corresponds to the model with a normal linear combination (the first kind of fusion function). Based on the results, we can see that our decaying strategy ($\eta = 0.9$) performs the best. This confirms our assumption that focusing more on the recent topic of the conversation is helpful. However, when $\eta = 0$, only the last utterance in the history is used and the performance is lower. This illustrates the necessity of using a larger context.

5 Conclusion and Future Work

In this paper, we proposed a document content selection network to select the relevant content to ground the conversation. We designed a gate mechanism that uses conversation context to retain the relevant document contents while filtering out irrelevant parts. In addition, we also use a decay factor on the conversation history to focus on more recent utterances. Our experiments on two large-scale datasets for document-grounded response selection demonstrated the effectiveness of our model. We showed that both document content selection (and filtering) and the use of decay factor contributed in increasing the effectiveness of response selection. As a future work, it would be interesting to study if the selection can be done at topic level, in addition to sentence and word levels.

References

1. Arora, S., Khapra, M.M., Ramaswamy, H.G.: On knowledge distillation from complex networks for response prediction. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 3813–3822. Association for Computational Linguistics, Minneapolis, June 2019
2. Cai, D., Wang, Y., Bi, W., Tu, Z., Liu, X., Shi, S.: Retrieval-guided dialogue response generation via a matching-to-generation framework. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 1866–1875. Association for Computational Linguistics, Hong Kong, November 2019
3. Ghazvininejad, M., et al.: A knowledge-grounded neural conversation model. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, 2–7 February 2018, pp. 5110–5117 (2018)
4. Gu, J.C., Ling, Z.H., Zhu, X., Liu, Q.: Dually interactive matching network for personalized response selection in retrieval-based chatbots. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 1845–1854. Association for Computational Linguistics, Hong Kong, November 2019
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
6. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, 8–13 December 2014, Montreal, Quebec, Canada, pp. 2042–2050 (2014)
7. Ji, Z., Lu, Z., Li, H.: An information retrieval approach to short text conversation. CoRR abs/1408.6988 (2014)

8. Li, J., Galley, M., Brockett, C., Gao, J., Dolan, B.: A diversity-promoting objective function for neural conversation models. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 110–119. Association for Computational Linguistics, San Diego, June 2016
9. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019 (2019)
10. Lowe, R., Pow, N., Serban, I., Pineau, J.: The Ubuntu dialogue corpus: a large dataset for research in unstructured multi-turn dialogue systems. In: Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 285–294. Association for Computational Linguistics, Prague, Czech Republic, September 2015
11. Mazaré, P.E., Humeau, S., Raison, M., Bordes, A.: Training millions of personalized dialogue agents. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2775–2779. Association for Computational Linguistics, Brussels, Belgium, October–November 2018
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held 5–8 December 2013, Lake Tahoe, Nevada, United States, pp. 3111–3119 (2013)
13. Paszke, A., et al.: Pytorch: an imperative style, high-performance deep learning library. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 December 2019, pp. 8024–8035. Canada, Vancouver, BC (2019)
14. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543. Association for Computational Linguistics, Doha, Qatar, October 2014
15. Qin, L., et al.: Conversing by reading: contentful neural conversation with on-demand machine reading. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 5427–5436. Association for Computational Linguistics, Florence, Italy, July 2019
16. Ram, A., et al.: Conversational AI: the science behind the alexa prize. CoRR abs/1801.03604 (2018)
17. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.C., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, 12–17 February 2016, Phoenix, Arizona, USA, pp. 3776–3784 (2016)
18. Shang, L., Lu, Z., Li, H.: Neural responding machine for short-text conversation. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 1577–1586. Association for Computational Linguistics, Beijing, China, July 2015
19. Shum, H., He, X., Li, D.: From Eliza to XiaoIce: challenges and opportunities with social chatbots. Front. Inf. Technol. Electron. Eng. **19**(1), 10–26 (2018)
20. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)

21. Tao, C., Wu, W., Xu, C., Hu, W., Zhao, D., Yan, R.: One time of interaction may not be enough: go deep with an interaction-over-interaction network for response selection in dialogues. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1–11. Association for Computational Linguistics, Florence, Italy, July 2019
22. Tian, Z., Yan, R., Mou, L., Song, Y., Feng, Y., Zhao, D.: How to make context more useful? An empirical study on context-aware neural conversational models. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 231–236. Association for Computational Linguistics, Vancouver, Canada, July 2017
23. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA, pp. 5998–6008 (2017)
24. Wu, L.Y., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: embed all the things! In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, 2–7 February 2018, pp. 5569–5577 (2018)
25. Wu, Y., Wu, W., Xing, C., Zhou, M., Li, Z.: Sequential matching network: a new architecture for multi-turn response selection in retrieval-based chatbots. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 496–505. Association for Computational Linguistics, Vancouver, Canada, July 2017
26. Xing, C., et al.: Topic aware neural response generation. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 4–9 February 2017, San Francisco, California, USA, pp. 3351–3357 (2017)
27. Yuan, C., et al.: Multi-hop selector network for multi-turn response selection in retrieval-based chatbots. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 111–120. Association for Computational Linguistics, Hong Kong, China, November 2019
28. Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D., Weston, J.: Personalizing dialogue agents: i have a dog, do you have pets too? In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2204–2213. Association for Computational Linguistics, Melbourne, Australia, July 2018
29. Zhao, X., Tao, C., Wu, W., Xu, C., Zhao, D., Yan, R.: A document-grounded matching network for response selection in retrieval-based chatbots. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, 10–16 August 2019, pp. 5443–5449 (2019)
30. Zhou, K., Prabhumoye, S., Black, A.W.: A dataset for document grounded conversations. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 708–713. Association for Computational Linguistics, Brussels, Belgium, October–November 2018
31. Zhou, X., et al.: Multi-turn response selection for chatbots with deep attention matching network. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1118–1127. Association for Computational Linguistics, Melbourne, Australia, July 2018

Correction to: Machine Translation Customization via Automatic Training Data Selection from the Web

Thuy Vu and Alessandro Moschitti

Correction to:
Chapter “Machine Translation Customization via Automatic Training Data Selection from the Web” in: D. Hiemstra et al. (Eds.): *Advances in Information Retrieval*, LNCS 12656, https://doi.org/10.1007/978-3-030-72113-8_44

In the original version of this chapter table 3 was not correct. This has been corrected.

Table 3. BLEU-based evaluation of CSE on WMT-18

ParaCrawl	Buckets		Clean & Bucket	
	2017	2018	2017	2018
0%	–	–	27.2	32.4
0%–25%	**28.1**	**34.3**	**29.8**	**36.2**
25%–50%	23.4	27.8	27.3	32.8
50%–75%	12.7	14.7	25.2	30.3
75%–100%	5.8	6.6	25.0	29.7
0%–100%	23.7	29.21	28.22	34.41

The updated version of this chapter can be found at https://doi.org/10.1007/978-3-030-72113-8_44

D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12656, p. C1, 2021.
https://doi.org/10.1007/978-3-030-72113-8_51

Author Index