Shlomo Geva
Jaap Kamps
Ralf Schenkel (Eds.)

LNCS 7424

Focused Retrieval of Content and Structure

10th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2011
Saarbrücken, Germany, December 2011
Revised Selected Papers

Springer

Lecture Notes in Computer Science 7424

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Shlomo Geva Jaap Kamps Ralf Schenkel (Eds.)

Focused Retrieval of Content and Structure

10th International Workshop of the Initiative
for the Evaluation of XML Retrieval, INEX 2011
Saarbrücken, Germany, December 12-14, 2011
Revised Selected Papers

Springer

Volume Editors

Shlomo Geva
Queensland University of Technology (QUT)
Faculty of Science and Technology
GPO Box 2434, Brisbane, QLD 4001, Australia
E-mail: s.geva@qut.edu.au

Jaap Kamps
University of Amsterdam
Archives and Information Studies/Humanities
Turfdraagsterpad 9, 1012 XT Amsterdam, The Netherlands
E-mail: kamps@uva.nl

Ralf Schenkel
Saarland University
Cluster of Excellence
Multimodal Computing and Interaction
Campus E1 7, 66123 Saarbrücken, Germany
E-mail: schenkel@mmci.uni-saarland.de

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-35733-6 e-ISBN 978-3-642-35734-3
DOI 10.1007/978-3-642-35734-3
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012954140

CR Subject Classification (1998): H.3, H.3.3-4, H.2.8, H.2.3, H.2.4, E.1

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

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

Printed on acid-free paper

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

Preface

Welcome to the proceedings of the tenth workshop of the Initiative for the Evaluation of XML Retrieval (INEX)!

Traditional IR focuses on pure text retrieval over "bags of words," but the use of structure—such as document structure, semantic metadata, entities, or genre/topical structure—is of increasing importance on the Web and in professional search. INEX has been pioneering the use of structure for focused retrieval since 2002, by providing large test collections of structured documents, uniform evaluation measures, and a forum for organizations to compare their results. Now, in its tenth year, INEX is an established evaluation forum, with over 100 organizations registered worldwide and over 30 groups participating actively in at least one of the tracks.

INEX 2011 was an exciting year for INEX in which a number of new tasks and tracks started, including Social Search, Faceted Search, Snippet Retrieval, and Tweet Contextualization. Five research tracks were included, which studied different aspects of focused information access:

The Books and Social Search Track investigated techniques to support users in searching and navigating books, metadata, and complementary social media. The *Social Search for Best Books Task* studied the relative value of authoritative metadata and user-generated content using a collection based on data from Amazon and LibraryThing. The *Prove It Task* asked for pages confirming or refuting a factual statement, using a corpus of the full texts of 50k digitized books.

The Data-Centric Track investigated retrieval over a strongly structured collection of documents based on IMDb. The *Ad Hoc Search Task* had informational requests to be answered by the entities in IMDb (movies, actors, directors, etc.). The *Faceted Search Task* asked for a restricted list of facets and facet-values that will optimally guide the searcher toward relevant information.

The Question Answering Track investigated tweet contextualization, answering questions of the form on "what is this tweet about?" with a synthetic summary of contextual information grasped from Wikipedia and evaluated by both the relevant text retrieved and the "last point of interest."

The Relevance Feedback Track investigated the utility of incremental passage-level relevance feedback by simulating a searcher's interaction. This was an unconventional evaluation track where submissions are executable computer programs rather than search results.

The Snippet Retrieval Track investigated how to generate informative snippets for search results. Such snippets should provide sufficient information to allow the user to determine the relevance of each document, without needing to view the document itself.

The aim of the INEX 2011 workshop was to bring together researchers who participated in the INEX 2011 campaign. During the past year, participating organizations contributed to the building of a large-scale test collection. The workshop concluded the results of this large-scale effort, summarized and addressed encountered issues, and devised a work plan for the future evaluation of XML retrieval systems. These proceedings report on the final results of INEX 2011. We received a total of 36 submissions, already being a selection of work at INEX, and accepted a total of 33 papers based on peer-reviewing, yielding a 92% acceptance rate.

All INEX tracks start from having available suitable text collections. We gratefully acknowledge the data made available by: Amazon and LibraryThing (Books and Social Search Track), Microsoft Research (Books and Social Search Track), the Internet Movie Database (Data Centric Track), and the Wikimedia Foundation (Question Answering Track and Relevance Feedback Track).

Finally, INEX is run for, but especially by, the participants. It was a result of tracks and tasks suggested by participants, topics created by participants, systems built by participants, and relevance judgments provided by participants. So the main thank you goes to each of these individuals!

May 2012

Shlomo Geva
Jaap Kamps
Ralf Schenkel

Organization

Steering Committee

Charles L.A. Clarke	University of Waterloo, Canada
Norbert Fuhr	University of Duisburg-Essen, Germany
Shlomo Geva	Queensland University of Technology, Australia
Jaap Kamps	University of Amsterdam, The Netherlands
Mounia Lalmas	Yahoo! Research, Spain
Stephen E. Robertson	Microsoft Research Cambridge, UK
Ralf Schenkel	Max-Planck-Institut für Informatik, Gemany
Andrew Trotman	University of Otago, New Zealand
Ellen M. Voorhees	NIST, USA
Arjen P. de Vries	CWI, The Netherlands

Chairs

Shlomo Geva	Queensland University of Technology, Australia
Jaap Kamps	University of Amsterdam, The Netherlands
Ralf Schenkel	Max-Planck-Institut für Informatik, Germany

Track Organizers

Books and Social Search

Antoine Doucet	University of Caen, France
Jaap Kamps	University of Amsterdam, The Netherlands
Gabriella Kazai	Microsoft Research Cambridge, UK
Marijn Koolen	University of Amsterdam, The Netherlands
Monica Landoni	University of Lugano, Switzerland

Data Centric

Jaap Kamps	University of Amsterdam, The Netherlands
Maarten Marx	University of Amsterdam, The Netherlands
Georgina Ramírez Camps	Universitat Pompeu Fabra, Spain
Martin Theobald	Max-Planck-Institut für Informatik, Germany
Qiuyue Wang	Renmin University, China

Question Answering

Patrice Bellot	University of Avignon, France
Véronique Moriceau	LIMSI-CNRS, University Paris-Sud 11, France
Josiane Mothe	IRIT, Toulouse, France
Eric SanJuan	University of Avignon, France
Xavier Tannier	LIMSI-CNRS, University Paris-Sud 11, France

Relevance Feedback

Timothy Chappell	Queensland University of Technology, Australia
Shlomo Geva	Queensland University of Technology, Australia

Snippet Retrieval

Shlomo Geva	Queensland University of Technology, Australia
Mark Sanderson	RMIT, Australia
Falk Scholer	RMIT, Australia
Matthew Trappett	Queensland University of Technology, Australia
Andrew Trotman	University of Otago, New Zealand

Table of Contents

Books and Social Search Track

Data Centric Track

Relevance Feedback Track

Snippet Retrieval Track

Overview of the INEX 2011 Books and Social Search Track

Marijn Koolen[1], Gabriella Kazai[2], Jaap Kamps[1], Antoine Doucet[3], and Monica Landoni[4]

[1] University of Amsterdam, Netherlands
{marijn.koolen,kamps}@uva.nl
[2] Microsoft Research, United Kingdom
v-gabkaz@microsoft.com
[3] University of Caen, France
doucet@info.unicaen.fr
[4] University of Lugano
monica.landoni@unisi.ch

Abstract. The goal of the INEX 2011 Books and Social Search Track is to evaluate approaches for supporting users in reading, searching, and navigating book metadata and full texts of digitized books. The investigation is focused around four tasks: 1) the Social Search for Best Books task aims at comparing traditional and user-generated book metadata for retrieval, 2) the Prove It task evaluates focused retrieval approaches for searching books, 3) the Structure Extraction task tests automatic techniques for deriving structure from OCR and layout information, and 4) the Active Reading task aims to explore suitable user interfaces for eBooks enabling reading, annotation, review, and summary across multiple books. We report on the setup and the results of the track.

1 Introduction

Prompted by the availability of large collections of digitized books, e.g., the Million Book project[1] and the Google Books Library project,[2] the Books and Social Search Track[3] was launched in 2007 with the aim to promote research into techniques for supporting users in searching, navigating and reading book metadata and full texts of digitized books. Toward this goal, the track provides opportunities to explore research questions around four areas:

- The relative value of professional and user-generated metadata for searching large collections of books,
- Information retrieval techniques for searching collections of digitized books,
- Mechanisms to increase accessibility to the contents of digitized books, and
- Users' interactions with eBooks and collections of digitized books.

[1] http://www.ulib.org/
[2] http://books.google.com/
[3] Until this year the Track was known as the Book Track.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 1–29, 2012.

Based around these main themes, the following four tasks were defined:

1. The Social Search for Best Books (SB) task, framed within the user task of searching a large online book catalogue for a given topic of interest, aims at comparing retrieval effectiveness from traditional book descriptions, e.g., library catalogue information, and user-generated content such as reviews, ratings and tags.
2. The *Prove It* (PI) task aims to test focused retrieval approaches on collections of books, where users expect to be pointed directly at relevant book parts that may help to confirm or refute a factual claim;
3. The *Structure Extraction* (SE) task aims at evaluating automatic techniques for deriving structure from OCR and building hyperlinked table of contents;
4. The *Active Reading task* (ART) aims to explore suitable user interfaces to read, annotate, review, and summarize multiple books.

In this paper, we report on the setup and the results of each of these tasks at INEX 2011. First, in Section 2, we give a brief summary of the participating organisations. The four task are described in detail in the following sections: the SB task in Section 3, the PI task in Section 4, the SE task in Section 5 and the ART in Section 6. We close in Section 7 with a summary and plans for INEX 2012.

2 Participating Organisations

A total of 47 organisations registered for the track (compared with 82 in 2010, 84 in 2009, 54 in 2008, and 27 in 2007). At the time of writing, we counted 10 active groups (compared with 16 in 2009, 15 in 2008, and 9 in 2007), see Table 1.[4]

3 The Social Search for Best Books Task

The goal of the Social Search for Best Books (SB) task is to evaluate the relative value of controlled book metadata, such as classification labels, subject headings and controlled keywords, versus user-generated or social metadata, such as tags, ratings and reviews, for retrieving the most relevant books for a given user request. Controlled metadata, such as the Library of Congress Classification and Subject Headings, is rigorously curated by experts in librarianship. It is used to index books to allow highly accurate retrieval from a large catalogue. However, it requires training and expertise to use effectively, both for indexing and for searching. On the other hand, social metadata, such as tags, are less rigorously defined and applied, and lack vocabulary control by design. However, such metadata is contributed directly by the users and may better reflect the terminology of everyday searchers. Clearly, both types of metadata have advantages and disadvantages. The task aims to investigate whether one is more suitable than the

[4] The last two groups participated in the SE task via ICDAR but did not register for INEX, hence have no ID.

Table 1. Active participants of the INEX 2011 Books and Social Search Track, the task they were active in, and number of contributed runs (SB = Social Search for Best Books, PI = Prove It, SE = Structure Extraction, ART = Active Reading Task)

ID	Institute	Tasks	Runs
4	University of Amsterdam	SB	6
7	Oslo University College	PI	15
18	Universitat Pompeu Fabra	SB	6
34	Nankai University	SE	4
50	University of Massachusettes	PI	6
54	Royal School of Library and Information Science	SB	4
62	University of Avignon	SB	6
113	University of Caen	SE	3
	Microsoft Development Center Serbia	SE	1
	Xerox Research Centre Europe	SE	2

other to support different types of search requests or how they may be fruitfully combined.

The SB task aims to address the following research questions:

- How can a system take full advantage of the available metadata for searching in an online book catalogue?
- What is the relative value of social and controlled book metadata for book search?
- How does the different nature of these metadata descriptions affect retrieval performance for different topic types and genres?

3.1 Scenario

The scenario is that of a user turning to Amazon Books and LibraryThing to search for books they want to read, buy or add to their personal catalogue. Both services host large collaborative book catalogues that may be used to locate books of interest.

On LibraryThing, users can catalogue the books they read, manually index them by assigning tags, and write reviews for others to read. Users can also post messages on a discussion forum asking for help in finding new, fun, interesting, or relevant books to read. The forums allow users to tap into the collective bibliographic knowledge of hundreds of thousands of book enthusiasts. On Amazon, users can read and write book reviews and browse to similar books based on links such as "customers who bought this book also bought... ".

Users can search online book collections with different intentions. They can search for specific books of which they know all the relevant details with the intention to obtain them (buy, download, print). In other cases, they search for a specific book of which they do not know those details, with the intention of

identifying that book and find certain information about it. Another possibility is that they are not looking for a specific book, but hope to discover one or more books meeting some criteria. These criteria can be related to subject, author, genre, edition, work, series or some other aspect, but also more serendipitously, such as books that merely look interesting or fun to read.

Although book metadata can often be used for browsing, this task assumes a user issues a query to a retrieval system, which returns a (ranked) list of book records as results. This query can be a number of keywords, but also one or more book records as positive or negative examples. We assume the user inspects the results list starting from the top and works her way down until she has either satisfied her information need or gives up. The retrieval system is expected to order results by relevance to the user's information need.

3.2 Task Description

The SB task is to reply to a user's request that has been posted on the LibraryThing forums (see Section 3.5) by returning a list of recommended books. The books must be selected from a corpus that consists a collection of book metadata extracted from Amazon Books and LibraryThing, extended with associated records from library catalogues of the Library of Congress and the British Library (see the next section). The collection includes both curated and social metadata. User requests vary from asking for books on a particular genre, looking for books on a particular topic or period or books by a given author. The level of detail also varies, from a brief statement to detailed descriptions of what the user is looking for. Some requests include examples of the kinds of books that are sought by the user, asking for similar books. Other requests list examples of known books that are related to the topic but are specifically of no interest. The challenge is to develop a retrieval method that can cope with such diverse requests. Participants of the SB task are provided with a set of book search requests and are asked to submit the results returned by their systems as ranked lists.

3.3 Submissions

We want to evaluate the book ranking of retrieval systems, specifically the top ranks. We adopt the submission format of TREC, with a separate line for each retrieval result, consisting of six columns:

1. topic_id: the topic number, which is based on the LibraryThing forum thread number.
2. Q0: the query number. Unused, so should always be Q0.
3. isbn: the ISBN of the book, which corresponds to the file name of the book description.
4. rank: the rank at which the document is retrieved.
5. rsv: retrieval status value, in the form of a score. For evaluation, results are ordered by descending score.
6. run_id: a code to identifying the participating group and the run.

Participants are allowed to submit up to six runs, of which at least one should use only the *title* field of the topic statements (the topic format is described in Section 3.5). For the other five runs, participants could use any field in the topic statement.

3.4 Data

To study the relative value of social and controlled metadata for book search, we need a large collection of book records that contains controlled subject headings and classification codes as well as social descriptions such as tags and reviews, for a set of books that is representative of what readers are searching for. We use the Amazon/LibraryThing corpus crawled by the University of Duisburg-Essen for the INEX Interactive Track [1].

The collection consists of 2.8 million book records from Amazon, extended with social metadata from LibraryThing. This set represents the books available through Amazon. These records contain title information as well as a Dewey Decimal Classification (DDC) code and category and subject information supplied by Amazon. From a sample of Amazon records we noticed the subject descriptors to be noisy, with many inappropriately assigned descriptors that seem unrelated to the books to which they have been assigned.

Each book is identified by ISBN. Since different editions of the same work have different ISBNs, there can be multiple records for a single intellectual work. The corpus consists of a collection of 2.8 million records from Amazon Books and LibraryThing.com. See https://inex.mmci.uni-saarland.de/data/nd-agreements.jsp for information on how to get access to this collection. Each book record is an XML file with fields like <isbn>, <title>, <author>, <publisher>, <dimensions>, <numberofpage> and <publicationdate>. Curated metadata comes in the form of a Dewey Decimal Classification in the <dewey> field, Amazon subject headings are stored in the <subject> field, and Amazon category labels can be found in the <browseNode> fields. The social metadata from Amazon and LibraryThing is stored in the <tag>, <rating>, and <review> fields. The full list of fields is shown in Table 2.

How many of the book records have curated metadata? There is a DDC code for 61% of the descriptions and 57% of the collection has at least one subject heading. The classification codes and subject headings cover the majority of records in the collection.

More than 1.2 million descriptions (43%) have at least one review and 82% of the collection has at least one LibraryThing tag.

The distribution of books over the Amazon subject categories shows that *Literature*, *History*, *Professional and Technical* and *Religion* are some of the largest categories (see Table 3). There are also administrative categories related to sales, edition (paperback, hardcover) and others, but we show only the genre-related categories. If we look at the distribution over DDC codes (showing only

Table 2. A list of all element names in the book descriptions

tag name			
book	similarproducts	title	imagecategory
dimensions	tags	edition	name
reviews	isbn	dewey	role
editorialreviews	ean	creator	blurber
images	binding	review	dedication
creators	label	rating	epigraph
blurbers	listprice	authorid	firstwordsitem
dedications	manufacturer	totalvotes	lastwordsitem
epigraphs	numberofpages	helpfulvotes	quotation
firstwords	publisher	date	seriesitem
lastwords	height	summary	award
quotations	width	editorialreview	browseNode
series	length	content	character
awards	weight	source	place
browseNodes	readinglevel	image	subject
characters	releasedate	imageCategories	similarproduct
places	publicationdate	url	tag
subjects	studio	data	

Table 3. Amazon category distribution (in percentages)

Category	%	**Category**	%
Non-fiction	20	Science	7
Literature and fiction	20	Fiction	7
Children	14	Literature	7
History	13	Christianity	7
Reference	11	Health, Mind and Body	6
Professional and Technical.	11	Arts and Photography	5
Religion and Spirituality	10	Business and Investing	5
Social science	10	Biography and Memoirs	5

the main classes in Table 4), we see a somewhat different distribution. *Literature* is still the largest class, but is followed by *Social sciences*, *Arts and recreation*, *Technology*, then *History* and *Religion*. Note that a book has only one DDC code—it can only have one physical location on a library shelf—but can have multiple Amazon categories, which could explain the difference in distribution. Note also that all but 296 books in the collection have at least one Amazon category, while only 61% of the records have DDC codes.

Table 4. Distribution over DDC codes (in percentages)

DDC main class	%
Computer science, information and general works	4
Philosophy and psychology	4
Religion	8
Social sciences	16
Language	2
Science (including mathematics)	5
Technology and applied Science	13
Arts and recreation	13
Literature	25
History, geography, and biography	11

3.5 Information Needs

LibraryThing users discuss their books in the discussion forums. Many of the topic threads are started with a request from a member for interesting, fun new books to read. They describe what they are looking for, give examples of what they like and do not like, indicate which books they already know and ask other members for recommendations. Other members often reply with links to works catalogued on LibraryThing, which have direct links to the corresponding records on Amazon. These requests for recommendation are natural expressions of information needs for a large collection of online book records. We aim to evaluate the SB task using a selection of these forum topics.

The books suggested by members in replies to the initial message are collected in a list on the side of the topic thread (see Figure 1). A technique called *touchstone* can be used by members to easily identify books they mention in the topic thread, giving other readers of the thread direct access to a book record on LibraryThing, with associated ISBNs and links to Amazon. We use these suggested books as initial relevance judgements for evaluation. Some of these touchstones identify an incorrect book, and suggested books may not always be what the topic creator asked for, but merely be mentioned as a negative example or for some other reason. From this it is clear that the collected list of suggested books can contain false positives and is probably incomplete as not all relevant books will be suggested (false negatives), so may not be appropriate for reliable evaluation. We discuss this in more detail in Section 3.7. We first describe how we created a large set of topics, then analyse what type of topics we ended up with and how suitable they for this task.

Topic Analysis. We crawled 18,427 topic threads from 1,560 discussion groups. From these, we extracted 943 topics where the initial message contains a request for book suggestions. Each topic has a title and is associated with a group on the discussion forums. For instance, topic 99309 in Figure 1 has title *Politics of Multiculturalism Recommendations?* and was posted in the group *Political Philosophy*.

Fig. 1. A topic thread in LibraryThing, with suggested books listed on the right hand side

Not all titles are good descriptions of the information need expressed in the initial message. To identify which of these 943 topics have good descriptive titles, we used the titles as queries and retrieved records from the Amazon/LibraryThing collection and evaluated them using the suggested books collected through the touchstones. We selected all topics for which at least 50% of the suggested books were returned in the top 1000 results and manually labelled them with information about topic type, genre and specificity and extracted positive and negatives example books and authors mentioned in the initial message. Some topics had very vague requests or relied on external source to derive the information need (such as recommendations of books listed on certain web page), leaving 211 topics in the official test topic set from 122 different discussion groups.

To illustrate how we marked up the topics, we show topic 99309 from Figure 1 as an example:

```
<topic id="99309">
  <title>Politics of Multiculturalism</title>
  <group>Political Philosophy</group>
  <narrative>I'm new, and would appreciate any recommended reading on the
    politics of multiculturalism. <author>Parekh</author>'s
    <work id="164382"> Rethinking Multiculturalism: Cultural Diversity and
    Political Theory</work> (which I just finished) in the end left me un-
    convinced, though I did find much of value I thought he depended way
    too much on being able to talk out the details later. It may be that I
    found his writing style really irritating so adopted a defiant skepti-
    cism, but still... Anyway, I've read <author>Sen</author>, <author>
    Rawls</author>, <author>Habermas</author>, and <author>Nussbaum
```

```
    </author>, still don't feel like I've wrapped my little brain around
    the issue very well and would appreciate any suggestions for further
    anyone might offer.
  </narrative>
  <type>subject</type>
  <genre>politics</genre>
  <specificity>narrow</specificity>
  <similar>
    <work id="164382">
      <isbn>0333608828</isbn>
      <isbn>0674004361</isbn>
      <isbn>1403944539</isbn>
      <isbn>0674009959</isbn>
    </work>
    <author>Parekh</author>
    <author>Sen</author>
    <author>Rawls</author>
    <author>Habermas</author>
    <author>Nussbaum</author>
  </similar>
  <dissimilar><dissimilar>
</topic>
```

The distribution over topic type is shown on the left side of Table 5. The majority of topics have subject-related book requests. For instance, the topic in Figure 1 is a subject-related request, asking for books about politics and multiculturalism. Most requests (64%) are subject-related, followed by author-related (15%), then series (5%), genre (4%), edition and known-item (both 3%). Some topics can be classified with 2 types, such as subject and genre. For instance, in one topic thread, the topic creator asks for biographies of people with eating disorders. In this case, the subject is *people with eating disorders* and the genre is *biography*. The topic set covers a broad range of topic types, but for work- and language-related topics the numbers are too small to be representative. We will conduct a more extensive study of the topics to see if this distribution is representative or whether our selection method has introduced some bias.

Next, we classified topics by genre, roughly based on the main classes of the LCC and DDC (see right side of Table 5), using separate classes for philosophy and religion (similar to DDC, while LCC combines them in one main class). The two most requested genres are literature (42%, mainly prose and some poetry), and history (28%). We only show the 12 most frequent classes. There are more main classes represented by the topics, such as law, psychology and genealogy, but they only represent one or two topics each. If we compare this distribution with the Amazon category and DDC distributions in Tables 3 and 4, we see that military books are more popular among LibraryThing forum users than is represented by the Amazon book corpus, while social science is less popular. Literature, history, religion, technology are large class in both the book corpus and the topic set. The topic set is a reasonable reflection of the genre distribution of the books in the Amazon/LibraryThing collection.

Table 5. Distribution of topic types and genres

Type	Freq.	Genre	Freq.
subject	134	literature	89
author	32	history	60
series	10	biography	24
genre	8	military	16
edition	7	religion	16
known-item	7	technology	14
subject & genre	7	science	11
work	2	education	8
genre & work	1	politics	4
subject & author	1	philosophy	4
language	1	medicine	3
author & genre	1	geography	3

Furthermore, we added labels for specificity. The specificity of a topic is somewhat subjective and we based it on a rough estimation of the number of relevant books. It is difficult to come up with a clear threshold between broad and narrow, and equally hard to estimate how many books would be relevant. Broad topics have requests such as recommendations within a particular genre ("please recommend good science fiction books."), for which thousands of books could be considered relevant. The topic in Figure 1 is an example of a narrow topic. There are 177 topics labelled as narrow (84%) and 34 topics as broad (16%). We also labelled books mentioned in the initial message as either positive or negative examples of what the user is looking for. There are 58 topics with positive examples (27%) and 9 topics with negative examples (4%). These topics could be used as query-by-example topics, or maybe even for recommendation. The examples add further detail to the expressed information need and increase the realism of the topic set.

We think this topic set is representative of book information needs and expect it to be suitable for evaluating book retrieval techniques. We note that the titles and messages of the topic threads may be different from what these users would submit as queries to a book search system such as Amazon, LibraryThing, the Library of Congress or the British Library. Our topic selection method is an attempt to identify topics where the topic title describes the information need. In the first year of the task, we ask the participants to generate queries from the title and initial message of each topic. In the future, we could approach the topic creators on LibraryThing and ask them to supply queries or set up a crowdsourcing task where participants provide queries while searching the Amazon/LibraryThing collection for relevant books.

Touchstone Recommendations as Judgements. We use the recommended books for a topic as relevance judgements for evaluation. Each book in the Touchstone list is considered relevant. How many books are recommended to LT

Table 6. Statistics on the number of recommended books for the 211 topics from the LT discussion groups

# rel./topic	# topics	min.	max.	median	mean	std. dev.
All	211	1	79	7	11.3	12.5
Fiction	89	1	79	10	16.0	15.8
Non-fiction	132	1	44	6	8.3	8.3
Subject	142	1	68	6	9.6	10.0
Author	34	1	79	10	15.9	17.6
Genre	16	1	68	7	13.3	16.4

members requesting recommendations in the discussion groups? Are other members compiling exhaustive lists of possibly interesting books or do they only suggest a small number of the best available books? Statistics on the number of books recommended for the 211 topics are given in Table 6.

The number of relevant books per topic ranges between 1 and 79 with a mean of 11.3. The median is somewhat lower (7), indicating that most of the topics have a small number of recommended books. The topics requesting fiction books have more relevant books (16 on average) than the topics requesting non-fiction (8.3 on average). Perhaps this is because there is both more fiction in the collection and more fiction related topics in the topic set. The latter point suggests that fiction is more popular among LT members, such that requests for books get more responses. The breakdown over topic types *Subject*, *Author* and *Genre* shows that subject related topics have fewer suggested books than author and genre related topics. This is probably related to the distinction between fiction and non-fiction. Most of the *Subject* topics are also *Non-fiction* topics, which have fewer recommended books than *Fiction* books.

ISBNs and Intellectual Works

Each record in the collection corresponds to an ISBN, and each ISBN corresponds to a particular intellectual work. An intellectual work can have different editions, each with their own ISBN. The ISBN-to-work relation is a many-to-one relation. In many cases, we assume the user is not interested in all the different editions, but in different intellectual works. For evaluation we collapse multiple ISBN to a single work. The highest ranked ISBN is evaluated and all lower ranked ISBNs ignored. Although some of the topics on LibraryThing are requests to recommend a particular edition of a work—in which case the distinction between different ISBNs for the same work are important—we leave them out of the relevance assessment phase for this year to make evaluation easier.

However, one problem remains. Mapping ISBNs of different editions to a single work is not trivial. Different editions may have different titles and even have different authors (some editions have a foreword by another author, or a translator, while others have not), so detecting which ISBNs actually represent the same work is a challenge. We solve this problem by using mappings made by

the collective work of LibraryThing members. LT members can indicate that two books with different ISBNs are actually different manifestations of the same intellectual work. Each intellectual work on LibraryThing has a unique work ID, and the mappings from ISBNs to work IDs is made available by LibraryThing.[5]

However, the mappings are not complete and might contain errors. Furthermore, the mappings form a many-to-many relationship, as two people with the same edition of a book might independently create a new book page, each with a unique work ID. It takes time for members to discover such cases and merge the two work IDs, which means that at time, some ISBNs map to multiple work IDs. LibraryThing can detect such cases but, to avoid making mistakes, leaves it to members to merge them. The fraction of works with multiple ISBNs is small so we expect this problem to have a negligible impact on evaluation.

3.6 Crowdsourcing Judgements on Relevance and Recommendation

Members recommend books they have read or that they know about. This may be only a fraction of all the books that meet the criteria of the request. The list of recommended books in a topic thread may therefore be an incomplete list of appropriate books. Retrieval systems can retrieve many relevant books that are not recommended in the thread. On the other hand, LT members might leave out certain relevant books on purpose because they consider these books inferior to the books they do suggest.

To investigate this issue we ran an experiment on Amazon Mechanical Turk, where we asked workers to judge the relevance and make recommendations for books based on the descriptions from the Amazon/LT collection. For the PI task last year we found that relevance judgements for digitised book pages from AMT give reliable system rankings [5]. We expect that judging the relevance of an Amazon record given a narrative from the LibraryThing discussion forum has a lower cognitive load for workers, and with appropriate quality-control measures built-in, we expect AMT judgements on book metadata to be useful for reliable evaluation as well. An alternative or complement is to ask task participants to make judgements.

We pooled the top 10 results of all official runs for 24 topics and had each book judged by 3 workers. We explicitly asked workers to first judge the book on topical relevance and with a separate question asked them to indicate whether they would also recommend it as one the best books on the requested topic.

Topic Selection

For the Mechanical Turk judgements, we 24 topics from the set of 211, 12 fiction and 12 non-fiction. We selected the following 12 fiction topics: 17299, 25621, 26143, 28197, 30061, 31874, 40769, 74433, 84865, 94888, 92178 and 106721. We selected the following 12 non-fiction topics: 3963, 12134, 14359, 51583, 65140, 83439, 95533, 98106, 100674, 101766, 107464 and 110593.

[5] See: `http://www.librarything.com/feeds/thingISBN.xml.gz`

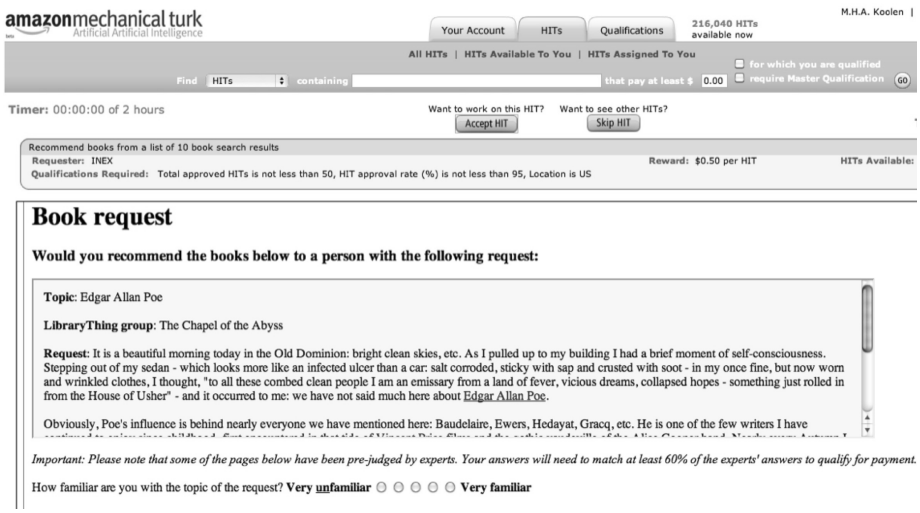


Fig. 2. Snapshot of the AMT book request

Pooling
We pooled the top 10 results per topic of all 22 submitted runs. If the resulting pool was smaller than 100 books, we continued the round-robin pooling until each pool contained at least 100 books.

Generating HITs
Each HIT contains 10 books, with at least one book that was recommended in the topic thread on the LibraryThing discussion group for validation. In total, 269 HITs were generated, and each HIT was assigned to 3 workers, who got paid $0.50 per HIT. With a 10% fee charged by Amazon per HIT, the total cost was $269 * 3 * \$0.50 * 1.1 = \$443.85$.

HIT Design
The design of the HIT is illustrated in Figures 2, 3, 4 and 5. The HIT starts with short instructions explaining what the task is and what the goal of the task is, after which the request is shown (see Figure 2). After the request, worker get a list of 10 book questionnaires, with each questionnaire containing frame with official metadata (Figure 3), user-generated metadata (Figure 4) and a list of questions (Figure 5). The official metadata consists of the title information, publisher information and the Amazon categories, subject headings and classification information. The user-generated metadata consists of user reviews and ratings from Amazon and user tags from LibraryThing.

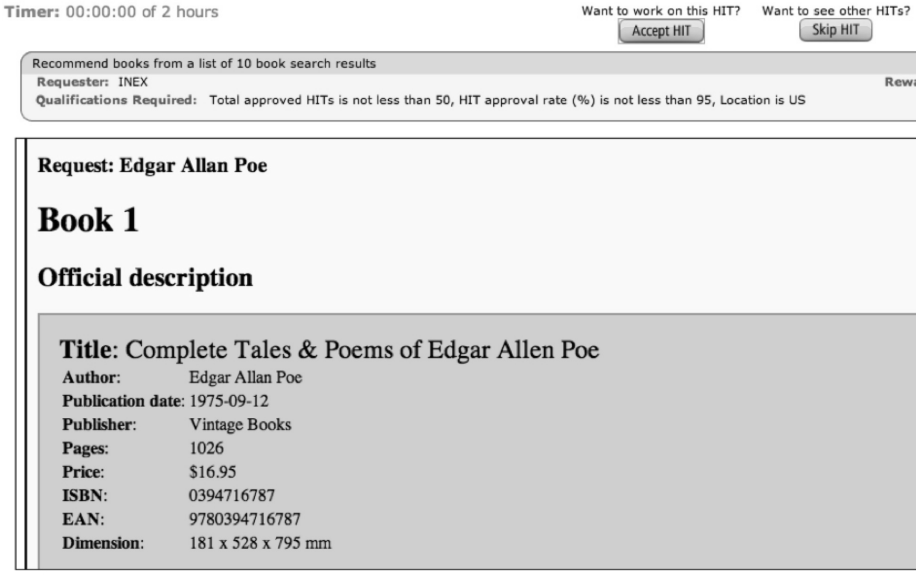


Fig. 3. Snapshot of the AMT design for the official description

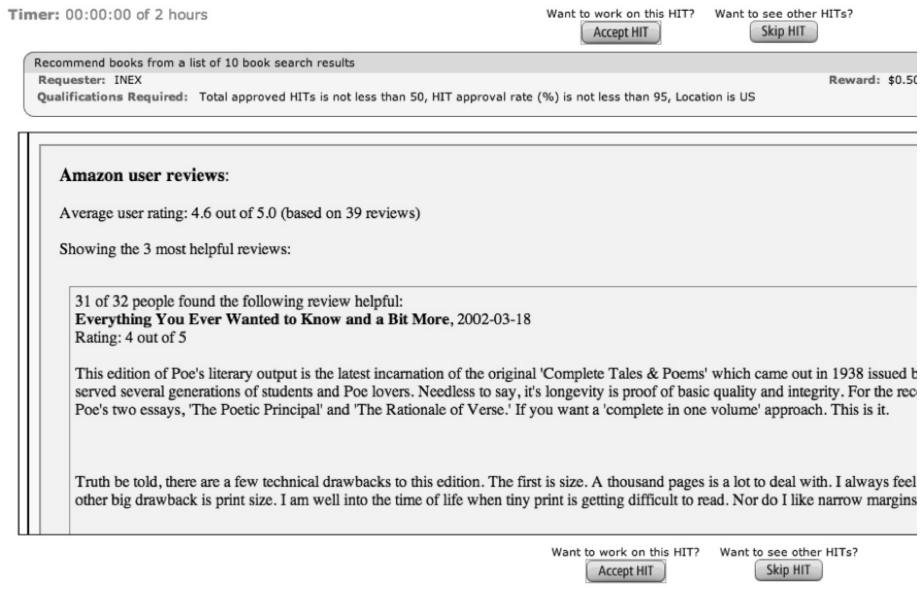


Fig. 4. Snapshot of the AMT design for the user-generated description

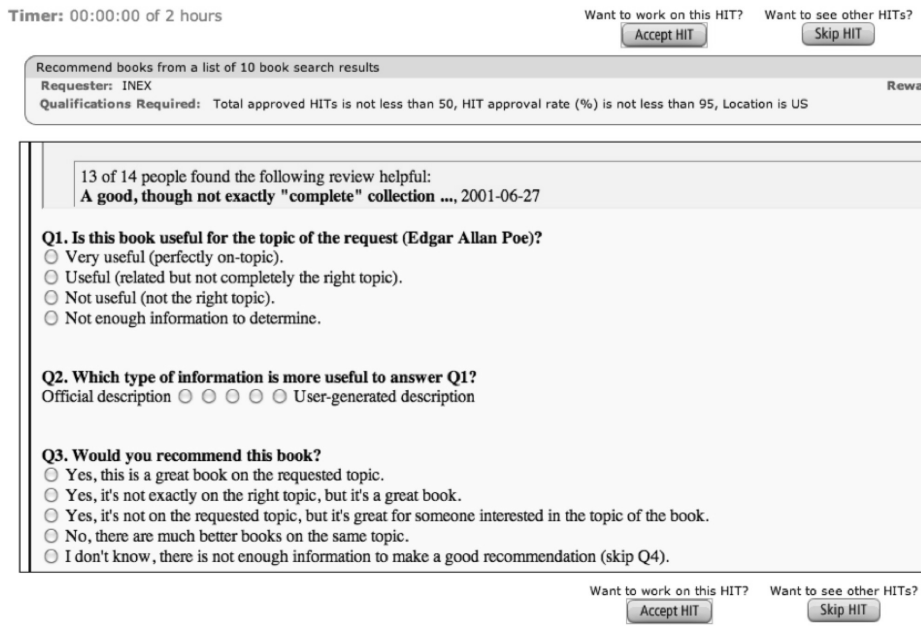


Fig. 5. Snapshot of the AMT questionnaire design

The questionnaire has 5 questions:

- **Q1. Is this book useful for the topic of the request?** Here workers can choose between
  - *perfectly on-topic*,
  - *related but not completely the right topic*,
  - *not the right topic* and
  - *not enough information*.
- **Q2. Which type of information is more useful to answer Q1?** Here workers have to indicate whether the official or user-generated metadata is more useful to determine relevance.
- **Q3. Would you recommend this book?** Here workers can choose between
  - *great book on the requested topic*,
  - *not exactly on the right topic, but it's a great book*,
  - *not on the requested topic, but it's great for someone interested in the topic of the book*,
  - *there are much better books on the same topic*, and
  - *not enough information to make a good recommendation*.
- **Q4. Which type of information is more useful to answer Q3?** Here workers have to indicate whether the official or user-generated metadata is more useful to base their recommendation on.

Table 7. Statistics on the number of recommended books for the 211 topics from the LT discussion groups

# rel./topic	# topics	min.	max.	median	mean	std. dev.
LT all	211	1	79	7	11.3	12.5
LT Fiction	89	1	79	10	16.0	15.8
LT Non-fiction	132	1	44	6	8.3	8.3
LT (24 AMT topics)	24	2	79	7	15.7	19.3
AMT all	24	4	56	25	25.0	12.7
AMT fiction	12	4	30	25	22.8	10.8
AMT non-fiction	12	4	56	29	27.3	13.7

– **Q5. Please type the most useful tag (in your opinion) from the LibraryThing tags in the User-generated description.** Here workers had to pick one of the LibraryThing user tags as the most useful, or tick the box *or tick here if there are no tags for this book* when the user-generated metadata has no tags.

There was also an optional comments field per book.

Agreement
What is the agreement among workers? We compute the pairwise agreement on relevance among workers per HIT in three different ways. The most strict agreement distinguishes between the four possible answers: 1) *Perfectly on-topic*, 2) *related but not perfect*, 3) *not the right topic* and 4) *not enough information*. In this case agreement is 0.54. If we consider only answer 1 as relevant and merge answers 2 and 3 (related means not relevant), agreement is 0.63. If we also take answer 4 to mean non-relevant (merging 2, 3 and 4, giving binary judgements), agreement is 0.68.

Recall that each HIT has at least one book that is recommended on the LT discussion thread. The average agreement between workers and forum members is 0.52. That is, on average, each worker considered 52% of the books recommended on LT as *perfectly on-topic*. We turn the AMT relevance data from multiple workers into binary relevance judgements per book using majority vote. We only consider the *perfectly on-topic* category as relevant and map the other categories to non-relevant. For most books we have 3 votes, which always leads to a majority. Some books occur in multiple HITs because they are added as known relevant books from the LT forums. If there are fewer recommended books in the LT forum than there are HITs, some books have to be included in multiple HITs. Books with judgements from an even number of workers could have tied votes. In these cases we use the fact that the book was recommended on the LT topic thread as the deciding vote and label the book as relevant.

How does the relevance distribution of the AMT judgements compare to the relevance judgements from the LT discussion groups? We compare the AMT relevance judgements with the recommendations from LT in Table 7. The fiction topics have more LT recommendations than the non-fiction, but fewer relevant

Table 8. Evaluation results for the official submissions using the LT relevance judgements of all 211 topics. Best scores are in bold

Run	nDCG@10	P@10	MRR	MAP
p4-inex2011SB.xml_social.fb.10.50	**0.3101**	**0.2071**	**0.4811**	**0.2283**
p54-run4.all-topic-fields.reviews-split.combSUM	0.2991	0.1991	0.4731	0.1945
p4-inex2011SB.xml_social	0.2913	0.1910	0.4661	0.2115
p54-run2.all-topic-fields.all-doc-fields	0.2843	0.1910	0.4567	0.2035
p62.recommandation	0.2710	0.1900	0.4250	0.1770
p62.sdm-reviews-combine	0.2618	0.1749	0.4361	0.1755
p18.UPF_QE_group_BTT02	0.1531	0.0995	0.2478	0.1223
p18.UPF_QE_genregroup_BTT02	0.1327	0.0934	0.2283	0.1001

books according to the AMT workers. This might be a sign that, without having read the book, judging the relevance of fiction books is harder than that of non-fiction books. For fiction there is often more to the utility of a book (whether it is interesting and/or fun) than the subject and genre information provided by book metadata. Or perhaps the relevance of fiction books is not harder to judge, but fiction is less readily considered relevant. For non-fiction information needs, the subject of a book may be one of the main aspects on which the relevance of the book is based. For fiction information needs, the subject of a book might play no role in determine its relevance. Another explanation might that the judgements pools based on the official runs might be better for non-fiction topics than for fiction topics.

3.7 Evaluation

For some topics, relevance may be both trivial and complex. Consider a topic where a user asks for good historical fiction books. The suggestions from the LT members will depend on their ideas of what are good historical fiction books. From the metadata alone it is hard to make this judgement. Should all historical fiction books be considered relevant, or only the ones suggested by the LT members? Or should relevance be graded?

For now, we will use a one-dimensional relevance scale, but like to explore alternatives in the future. One way would be to distinguish between books that a user considers as interesting options to read next and the actual book or books she decides to obtain and read. This roughly corresponds to the distinction between the library objective of helping to *find or locate* relevant items and the objective of helping to *choose* which of the relevant items to access [7].

We first show the results for the 211 topics and associated relevance judgements from the LT forums in Table 8. The best SB run (nDCG@10=0.3101) was submitted by the University of Amsterdam (p4-inex2011SB.xml_social.fb.10.50), which uses pseudo relevance feedback on an index with only reviews and tags in addition with the basic title information.

Table 9. Evaluation results for the official submissions using the AMT relevance judgements. Best scores are in bold

Run	nDCG@10	P@10	MRR	MAP
p62.baseline-sdm	**0.6092**	**0.5875**	0.7794	0.3896
p4-inex2011SB.xml_amazon	0.6055	0.5792	0.7940	0.3500
p62.baseline-tags-browsenode	0.6012	0.5708	0.7779	**0.3996**
p4-inex2011SB.xml_full	0.6011	0.5708	0.7798	0.3818
p54-run2.all-topic-fields.all-doc-fields	0.5415	0.4625	**0.8535**	0.3223
p54-run3.title.reviews-split.combSUM	0.5207	0.4708	0.7779	0.2515
p18.UPF_base_BTT02	0.4718	0.4750	0.6276	0.3269
p18.UPF_QE_group_BTT02	0.4546	0.4417	0.6128	0.3061

What is surprising is that some systems score high on MRR while the number of forum suggestions is small and not based on top-k pooling. With only a median of 7 suggested books per topic in a collection of 2.8 million books, the forum suggestions may well be highly incomplete. That is, there might be hundreds of other books that would have made equally good suggestions. If that were the case, it would be difficult for retrieval systems to obtain a high score, as they would likely place different books in the top ranks than the forum members. If there are a hundred relevant books in the collection and the forum members randomly picked 7 of them as suggestions, the probability that a retrieval system will rank several of those 7 in the top 10 is small. Of course, this could happen for a single topic, but an average MRR of 0.4811 over 211 topics would be extremely unlikely. This suggests that the forum suggestions are not drawn from a much larger set of equally relevant books, but form a more or less complete set of the best or most popular books for the requested topic.

Next we show the results for the 24 topics selected for the AMT experiment and associated relevance judgements in Table 9. The best SB run (nDCG@10= 0.6092) was submitted by the University of Avignon (p62-baseline-sdm). The most striking difference with the LT forum judgements is that here the scores for all runs are much higher. There are at least three possible explanations for this. First, the AMT judgements are based on the top 10 results of all runs, meaning all top 10 results of each run is judged, whereas many top ranked documents are not covered by the LT forum judgements. Second, the AMT judgements are explicitly based on topical relevance, whereas the LT forum judgements are probably more like recommendations, where users only suggest the best books on a topic and often only books they know about or have read. The high scores of the submitted runs indicates that systems are good a finding topically relevant books. The third possible explanation is that the two evaluations are based on different topic sets. The LT forum evaluation is based on 211 topics, while the AMT evaluation is based on a subset of 24 topics.

To rule out that last explanation, we also evaluated the submitted runs using the LT forum judgements only on the subset of 24 topics selected for the AMT experiment. The results for this are shown in Table 10. The topic set has little

Table 10. Evaluation results for the official submissions using the LT relevance judgements for the 24 topics used in AMT. Best scores are in bold.

Run	ndcg@10	P@10	MRR	MAP
p4-inex2011SB.xml_social.fb.10.50	**0.3039**	**0.2120**	**0.5339**	0.1994
p54-run2.all-topic-fields.all-doc-fields	0.2977	0.1940	0.5225	**0.2113**
p4-inex2011SB.xml_social	0.2868	0.1980	0.5062	0.1873
p54-run4.all-topic-fields.reviews-split.combSUM	0.2601	0.1940	0.4758	0.1515
p62.recommandation	0.2309	0.1720	0.4126	0.1415
p62.sdm-reviews-combine	0.2080	0.1500	0.4048	0.1352
p18.UPF_QE_group_BTT02	0.1073	0.0720	0.2133	0.0850
p18.UPF_QE_genregroup_BTT02	0.0984	0.0660	0.1956	0.0743

Table 11. Evaluation results using the LT recommendation Qrels across fiction and non-fiction topics

	nDCG@10		
Run	**All**	**Fiction**	**Non-fiction**
p4-inex2011SB.xml_social.fb.10.50	**0.3101**	**0.3469**	0.2896
p54-run4.all-topic-fields.reviews-split.combSUM	0.2991	0.3062	**0.2908**
p4-inex2011SB.xml_social	0.2913	0.3157	0.2783
p54-run2.all-topic-fields.all-doc-fields	0.2843	0.3145	0.2627
p62.recommandation	0.2710	0.2779	0.2694
p62.sdm-reviews-combine	0.2618	0.2680	0.2609
p18.UPF_QE_group_BTT02	0.1531	0.1505	0.1533
p18.UPF_QE_genregroup_BTT02	0.1327	0.1474	0.1238

impact, as the results for the subset of 24 topics are very similar to the results for the 211 topics. This is a first indication that the LT forum test collection is robust with respect to topic selection. It also suggests that the LT forum and AMT judgements reflect different tasks. The latter is the more traditional topical relevance task, while the former is closer to recommendation. We are still in the process of analysing the rest of the AMT data to establish to what extent the LT forum suggestions reflect relevance and recommendation tasks.

Recall that we added genre labels to all the topics. We divide the topics into two sets, one with fiction related topics and one with non-fiction related topics. All the topics with the label *literature* are considered fiction related. All other topics are considered non-fiction topics. Table 11 shows the nDCG@10 results over the topics sets split over topic genre. Most systems perform slightly better on the fiction topics than on the non-fiction topics. One reason might be that more books are suggested for fiction-related topics (see Table 7). Another reason might be that fiction books are more popular and therefore have more detailed descriptions in the form of tags and reviews and are easier to retrieve and rank.

We also split the 211 topics over topic types. The most frequent topic types are *subject* (books on a particular subject), *author* (books by a particular author)

Table 12. Evaluation results using the LT recommendation Qrels across fiction and non-fiction topics

Run	nDCG@10			
	All	**Subject**	**Author**	**Genre**
p4-inex2011SB.xml_social.fb.10.50	**0.3101**	0.2644	**0.4645**	0.1466
p54-run4.all-topic-fields.reviews-split.combSUM	0.2991	**0.2658**	0.4368	0.1905
p4-inex2011SB.xml_social	0.2913	0.2575	0.4006	0.1556
p54-run2.all-topic-fields.all-doc-fields	0.2843	0.2435	0.4002	**0.2029**
p62.recommandation	0.2710	0.2411	0.3866	0.1248
p62.sdm-reviews-combine	0.2618	0.2386	0.3686	0.1250
p18.UPF_QE_group_BTT02	0.1531	0.1116	0.2331	0.0401
p18.UPF_QE_genregroup_BTT02	0.1327	0.1021	0.1913	0.0566

and *genre* (books in a particular genre). The nDCG@10 results over topic types are shown in Table 12. The general pattern is that author topics are easier than subject and genre topics, and subject topics are easier than genre topics. This is not surprising, given that author names are often highly specific which makes them good retrieval cues. With only a small set of books matching the author name, it is not hard to retrieve the books suggested by forum members. Subject descriptions are less specific and target a larger set of books, making it harder to single out the suggested books from other books on the same subject. Finally, genre labels are even less specific and vague at best. Forum members argue over different definitions science fiction or whether a book is fiction or non-fiction. The set of books belonging to a genre can also be very large—thousands or ten of thousands of books—such that forum members disagree over what the best suggestions are. These topics may be harder than other topic types, and as a result, IR systems perform poorly on these topics.

To sum up, the forum suggestions represent a different task from traditional topical relevance search. They introduce no pooling bias, but the fact that some systems score high on early precision indicate the suggestions are relatively complete. The high scores for the AMT judgements indicates that many systems are capable of finding topically relevant books, which further indicates that the book suggestions represent an interesting and realistic new task. There are no big differences between book requests for fiction and non-fiction, and author-related topics are easier than subject-related topics, which in turn are easier than genre-related topics.

3.8 Discussion

Relevance or Recommendation?
Readers may not only base their judgement on the topical relevance—is this book a historical fiction book—but also on their personal taste. Reading a book is often not just about relevant content, but about interesting, fun or engaging content. Relevance in book search might require different dimensions of graded

judgements. The topical dimension (how topically relevant is this book?) is separate from the interestingness dimension (how interesting/engaging is this book?) Many topic creators ask for recommendations, and want others to explain their suggestions, so that they can better gauge how a book fits their taste.

So far, we only use the suggestions in the forum discussions as binary relevance judgements. However, the forum discussions contain more information than that. Some books are suggested by multiple forum members, and some books receive a negative recommendation. On top of that, the suggestions come from other members than the topic creator, and might not coincide with her actual interest.

We will report on the analysis of the AMT questionnaire data separately, but preliminary results suggest that workers treat topical relevance and recommendation similarly. When they consider a book topically relevant, they almost always recommend it as well, and do not recommend it when it is not relevant, even though there is an answer category for books that are on a different topic but are very good books for that topic. This might be the case because we asked workers to judge topical relevance and recommendation in the same questionnaire. Because the questions about recommendation were framed in the context of a specific book request, workers may have interpreted recommendation in terms of that request. Also, most books have favourable reviews, which makes it harder to distinguish between books other than to look at their relation to the requested topic. For books with no reviews, workers often indicated they did not have enough information to make a recommendation judgement. Furthermore, it seems that systems that focus more reviews for ranking are relatively more effective for recommendation than for topical relevance. This suggests, not surprisingly, that assessors mainly base their recommendations on reviews.

Next year we will look more carefully at different aspects of relevance, such as topical relevance, recommendation, reading level and whether a books looks interesting or engaging. We also plan to analyse the suggestions in more detail and differentiate between books suggested by single and multiple forum members, positive and negative suggestions and suggested books that the topic creator decided to add to her personal catalogue.

Judging Metadata or Book Content

In a realistic scenario, a user judges the relevance or interestingness of the book metadata, not of the content of the book. The decision to read a book comes before the judgement of the content. This points at an important problem with the suggested books collected through the touchstones. Members often suggest books they have actually read, and therefore base their suggestion on the actual content of the book. Such a relevance judgement—from someone other than the topic creator—is very different in nature from the judgement that the topic creator can make about books she has not read. Considering the suggested books as relevant brushes over this difference. We will further analyse the relevance and recommendation judgements from AMT to find out to what extent the LT forum suggestions reflect traditional topical relevance judgements and to what extent they reflect recommendation.

Extending the Collection
The Amazon/LibraryThing collection has a limited amount of professional metadata. Only 61% of the books have a DDC code and the Amazon subjects are noisy with many seemingly unrelated subject headings assigned to books. To make sure there is enough high-quality metadata from traditional library catalogues, we will extend the data set next year with library catalogue records from the Library of Congress and the British Library. We only use library records of ISBNs that are already in the collection. These records contain formal metadata such as classification codes (mainly DDC and LCC) and rich subject headings based on the Library of Congress Subject Headings (LCSH).[6] Both the LoC records and the BL records are in MARCXML[7] format. We obtained MARCXML records for 1.76 million books in the collection. Although there is no single library catalogue that covers all books available on Amazon, we think these combined library catalogues can improve both the quality and quantity of professional book metadata.

4 The Prove It (PI) Task

The goal of this task was to investigate the application of focused retrieval approaches to a collection of digitized books. The scenario underlying this task is that of a user searching for specific information in a library of books that can provide evidence to confirm or reject a given factual statement. Users are assumed to view the ranked list of book parts, moving from the top of the list down, examining each result. No browsing is considered (only the returned book parts are viewed by users).

Participants could submit up to 10 runs. Each run could contain, for each of the 83 topics (see Section 4.2), a maximum of 1,000 book pages estimated relevant to the given aspect, ordered by decreasing value of relevance.

A total of 18 runs were submitted by 2 groups (6 runs by UMass Amhers (ID=50) and 12 runs by Oslo University College (ID=100)), see Table 1.

4.1 The Digitized Book Corpus

The track builds on a collection of 50,239 out-of-copyright books[8], digitized by Microsoft. The corpus is made up of books of different genre, including history books, biographies, literary studies, religious texts and teachings, reference works, encyclopaedias, essays, proceedings, novels, and poetry. 50,099 of the books also come with an associated MAchine-Readable Cataloging (MARC) record, which contains publication (author, title, etc.) and classification information. Each book in the corpus is identified by a 16 character long

[6] For more information see: http://www.loc.gov/aba/cataloging/subject/
[7] MARCXML is an XML version of the well-known MARC format. See: http://www.loc.gov/standards/marcxml/
[8] Also available from the Internet Archive (although in a different XML format).

bookID – the name of the directory that contains the book's OCR file, e.g., A1CD363253B0F403.

The OCR text of the books has been converted from the original DjVu format to an XML format referred to as BookML, developed by Microsoft Development Center Serbia. BookML provides additional structure information, including markup for table of contents entries. The basic XML structure of a typical book in BookML is a sequence of pages containing nested structures of regions, sections, lines, and words, most of them with associated coordinate information, defining the position of a bounding rectangle ([coords]):

```
<document>
 <page pageNumber="1" label="PT_CHAPTER" [coords] key="0" id="0">
  <region regionType="Text" [coords] key="0" id="0">
   <section label="SEC_BODY" key="408" id="0">
    <line [coords] key="0" id="0">
     <word [coords] key="0" id="0" val="Moby"/>
     <word [coords] key="1" id="1" val="Dick"/>
    </line>
    <line [...]><word [...] val="Melville"/>[...]</line>[...]
   </section>   [...]
  </region>     [...]
 </page>        [...]
</document>
```

BookML provides a set of labels (as attributes) indicating structure information in the full text of a book and additional marker elements for more complex structures, such as a table of contents. For example, the first label attribute in the XML extract above signals the start of a new chapter on page 1 (label="PT_CHAPTER"). Other semantic units include headers (SEC_HEADER), footers (SEC_FOOTER), back-of-book index (SEC_INDEX), table of contents (SEC_TOC). Marker elements provide detailed markup, e.g., for table of contents, indicating entry titles (TOC_TITLE), and page numbers (TOC_CH_PN), etc.

The full corpus, totaling around 400GB, was made available on USB HDDs. In addition, a reduced version (50GB, or 13GB compressed) was made available for download. The reduced version was generated by removing the word tags and propagating the values of the `val` attributes as text content into the parent (i.e., line) elements.

4.2 Topics

We use the same topic set as last year [6], consisting of 83 topics. Last year, relevance judgements were collected for 21 topics from two sources. In the first phase, INEX participants judged pages using the relevance assessment system developed at Microsoft Research Cambridge.[9] In the second phase, relevance judgements were collected from Amazon Mechanical Turk.

[9] http://www.booksearch.org.uk

Table 13. Results for the 2011 Prove It evaluation using the 21 topics and judgements of the 2010 Prove It task. The run names of participant p100 (UMass) have been shortened to fit on the page. Best scores are in bold.

Run	MAP	MRR	P@10	nDCG@10 (0-1-2)	nDCG@10 (0-1-10)
p100-spec_10x_ge_55.res	0.0285	0.3271	0.1667	0.1238	0.0862
p100-spec_2x_ge_55.res	0.0255	0.3015	0.1619	0.1144	0.0788
p100-spec_5x_ge_55.res	0.0278	0.2995	0.1571	0.1145	0.0767
p100-to_g_10xover2.res	0.0491	0.4137	0.2810	0.2046	0.1469
p100-to_g_2xover2.res	0.0488	0.3855	0.2714	0.1963	0.1406
p100-to_g_5xover2.res	0.0490	0.4102	0.2762	0.2013	0.1439
p50.sdm.pass100.lambda0.025	0.3360	**1.0000**	0.7905	0.7809	0.7358
p50.sdm.pass50.lambda0.025	**0.3364**	**1.0000**	**0.8000**	**0.7842**	0.7365
p50.sdm	0.3330	**1.0000**	0.7905	0.7806	0.7356
p50.stopped.sdm.pass100.lambda0.025	0.3172	0.9762	0.7905	0.7767	0.7364
p50.stopped.sdm.pass50.lambda0.025	0.3177	0.9762	0.7905	0.7771	0.7369
p50.stopped.sdm	0.3136	0.9762	0.7905	0.7772	**0.7382**

4.3 Collected Relevance Assessments

The 2011 topic set is the same as the 2010 topic set, consisting of 21 topics. We reuse the judgements from last year and extend them with judgements based on top 10 pools of the official submissions. The UMass group (participant ID p100) provided their own judgements which we were kindly allowed to use for evaluation. In total, they judged 535 extra pages on top of the 2010 data set. The top 10 results of the runs submitted by OUC (participant ID p50) were pooled and judged using Mechanical Turk.

We used roughly the same design as last year [5], but with 6 pages per HIT instead of 10 and paying $0.30 per HIT, resulting in 92 HITs. Also, instead of asking workers to type the first word of the confirming/refuting sentence, we ask them to click on the first word of that sentence in the book page. We log the clicks, which allows us to check whether workers clicked inside the book page. This gives 419 new page-level judgements: 352 non-relevant pages, 21 relevant pages, 2 pages refuting the factual statement and 44 confirming it.

4.4 Evaluation Measures and Results

Similar to last year, the official evaluation measure is nDCG@10. Pages that confirm or refute a statement have a relevance value rv=2 and pages that are merely related to the topic have a relevance value rv=1. The evaluation results are shown in Table 13. As an alternative evaluation, we use judgements with extra weight on the confirm refute pages. The scores in column 6 in the table represent scores for the judgment where confirm/refute pages are weighted 10 (rv=10) times as much as pages that are merely relevant (rv=1). The runs submmited by UMass score very high on the official measure nDCG@10 and

three runs (starting with p50.sdm) get a perfect score on MRR. Their runs are based on Sequential Dependence Modelling, which is an interpolation between three language models based unigrams, bigrams and proximity respectively. By adjusting the Dirichlet smoothing parameter to the average number of words per page ($\mu = 363$), the SDM model is very effective in locating confirming and refuting pages.

The evaluation results of this year show that the current Prove It task can be adequately solved. For next year's Prove It task, we will introduce further challenges in identifying confirming and refuting information. One possibility is to use the confirm/refute label in the evaluation measure. That is, systems have to determine whether a page confirms or refutes a statement. As most systems find almost no refuting pages, it would seem that a trivial solution of labelling all returned results as confirming would score very high. This could be used as a baseline, which might encourage participants to focus more on finding refute pages so as to beat this baseline.

Again, the complexity of the factual statement of many topics caused problems for assessors. The statements often consist of multiple atomic facts, which confronts assessors with the problem of deciding whether a page confirms or refutes a statement when only one or some of the atomic facts are confirmed or refuted. A possible solution may be to make the topics more structured by splitting complex statements into their atomic parts, and asking assessors to judge pages on each part of the statement.

5 The Structure Extraction (SE) Task

The goal of the SE task was to test and compare automatic techniques for extracting structure information from digitized books and building a hyperlinked table of contents (ToC). The task was motivated by the limitations of current digitization and OCR technologies that produce the full text of digitized books with only minimal structure markup: pages and paragraphs are usually identified, but more sophisticated structures, such as chapters, sections, etc., are typically not recognised.

In 2011, the task was run for the second time as a competition of the International Conference on Document Analysis and Recognition (ICDAR). Full details are presented in the corresponding specific competition description [4]. This year, the main novelty was the fact that the ground truth data built in 2009 and 2010 was made available online[10]. Participants were hence able to build and fine tune their systems using training data.

Participation

Following the call for participation issued in January 2011, 11 organizations registered. As in previous competitions, several participants expressed interest

[10] http://users.info.unicaen.fr/~doucet/StructureExtraction/training/

but renounced due to time constraints. Of the 11 organizations that signed up, 5 dropped out; that is, they neither submitted runs, nor participated in the ground truth annotation process. The list of active participants is given in Table 14. Interestingly, half of them are newcomers (Nankai University, NII Tokyo and University of Innsbrück).

Table 14. Active participants of the Structure Extraction task

Organization	Submitted runs	Ground truthing
Microsoft Development Center (Serbia)	1	y
Nankai University (PRC)	4	y
NII Tokyo (Japan)	0	y
University of Caen (France)	3	y
University of Innsbrück (Austria)	0	y
Xerox Research Centre Europe (France)	2	y

Results

As in previous years [3], the 2011 task permitted to gather manual annotations in a collaborative fashion. The efforts of the 2011 round gave way to the gathering and addition of 513 new annotated book ToCs to the previous 527.

A summary of the performance of all the submitted runs is given in Table 15.

Table 15. Summary of performance scores for the Structure Extraction competition 2011 (F-measures)

RunID	Participant	Title-based [3]	Link-based [2]
MDCS	MDCS	40.75%	65.1%
Nankai-run1	Nankai U.	33.06%	63.2%
Nankai-run4	Nankai U.	33.06%	63.2%
Nankai-run2	Nankai U.	32.46%	59.8%
Nankai-run3	Nankai U.	32.43%	59.8%
XRCE-run1	XRCE	20.38%	57.6%
XRCE-run2	XRCE	18.07%	58.1%
GREYC-run2	University of Caen	8.99%	50.7%
GREYC-run1	University of Caen	8.03%	50.7%
GREYC-run3	University of Caen	3.30%	24.4%

The Structure Extraction task was launched in 2008 to compare automatic techniques for extracting structure information from digitized books. While the construction of hyperlinked ToCs was originally thought to be a first step on the way to the structuring of digitized books, it turns out to be a much tougher nut to crack than initially expected.

Future work aims to investigate into the usability of the extracted ToCs. In particular we wish to use qualitative measures in addition to the current precision/recall evaluation. The vast effort that this requires suggests that this can hardly be done without crowdsourcing. We shall naturally do this by building on the experience of the Book Search tasks described earlier in this paper.

6 The Active Reading Task (ART)

The main aim of the Active Reading Task (ART) is to explore how hardware or software tools for reading eBooks can provide support to users engaged with a variety of reading related activities, such as fact finding, memory tasks, or learning. The goal of the investigation is to derive user requirements and consequently design recommendations for more usable tools to support active reading practices for eBooks. The task is motivated by the lack of common practices when it comes to conducting usability studies of e-reader tools. Current user studies focus on specific content and user groups and follow a variety of different procedures that make comparison, reflection, and better understanding of related problems difficult. ART is hoped to turn into an ideal arena for researchers involved in such efforts with the crucial opportunity to access a large selection of titles, representing different genres, as well as benefiting from established methodology and guidelines for organising effective evaluation experiments.

The ART is based on the evaluation experience of EBONI [8], and adopts its evaluation framework with the aim to guide participants in organising and running user studies whose results could then be compared.

The task is to run one or more user studies in order to test the usability of established products (e.g., Amazon's Kindle, iRex's Ilaid Reader and Sony's Readers models 550 and 700) or novel e-readers by following the provided EBONI-based procedure and focusing on INEX content. Participants may then gather and analyse results according to the EBONI approach and submit these for overall comparison and evaluation. The evaluation is task-oriented in nature. Participants are able to tailor their own evaluation experiments, inside the EBONI framework, according to resources available to them. In order to gather user feedback, participants can choose from a variety of methods, from low-effort online questionnaires to more time consuming one to one interviews, and think aloud sessions.

6.1 Task Setup

Participation requires access to one or more software/hardware e-readers (already on the market or in prototype version) that can be fed with a subset of the INEX book corpus (maximum 100 books), selected based on participants' needs and objectives. Participants are asked to involve a minimum sample of 15/20 users to complete 3-5 growing complexity tasks and fill in a customised version of the EBONI subjective questionnaire, allowing to gather meaningful

and comparable evidence. Additional user tasks and different methods for gathering feedback (e.g., video capture) may be added optionally. A crib sheet is provided to participants as a tool to define the user tasks to evaluate, providing a narrative describing the scenario(s) of use for the books in context, including factors affecting user performance, e.g., motivation, type of content, styles of reading, accessibility, location and personal preferences.

Our aim is to run a comparable but individualized set of studies, all contributing to elicit user and usability issues related to eBooks and e-reading.

7 Conclusions and Plans

This paper presents an overview of the INEX 2011 Books and Social Search Track. The track has four tasks: 1) Social Search for Best Books, 2) Prove It, 3) Structure Extraction, and 4) Active Reading Task.

This was the first year for The Social Search for Best Books (SB) task, but the amount of activity and the results promise a bright future for this task. The comparison of the LT forum suggestions and the relevance judgements from the Mechanical Turk experiment show that forum suggestions represent a different task from traditional ad hoc topical relevance search. The two sets of judgements give us an interesting data set to address questions about the relative value of professional controlled metadata and user-generated content for book search, for subject search topics as well as more recommendation oriented topics.

Preliminary analysis of the crowdsourcing data suggests that assessors treat topical relevance and recommendation similarly. If they consider book topically relevant, they often recommend it and vice versa, do not recommend when it is not on the right topic. Next year, we want to focus more specifically on the various aspects of relevance for book suggestions, such as topical relevance, recommendation, reading level, engagement etc.

This year the Prove It task continued unchanged with respect to last year. The number of participants for the PI task was low. We gathered relevance judgements from participants and from Mechanical Turk based on top 10 pools. The Mechanical Turk experiment is still running, but preliminary results show that the runs submitted by the University of Massachusetts Amherst leave little room for improvement. We will introduce new interesting challenges in the Prove It task for next year. One idea is to require systems to indicate whether a page contains confirming or refuting information.

In 2011, the SE track was run conjointly within the ICDAR conference for the second time. This effort gave way to the gathering and addition of 513 new annotated book ToCs to the previous 527, available for download on the track's Web site. The SE task will be run again at ICDAR 2013.

The ART was offered as last year. The task has so far only attracted 2 groups, none of whom submitted any results at the time of writing.

References

[1] Beckers, T., Fuhr, N., Pharo, N., Nordlie, R., Fachry, K.N.: Overview and Results of the INEX 2009 Interactive Track. In: Lalmas, M., Jose, J., Rauber, A., Sebastiani, F., Frommholz, I. (eds.) ECDL 2010. LNCS, vol. 6273, pp. 409–412. Springer, Heidelberg (2010)

[2] Déjean, H., Meunier, J.-L.: Reflections on the INEX structure extraction competition. In: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, DAS 2010, pp. 301–308. ACM, New York (2010)

[3] Doucet, A., Kazai, G., Dresevic, B., Uzelac, A., Radakovic, B., Todic, N.: Setting up a competition framework for the evaluation of structure extraction from ocr-ed books. International Journal of Document Analysis and Recognition (IJDAR), Special Issue on Performance Evaluation of Document Analysis and Recognition Algorithms 14(1), 45–52 (2011)

[4] Doucet, A., Kazai, G., Meunier, J.-L.: ICDAR 2011 Book Structure Extraction Competition. In: Proceedings of the Eleventh International Conference on Document Analysis and Recognition (ICDAR 2011), Beijing, China, pp. 1501–1505 (September 2011)

[5] Kazai, G., Kamps, J., Koolen, M., Milic-Frayling, N.: Crowdsourcing for book search evaluation: Impact of hit design on comparative system ranking. In: Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 205–214. ACM Press, New York (2011)

[6] Kazai, G., Koolen, M., Kamps, J., Doucet, A., Landoni, M.: Overview of the INEX 2010 Book Track: Scaling up the Evaluation Using Crowdsourcing. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 98–117. Springer, Heidelberg (2011)

[7] Svenonius, E.: The Intellectual Foundation of Information Organization. MIT Press (2000)

[8] Wilson, R., Landoni, M., Gibb, F.: The web experiments in electronic textbook design. Journal of Documentation 59(4), 454–477 (2003)

The Importance of Document Ranking and User-Generated Content for Faceted Search and Book Suggestions

Frans Adriaans[1,2], Jaap Kamps[1,3], and Marijn Koolen[1]

[1] Archives and Information Studies, Faculty of Humanities, University of Amsterdam
[2] Department of Psychology, University of Pennsylvania
[3] ISLA, Faculty of Science, University of Amsterdam

Abstract. In this paper we describe our participation in INEX 2011 in the Books and Social Search Track and the Data Centric Track. For the Books and Social Search Track we focus on the impact of different document representations of book metadata for book search, using either professional metadata, user-generated content or both. We evaluate the retrieval results against ground truths derived from the recommendations in the LibraryThing discussion groups and from relevance judgements obtained from Amazon Mechanical Turk. Our findings show that standard retrieval models perform better on user-generated metadata than on professional metadata. For the Data Centric Track we focus on the selection of a restricted set of facets and facet values that would optimally guide the user toward relevant information in the Internet Movie Database (IMDb). We explore different methods for effective result summarisation by means of weighted aggregation. These weighted aggregations are used to achieve maximal coverage of search results, while at the same time penalising overlap between sets of documents that are summarised by different facet values. We found that weighted result aggregation combined with redundancy avoidance results in a compact summary of available relevant information.

1 Introduction

Our aim for the Books and Social Search Track was to look at the relative value of user tags and reviews and traditional book metadata for ranking book search results. The Social Search for Best Books task is newly introduced this year and uses a large catalogue of book descriptions from Amazon and LibraryThing. The descriptions are a mix of traditional metadata provided by professional cataloguers and indexers and user-generated content in the form of ratings, reviews and tags.

Because both the task and collection are new, we keep our approach simple and mainly focus on a comparison of different document representations. We made separate indexes for representations containing a) only title information, b) all the professional metadata, c) the user-generated metadata, d) the metadata from Amazon, e) the data from LibraryThing and f) all metadata. With these indexes

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 30–44, 2012.

we compare standard language model retrieval systems and evaluate them using the relevance judgements from the LibraryThing discussion forums and from Amazon Mechanical Turk. We break down the results to look at performance on different topic types and genres to find out which metadata is effective for particular categories of topics.

For the Data Centric Track we focus on the selection of a restricted set of facets and facet values that would optimally guide the user toward relevant information. We aim to improve faceted search by addressing two issues: a) weighted result aggregation, and b) redundancy avoidance.

The traditional approach to faceted search is to summarise search results by providing counts of the number of documents that are associated with different facet values [6, 14]. Those facet values that have the highest number of counts are returned to the user. We extend this approach by exploring the aggregation of results using weighted document counts. The underlying intuition is that facet values with the *most* documents are not necessarily the *most relevant* values [2]. That is, buying a dvd by the director who directed the most movies does not necessarily meet the search demands of a user. It may be more suitable to return directors who made a large number of important (and/or popular) movies. More sophisticated result aggregations, acknowledging the importance of an entity, may thus provide better hints for further faceted navigation than simple document counts. We therefore explore different methods for effective result summarisation by means of weighted aggregation.

Another problem in faceted search concerns the avoidance of overlapping facets [8]. That is, facets whose values describe highly similar set of documents should be avoided. We therefore aim at penalising overlap between sets of documents that are summarised by different facet values. We expect that weighted result aggregation combined with redundancy avoidance results in a compact summary of the available relevant information.

We describe our experiments and results for the Books and Social Search Track in Section 2 and for the Data Centric Track in Section 3. In Section 4, we discuss our findings and draw conclusions.

2 Book Track

In the INEX 2011 Books and Social Search Track we participated in the Social Search for Best Books task. Our aim was to investigate the relative importance of professional and user-generated metadata. The document collection consists of 2.8 million book description, with each description combining information from Amazon and LibraryThing. The Amazon data has both traditional book metadata such as title information, subject headings and classification numbers, and user-generated metadata as well as user ratings and reviews. The data from LibraryThing consists mainly of user tags.

Professional cataloguers and indexers aim to keep metadata mostly objective. Although subject analysis to determine headings and classification codes is somewhat subjective, the process follows a formal procedure and makes use of

controlled vocabularies. Readers looking for interesting or fun books to read may not only want objective metadata to determine what book to read or buy next, but also opinionated information such as reviews and ratings. Moreover, subject headings and classification codes might give a very limited view of what a book is about. LibraryThing users tag books with whatever keywords they want, including personal tags like *unread* or *living room bookcase*, but also highly specific, descriptive tags such *WWII pacific theatre* or *natives in Oklahoma*.

We want to investigate to what extent professional and user-generated metadata provide effective indexing terms for book retrieval. The Cranfield tests [4] showed that using natural language terms from documents for indexing was at least as effective for retrieval as using controlled vocabularies. However, controlled vocabularies still hold the potential to improve completeness and accuracy of search results by providing consistent and rigorous index terms and ways to deal with synonymy and homonymy [7, 13]. [5] found that "if subject headings were to be removed from or no longer included in catalog records, users performing keyword searches would miss more than one third of the hits they currently retrieve." Authors, indexers and searchers all have different vocabularies [3] which, when all used in a single search process, may very well lead to the possibility of term mismatches. Bates [1, p.7] states that users of library catalogues prefer to use keyword search, which often does not match the appropriate subject headings.

One of the interesting aspects of user-generated metadata in this respect is that it has a smaller gap with the vocabulary of searchers [9]. User tags may (partially) compensate for missing subject headings. Yi and Chan [16] explored the possibility of mapping user tags from folksonomies to Library of Congress subject headings (LCSH), and found that with word matching, they could link two-thirds of all tags to LC subject headings. [10] looked at the retrieval effectiveness of tags taking into account the tag frequency. They found that the tags with the highest frequency are the most effective.

2.1 Experimental Setup

We used Indri [12] for indexing, removed stopwords and stemmed terms using the Krovetz stemmer. We made 5 separate indexes:

Full: the whole description is indexed.
Amazon: only the elements derived from the Amazon data are indexed.
LT: only the elements derived from the LibraryThing data are indexed.
Title: only the title information fields (title, author, publisher, publication date, dimensions, weight, number of pages) are indexed.
Professional: only the traditional metadata fields from Amazon are indexed, including the title information (see Title index) and classification and subject heading information.
Social: only the user-generated content such as reviews, tags and ratings are indexed.

Table 1. Evaluation results for the Social Search for Best Books task runs using the LT suggestion Qrels. Runs marked with * are official submissions.

Run	nDCG@10	P@10	MRR	MAP
xml_amazon.fb.10.50	0.2665	0.1730	0.4171	0.1901
*xml_amazon	0.2411	0.1536	0.3939	0.1722
*xml_full.fb.10.50	0.2853	0.1858	0.4453	0.2051
*xml_full	0.2523	0.1649	0.4062	0.1825
xml_lt.fb.10.50	0.1837	0.1237	0.2940	0.1391
*xml_lt	0.1592	0.1052	0.2695	0.1199
xml_prof	0.0720	0.0502	0.1301	0.0567
*xml_social.fb.10.50	**0.3101**	**0.2071**	**0.4811**	**0.2283**
*xml_social	0.2913	0.1910	0.4661	0.2115
xml_title	0.0617	0.0403	0.1146	0.0563

The topics are taken from the LibraryThing discussion groups and contain a *title* field which contains the title of a topic thread, a *group* field which contains the discussion group name and a *narrative* field which contains the first message from the topic thread. In our experiments we only used the *title* fields as queries and default settings for Indri (Dirichlet smoothing with $\mu = 2500$). We submitted the following six runs:

xml_amazon: a standard LM run on the Amazon index.
xml_full: a standard LM run on the Full index.
xml_full.fb.10.50: a run on the Full index with pseudo relevance feedback using 50 terms from the top 10 results.
xml_lt: a standard LM run on the LT index.
xml_social: a standard LM run on the Social index.
xml_social.fb.10.50: a run on the Social index with pseudo relevance feedback using 50 terms from the top 10 results.

Additionally we created the following runs:

xml_amazon.fb.10.50: a standard LM run on the Amazon index.
xml_lt.fb.10.50: a standard LM run on the LT index.
xml_prof: a standard LM run on the Professional index.
xml_title: a standard LM run on the Title index.

2.2 Results

The Social Search for Best Books task has two sets of relevance judgements. One based on the lists of books that were suggested on the LT discussion groups, and one based on document pools of the top 10 results of all official runs, judged by Mechanical Turk workers. For the latter set of judgements, a subset of 24 topics was selected from the larger set of 211 topics from the LT forums.

Table 2. Evaluation results for the Social Search for Best Books task runs using the AMT Qrels. Runs marked with * are official submissions.

Run	nDCG@10	P@10	MRR	MAP
xml_amazon.fb.10.50	0.5954	0.5583	0.7868	0.3600
*xml_amazon	**0.6055**	**0.5792**	0.7940	0.3500
*xml_full.fb.10.50	0.5929	0.5500	**0.8075**	**0.3898**
*xml_full	0.6011	0.5708	0.7798	0.3818
xml_lt.fb.10.50	0.4281	0.3792	0.7157	0.2368
*xml_lt	0.3949	0.3583	0.6495	0.2199
xml_prof	0.1625	0.1375	0.3668	0.0923
*xml_social.fb.10.50	0.5425	0.5042	0.7210	0.3261
*xml_social	0.5464	0.5167	0.7031	0.3486
xml_title	0.2003	0.1875	0.3902	0.1070

We first look at the results based on the Qrels derived from the LT discussion groups in Table 1. The runs on the Social index outperform the others on all measures. The indexes with no user-generated content–Professional and Title—lead to low scores. The user-provided content seems to add more useful information to the title fields than the professional metadata. The LT index also leads to better performance than the Professional index, suggesting tags can indeed compensate and improve upon controlled subject access. The indexes that have reviews—Amazon, Full and Social—outperform the LT index which has user tags but no reviews. Reviews seem to be effective document representations. Feedback is effective on the four indexes Amazon, Full, LT and Social.

Next we look at the results based on the Mechanical Turk judgements in Table 2. Here we see a different pattern. With the top 10 results judged on relevance, all scores are higher than with the LT judgements. This is probably due in part to the larger number of judged documents, but perhaps also to the difference in the tasks. The Mechanical Turk workers were asked to judged the topical relevance of books—is the book on the same topic as the request from the LT forum—whereas the LT forum members were asked by the requester to recommend books from a possibly long list of topically relevant books. Another interesting observation is that feedback is not effective for the AMT evaluation on the Full, Amazon and Social indexes, whereas it was effective for the LT evaluation. The main difference between the Full, Amazon and Social indexes on the one hand and the LT index on the other hand is that the LT index has no reviews. This might suggest the AMT workers paid more attention to the tags than to the reviews when making their judgements. A rationale for this could be that tags provide a faster way to judge a book than reviews, which is in the interest of workers who wish to minimise the time spent on a HIT.

Perhaps another reason is that the two evaluations use different topic sets. To investigate the impact of the topic set, we filtered the LT judgements on the 24 topics selected for AMT, such that the LT and AMT judgements are more directly comparable. The results are shown in Table 3. The pattern is similar

Table 3. Evaluation results for the Social Search for Best Books task runs using the LT recommendation Qrels for the 24 topics selected for the AMT experiment. Runs marked with * are official submissions.

Run	nDCG@10	P@10	MRR	MAP
xml_amazon.fb.10.50	0.2103	0.1625	0.3791	0.1445
xml_amazon	0.1941	0.1583	0.3583	0.1310
xml_full.fb.10.50	0.2155	0.1708	0.3962	0.1471
xml_full	0.1998	0.1625	0.3550	0.1258
xml_lt.fb.10.50	0.1190	0.0833	0.3119	0.0783
xml_lt	0.1149	0.0708	0.3046	0.0694
xml_prof	0.0649	0.0500	0.1408	0.0373
xml_social.fb.10.50	**0.3112**	**0.2333**	**0.5396**	**0.1998**
xml_social	0.2875	0.2083	0.5010	0.1824
xml_title	0.0264	0.0167	0.0632	0.0321

to that of the LT judgements over the 211 topics, indicating that the impact of the topic set is small. The runs on the Social index outperform the others, with the Amazon and Full runs scoring better than the LT runs, which in turn perform better than the Official and Title runs. Feedback is again effective for all reported measures. In other words, the observed difference between the LT and AMT evaluations is not caused by difference in topics but probably caused by the difference in the tasks.

2.3 Analysis

The topics of the SB Track are labelled with topic type and genre. There are 8 different type labels: *subject* (134 topics), *author* (32), *genre* (17), *series* (10), *known-item* (7), *edition* (7), *work* (3) and *language* (2). The genre labels can be grouped into fiction, with genre label *Literature* (89 topics) and non-fiction, with genre labels such as *history* (60 topics), *biography* (24), *military* (16), *religion* (16), *technology* (14) and *science* (11).

We break down the evaluation results over topic types and take a closer look at the *subject*, *author* and *genre* types. The other types have either very small numbers of topics (*work* and *language*), or are hard to evaluate with the current relevance judgements. For instance, the *edition* topics ask for a recommended edition of a particular work. In the relevance judgements the multiple editions of a work are all mapped to a single work ID in LibraryThing. Some books have many more editions than others, which would create in imbalance in the relevance judgements for most topics.

The evaluation results are shown in Table 4. For most runs there is no big difference in performance between *fiction* and *non-fiction* topics, with slightly better performance on the *fiction* topics. For the two runs on the Social index the difference is bigger. Perhaps this is due to a larger amount of social metadata for fiction books. The standard run on the LT index (xml_lt) performs better on

Table 4. Evaluation results using the LT recommendation Qrels across different topic genres and types. Runs marked with * are official submissions.

	nDCG@10				
Run	**Fiction**	**Non-fiction**	**Subject**	**Author**	**Genre**
xml_amazon.fb.10.50	0.2739	0.2608	0.2203	0.4193	0.0888
*xml_amazon	0.2444	0.2386	0.1988	0.3630	0.0679
*xml_full.fb.10.50	0.2978	0.2765	0.2374	0.4215	0.1163
*xml_full	0.2565	0.2491	0.2093	0.3700	0.0795
xml_lt.fb.10.50	0.1901	0.1888	0.1597	0.2439	0.0850
*xml_lt	0.1535	0.1708	0.1411	0.2093	0.0762
xml_prof	0.0858	0.0597	0.0426	0.1634	0.0225
*xml_social.fb.10.50	**0.3469**	**0.2896**	**0.2644**	**0.4645**	0.1466
*xml_social	0.3157	0.2783	0.2575	0.4006	**0.1556**
xml_title	0.0552	0.0631	0.0375	0.1009	0.0000

the non-fiction topics, suggesting the tags for non-fiction are more useful than for fiction books. Among the topic types we see the same pattern across all measures and all runs. The *author* topic are easier than the *subject* topics, which are again easier than the *genre* topics. We think this is a direct reflection of the clarity and specificity of the information needs and queries. For author related topics, the name of the author is a very clear and specific retrieval cue. Subject are somewhat broader and less clearly defined, making it harder to retrieve exactly the right set of books. For genre-related topics it is even more difficult. Genres are broad and even less clearly defined. For many genres there are literally (tens of) thousands of books and library catalogues rarely go so far in classifying and indexing specific genres. This is also reflected by the very low scores of the Official and Title index runs for *genre* topics.

3 Data Centric Track

For the Data Centric Track we participated in the Ad Hoc Task and the Faceted Search Task. Our particular focus was on the Faceted Search Task where we aim to discover for each query a restricted set of facets and facet values that best describe relevant information in the results list. Our general approach is to use weighted result aggregations to achieve maximal coverage of relevant documents in IMDb. At the same time we aim to penalise overlap between sets of documents that are summarised by different facet values. We expect that this results in a compact summary of the available relevant information. Below we describe our setup and results.

3.1 Experimental Setup

We use Indri [12] with Krovetz stemming and default smoothing (Dirichlet with $\mu = 2500$) for indexing. All XML leaf elements in the IMDb collection are indexed

as fields. Documents were retrieved using title fields only. The maximum number of retrieved documents was set to 1000 (Ad Hoc Task) and 2000 (Faceted Search Task). We submitted one run for the Ad Hoc Search Task and three runs for the Faceted Search Task.

Ad Hoc Task: One run was generated using the settings described above: *UAms2011adhoc*.

Faceted Search Task: Two Ad Hoc result files were used as a basis for facet selection: the *2011-dc-lucene.trec* file provided by the INEX organisation, and an Ad Hoc run that was created using Indri. The maximum number of results for this run was set to 2000. We submitted three Faceted Search runs: *UAms2011indri-c-cnt*, *UAms2011indri-cNO-scr2*, *UAms2011lucene-cNO-lth*. In each run, a hierarchy of recommended facet values is constructed for each topic. A path through the hierarchy represents an accumulated set of conditions on the retrieved documents. The search results become more refined at every step, and the refinement ultimately narrows down to a set of potentially interesting documents.

3.2 Facet Selection

The set of candidate facets consists of all numerical and categorical fields in the IMDb collection. The goal is to select useful facets (and values) from the set of candidate facets.

Result Aggregation. We explored two different methods of weighted result aggregation. The first method aggregates document lengths rather than number of documents. Since popular movies in IMDb have larger entries (which we measure by file size), we assume that document lengths push facet values associated with popular movies to the top of the ranked set of facet values. The second method aggregates documents from the Ad Hoc run by summing retrieval scores. The idea is that higher-ranked documents display facet values that are most likely to be of interest to the user. Note that document length is a static ('global') measure of document importance, whereas retrieval scores are dynamic ('local'), resulting in different degrees of importance for different topics. We compare both methods to traditional non-weighted aggregation of search results using document counts. The result aggregations form the basis of facet selection.

Coverage. For facet selection we use the intuition that facets which provide compact summaries of the available data allow fast navigation through the collection. This intuition was implemented as *facet coverage*: the number of documents that are summarised by a facet's top n values. Two types of coverage were implemented. The first version, *coverage*, sums up the (weighted) document counts that are associated with the facet's top n values. A potential pitfall of this approach is that this method favours redundancy. That is, the sets of documents that are associated with different facet values may have a high degree of overlap. For example, the keywords 'murder' and 'homicide' may point to almost identical sets of documents. We assume a user wants compact overviews

Table 5. Selected facets and values for the query 'Vietnam' (topic 2011205). Facets are ranked by coverage based on document counts.

Rank	Coverage	Facet	Top-5 values
1	945	genre	Drama (306) Documentary (207) War (199) Action (157) Comedy (76)
2	850	keyword	vietnam (286) vietnam-war (220) independent-film (162) vietnam-veteran (110) 1960s (72)
3	477	language	English (400) Vietnamese (42) French (16) Spanish (10) German (9)
4	437	country	USA (345) UK (30) Canada (27) France (19) Vietnam (16)
5	397	color	Color (291) Color - (Technicolor) (45) Black and White (40) Color - (Eastmancolor) (11) Color - (Metrocolor) (10)

of different, non-overlapping sets of documents that may be of interest to the searcher. Therefore, we implemented a second version: *coverageNO* ('coverage, no overlap') counts the number of unique documents that are summarised by the facet's top n values. As a consequence, redundancy in facet values is penalised.

Coverage-based facet selection is applied recursively. Starting with the complete set of Ad Hoc results (corresponding to the root node of the facet hierarchy), the facet with the highest coverage is chosen. The set of results is then narrowed down to the set of documents that are covered by this facet. In this new set, a second facet is chosen with the highest coverage. This selection process continues until a specified number of facets has been selected. We apply facet selection to movie facets and person facets independently, since these facets describe different types of documents (i.e., you cannot drill-down into person files after you have narrowed down the results using a movie facet). An example of a ranked set of movie facets for the query 'Vietnam' is given in Table 5.

3.3 Path Construction

The facet hierarchy is based on the selected set of facets and corresponding top n ranked values. Each path starts with a value from the first facet, followed by a value from the second facet, etc. The paths are ordered by rankings of the values within a facet. Not all logically possible paths are considered relevant. As a formal criterium, we assume that only paths leading to between 10 and 20 documents are useful recommendations for the user. Paths that lead to fewer documents are deemed too specific. Paths to a larger number of documents are deemed too general, and the system will attempt to branch into a deeper, more specific level. We generate trees for 'movies' and 'persons' independently and join them in the order of the largest number of paths. (For most queries there were more movie paths than person paths.) As an example, we display a partial tree corresponding to the query 'Vietnam', using the facets from Table 5:

```
<topic tid="2011205">
<fv f="/movie/overview/genres/genre" v="Drama">
  <fv f="/movie/overview/keywords/keyword" v="vietnam">
    <fv f="/movie/additional_details/languages/language" v="Vietnamese">
      <fv f="/movie/additional_details/countries/country" v="USA">
        <fv f="/movie/additional_details/colors/color" v="Color"/>
          </fv>
        </fv>
      </fv>
  <fv f="/movie/overview/keywords/keyword" v="vietnam-war">
    <fv f="/movie/additional_details/languages/language" v="English">
      <fv f="/movie/additional_details/countries/country" v="USA">
        <fv f="/movie/additional_details/colors/color" v="Color - (Technicolor)"/>
          </fv>
        </fv>
    <fv f="/movie/additional_details/languages/language" v="Vietnamese">
      <fv f="/movie/additional_details/countries/country" v="USA"/>
        </fv>
      </fv>
  <fv f="/movie/overview/keywords/keyword" v="vietnam-veteran">
    <fv f="/movie/additional_details/languages/language" v="English">
      <fv f="/movie/additional_details/countries/country" v="USA">
        <fv f="/movie/additional_details/colors/color" v="Color - (Technicolor)"/>
          </fv>
        </fv>
      </fv>
<fv f="/movie/overview/genres/genre" v="Documentary">
  <fv f="/movie/overview/keywords/keyword" v="vietnam">
    <fv f="/movie/additional_details/languages/language" v="English">
      <fv f="/movie/additional_details/countries/country" v="USA">
        <fv f="/movie/additional_details/colors/color" v="Black and White"/>
...
```

Table 6. Experimental parameters. The values of the first three parameters were combined to generate a total of 2x4x2 = 16 different runs. The other parameters (4-7) were kept constant.

Parameter	Values
1. Ad hoc input	Indri, Lucene
2. Document weights	count (cnt), length (lth), score (scr), score^2 (scr2)
3. Selection method	coverage (c), coverageNO (cNO)
4. Number of facets	5
5. Number of values	5
6. Min. number of path results	10
7. Max. number of path results	20

3.4 The Faceted Seach Runs

We generated a total of 16 runs by varying the parameters listed in Table 6. From this set, three runs were selected for submission to the INEX workshop:

UAms2011indri-c-cnt: This is our baseline run which implements the standard approach of selecting those facet values that summarize the largest number of documents.

UAms2011indri-cNO-scr2: This run uses weighted result aggregation (using retrieval scores, in contrast to unranked aggregation in the baseline run). This run also penalises overlap between document sets that correspond to different facet values.

UAms2011lucene-cNO-lth: The third run uses the Lucene reference results file that was provided by INEX. The run uses weighted result aggregation based on document lengths (file sizes, as opposed to retrieval scores).

3.5 Results and Discussion

Our run for the Ad Hoc Task was the best scoring run out of a total of 35 submitted runs by 9 different institutes, with a MAP of 0.3969 [15]. The success of our Ad Hoc run indicates that indexing the complete XML structure of IMDb is not necessary for effective document retrieval. It appears, at least for the Ad Hoc case, that it suffices to index leaf elements.

The runs for the Faceted Search Task were evaluated with respect to two different metrics. The first measure assesses the effectiveness of a faceted system by calculating the interaction cost. This is defined as the number of results, facets, or facet values that the user examines before encountering the first relevant result. The measure is referred to as the Normalised Gain (NG), and the Average Normalised Gain (see [15] for more details). The second measure is the Normalised Discounted Cumulated Gain (nDCG), which assesses the relevance of a hierarchy of facet values based on the relevance of the results that are associated with the values [11].

Table 7. Evaluation results for Faceted Search runs in terms of NGs and ANG

topic	UAms2011indri-c-cnt	UAms2011indri-cNO-scr2	UAms2011lucene-cNO-lth
201	**0.64**	0.60	-
202	0	0	**0.21**
203	0	0	0
204	0.63	0.75	**0.94**
205	0	0	**0.81**
207	0	**0.77**	0
208	0	0	0
209	0	0	0
210	**0.75**	0.74	-
211	0.18	0	**0.53**
212	**0.89**	0.88	-
213	**0.76**	**0.76**	-
214	0	0	**0.64**
ANG	0.30	**0.35**	0.24

Note. c = coverage, cNO = coverage with no overlap, cnt = count, lth = length, scr = retrieval score, scr2 = retrieval score^2. Best scores are in bold.

Table 7 shows the NG and ANG scores of the three runs that we submitted for the Faceted Search Task. While the results vary substantially between topics, our UAms2011indri-cNO-scr2 run (which uses retrieval scores as document weights, and penalises facet values with overlapping sets of documents) has a higher overall score than our baseline run (UAms2011indri-c-cnt). This confirms our expectation that faceted search can be improved by exploiting information from the Ad Hoc results list, and by penalising redundancy. The two Indri-based runs outperform the Lucene-based run (which uses document lengths as weights). The superior performance could thus be due to two factors: the underlying Ad Hoc run, or the aggregation method. Our run UAms2011indri-cNO-scr2 had the highest ANG score out of the 12 runs that had been submitted to the workshop by 5 different groups [15]. Most other groups used the Lucene reference result file, so, again, it is possible that the superior performance of our run is due to a better underlying results file, rather than to effective facet selection. We therefore examine a larger set of runs, allowing us to analyse the results in a more systematic way.

Table 8 shows the nDCG scores for all of our runs (including the three runs that we had submitted).[1] The nDCG scores confirm that the UAms2011indri-cNO-scr2 run was our best one, and the run has a higher mean nDCG than any of the runs that had been submitted by other participating groups (as reported in

[1] Out of all 16 different runs described in Table 6 only 12 produced positive results on the nDCG metric.

Table 8. Evaluation results for the Faceted Search runs in terms of nDCG

	Indri							Lucene				
topic	c cnt	c lth	c scr	c scr2	cNO cnt	cNO lth	cNO scr2	c cnt	c scr2	cNO lth	cNO scr	cNO scr2
201	**0.035**	**0.035**	**0.035**	**0.035**	**0.035**	**0.035**	**0.035**	0	0	0	0	0
202	0	0	0	0	0	0	0	0	0	0	0	0
203	0	0	0	0	0	0	0	0	0	0	0	0
204	0	0	0	0	0	0	0	0	0	0	0	0
205	0	0	0	0	**0.429**	0.198	**0.429**	0	0	0.215	0.209	0.066
207	0	0	0	0	0	0	**0.162**	0	0	0	0	0
208	0	0	0	0	**0.455**	0.452	**0.455**	0	0	0	0	0
209	0	0	0	0	0	0	0	0	0.360	0	0.360	0.360
210	0	0	0	0	0	0	0	0	0	0	0	0
211	0	0	0	0	0	0	0	0	0	0	0	0
212	0	0	0	0	0	0	0	0	0	0	0	0
213	0	0	0	0	0	0	0	0	0	0	0	0
214	**0.185**	0	0.160	0	**0.185**	**0.185**	**0.185**	0.091	0	0.091	0.091	0.022
mean	0.017	0.003	0.015	0.003	0.085	0.067	**0.097**	0.007	0.028	0.024	0.051	0.034

Note. c = coverage, cNO = coverage with no overlap, cnt = count, lth = length, scr = retrieval score, scr2 = retrieval score^2. Best scores are in bold.

[15]).[2] The nDCG results show that the Indri results file indeed provided a better basis for the selection of facet values than the Lucene reference file. However, if we compare different runs that are based on the same Lucene results file, we find that retrieval scores (scr and scr2) improve performance as compared to the Lucene run that we submitted (which was based on document length). Although we have to be careful with interpreting these results where scores for most topics are zero, the success of our run seems to be due to both the results file and the aggregation method. Moreover, the results indicate that our method for penalising overlapping facet values was effective: the *coverage, no overlap* runs had higher nDCG means than their *coverage* counterparts. Result aggregation using retrieval scores proved to be especially useful in combination with the overlap penalty.

In sum, the results confirm our expectation that weighted result aggregation combined with redundancy avoidance results in a compact summary of available relevant information. The findings show the importance of good Ad Hoc results as a basis for faceted search (the well-known 'garbage in, garbage out' principle), and the importance of penalising redundancy in different facet values. Finally, while the effect of different result aggregations varies, it seems that retrieval scores are useful for the detection of relevant facet values.

[2] The mean nDCG score of our run is still quite low, and the fact that many topics yielded NDGC = 0 suggests that either the topic set, the collection and/or the metric may have been inappropriate for the evaluation of faceted search systems.

4 Conclusion

In this paper we discussed our participation in the INEX 2011 Books and Social Search Track and the Data Centric Track.

In the Books and Social Search Track we participated in the Social Search for Best Books task and focused on comparing different document representations based on professional metadata and user-generated metadata. Our main finding is that standard language models perform better on representations of user-generated metadata than on representations of professional metadata.

In our result analysis we differentiated between topics requesting fiction and non-fiction books and between subject-related topics, author-related topics and genre-related topics. Although the patterns are similar across topic types and genres, we found that social metadata is more effective for fiction topics than for non-fiction topics, and that regardless of document representation, all systems perform better on author-related topics than on subject related topics and worst on genre-related topics. We expect this is related to the specificity and clarity of these topic types. Author-related topics are highly specific and target a clearly defined set of books. Subject-related topics are broader and less clearly defined, but can still be specific. Genre-related topics are very broad—many genres have tens of thousands of books—and are also more vague information needs that are closer to exploratory search.

In future work we will look closer at the relative value of various types of metadata and directly compare individual types of metadata such as reviews, tags and subject headings. We will also look at the different search scenarios underlying the relevance judgements and topic categories, such as subject search, recommendation and exploratory search.

In the Data Centric Track we participated in the Ad Hoc and Faceted Search Task. Our main finding is that faceted search can be improved through aggregation of search results that are weighted by their Ad Hoc retrieval score, expressing the local importance of different documents in the results list. In addition, we found that avoiding redundancy leads to a more compact representation of the results list. Although the results are based on a small number of topics, weighted result aggregation and redundancy avoidance together seem to provide an effective means of creating a compact summary of available relevant information.

Acknowledgments. Frans Adriaans was supported by the Netherlands Organization for Scientific Research (NWO) grants # 612.066.513 and 446.010.-027. Jaap Kamps was supported by NWO under grants # 612.066.513, 639.-072.601, and 640.005.001. Marijn Koolen was supported by NWO under grant # 639.072.601.

References

[1] Bates, M.J.: Task Force Recommendation 2.3 Research and Design Review: Improving user access to library catalog and portal information. In: LoC Bicentennial Conf. on Bibliographic Control for the New Millennium (2003)

[2] Ben-Yitzhak, O., Golbandi, N., Har'El, N., Lempel, R.: Beyond basic faceted search. In: WSDM 2008 (2008)

[3] Buckland, M.: Vocabulary as a Central Concept in Library and Information Science. In: Digital Libraries: Interdisciplinary Concepts, Challenges, and Opportunities. Proceedings of CoLIS3 (1999)

[4] Cleverdon, C.W.: The Cranfield tests on index language devices. Aslib 19, 173–192 (1967)

[5] Gross, T., Taylor, A.G.: What Have We Got to Lose? The Effect of Controlled Vocabulary on Keyword Searching Results. College & Research Libraries 66(3) (2005)

[6] Hearst, M.A., Elliott, A., English, J., Sinha, R., Swearingen, K., Yee, K.-P.: Finding the flow in web site search. Communications of the ACM 45, 42–49 (2002)

[7] Lancaster, F.W.: Vocabulary control for information retrieval, 2nd edn. Information Resources Press, Arlington (1986)

[8] Li, C., Yan, N., Roy, S.B., Lisham, L., Das, G.: Facetedpedia: Dynamic generation of query-dependent faceted interfaces for wikipedia. In: Proceedings of WWW 2010 (2010)

[9] Mathes, A.: Folksonomies - cooperative classification and communication through shared metadata (December 2004), http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html

[10] Peters, I., Schumann, L., Terliesner, J., Stock, W.G.: Retrieval Effectiveness of Tagging Systems. In: Grove, A. (ed.) Proceedings of the 74th ASIS&T Annual Meeting, vol. 48 (2011)

[11] Schuth, A., Marx, M.: Evaluation Methods for Rankings of Facetvalues for Faceted Search. In: Forner, P., Gonzalo, J., Kekäläinen, J., Lalmas, M., de Rijke, M. (eds.) CLEF 2011. LNCS, vol. 6941, pp. 131–136. Springer, Heidelberg (2011)

[12] Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: a language-model based search engine for complex queries. In: Proceedings of the International Conference on Intelligent Analysis (2005)

[13] Svenonius, E.: Unanswered questions in the design of controlled vocabularies. JASIS 37(5), 331–340 (1986)

[14] Tunkelang, D.: Faceted Search. Morgan and Claypool Publishers (2009)

[15] Wang, Q., Ramírez, G., Marx, M., Theobald, M., Kamps, J.: Overview of the INEX 2011 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 118–137. Springer, Heidelberg (2012)

[16] Yi, K., Chan, L.M.: Linking folksonomy to Library of Congress subject headings: an exploratory study. Journal of Documentation 65(6), 872–900 (2009)

RSLIS at INEX 2011: Social Book Search Track

Toine Bogers[1], Kirstine Wilfred Christensen[2], and Birger Larsen[1]

[1] Royal School of Library and Information Science
Birketinget 6, 2300 Copenhagen, Denmark
{tb,blar}@iva.dk
[2] DBC, Tempovej 7, 2750 Ballerup, Denmark
kwc@dbc.dk

Abstract. In this paper, we describe our participation in the INEX 2011 Social Book Search track. We investigate the contribution of different types of document metadata, both social and controlled, and examine the effectiveness of re-ranking retrieval results using social features. We find that the best results are obtained using all available document fields and topic representations.

Keywords: XML retrieval, social tagging, controlled metadata, book recommendation.

1 Introduction

In this paper, we describe our participation in the INEX 2011 Social Book Search track [1]. Our goals for the Social Book Search task were (1) to investigate the contribution of different types of document metadata, both social and controlled; and (2) to examine the effectiveness of using social features to re-rank the initial content-based search results.

The structure of this paper is as follows. We start in Section 2 by describing our methodology: pre-processing the data, which document and topic fields we used for retrieval, and our evaluation. In Section 3, we describe the results of our content-based retrieval runs. Section 4 describes our use of social features to re-rank the content-based search results. Section 5 describes which runs we submitted to INEX, with the results of those runs presented in Section 6. We discuss our results and conclude in Section 7.

2 Methodology

2.1 Data and Preprocessing

In our experiments we used the Amazon/LibraryThing collection provided by the organizers of the INEX 2011 Social Book Search track. This collection contains XML representations of 2.8 million books, with the book representation data crawled from both Amazon.com and LibraryThing.

A manual inspection of the collection revealed the presence of several XML fields that are unlikely to contribute to the successful retrieval of relevant books.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 45–56, 2012.

Examples include XML fields like <image>, <listprice>, and <binding>. While it is certainly not impossible that a user would be interested only in books in a certain price range or in certain bindings, we did not expect this to be likely in this track's particular retrieval scenario of recommending books based on a topical request. We therefore manually identified 22 such fields and removed them from the book representations.

In addition, we converted the original XML schema into a simplified version. After these pre-processing steps, we were left with the following 19 content-bearing XML fields in our collection: <isbn>, <title>, <publisher>, <editorial>[1], <creator>[2], <series>, <award>, <character>, <place>, <blurber>, <epigraph>, <firstwords>, <lastwords>, <quotation>, <dewey>, <subject>, <browseNode>, <review>, and <tag>.

One of the original fields (<dewey>) contains the numeric code representing the Dewey Decimal System category that was assigned to a book. We replaced these numeric Dewey codes by their proper textual descriptions using the 2003 list of Dewey category descriptions[3] to enrich the controlled metadata assigned to each book. For example, the XML element <dewey>519</dewey> was replaced by the element <dewey>Probabilities & applied mathematics</dewey>.

2.2 Field Categories and Indexing

The 19 remaining XML fields in our collection's book representations fall into different categories. Some fields, such as <dewey> and <subject>, are examples of *controlled metadata* produced by LIS professionals, whereas other fields contains *user-generated metadata*, such as <review> and <tag>. Yet other fields contain 'regular' book metadata, such as <title> and <publisher>. Fields such as <quotation> and <firstwords> represent a book's content more directly.

To examine the influence of these different types of fields, we divided the document fields into five different categories, each corresponding to an index. In addition, we combined all five groups of relevant fields for an index containing all fields. This resulted in the following six indexes:

All fields. For our first index all-doc-fields we simply indexed all of the available XML fields (see the previous section for a complete list).

Metadata. In our metadata index, we include all metadata fields that are immutably tied to the book itself and supplied by the publisher: <title>, <publisher>, <editorial>, <creator>, <series>, <award>, <character>, and <place>.

Content. For lack of access to the actual full-text books, we grouped together all XML fields in the content index that contain some part of the book text: blurbs, epigraphs, the first and last words, and quotations. This corresponded

[1] Our <editorial> fields contain a concatenation of the original <source> and <content> fields for each editorial review.

[2] For our <creator> field, we disregard the different roles the creators could have in the original XML schema and simply treat all roles the same.

[3] Available at http://www.library.illinois.edu/ugl/about/dewey.html

to indexing the fields <blurber>, <epigraph>, <firstwords>, <lastwords>, and <quotation>.

Controlled Metadata. In our controlled-metadata index, we include the three controlled metadata fields curated by library professionals: <browseNode>, <dewey>, and <subject>.

Tags. We split the social metadata contained in the document collection into two different types: tags and reviews. For the tags index, we used the tag field, expanding the tag count listed in the original XML. For example, the original XML element <tag count="3">fantasy</tag> would be expanded as <tag>fantasy fantasy fantasy</tag>. This ensures that the most popular tags have a bigger influence on the final query-document matching.

Reviews. The user reviews from the <review> fields were indexed in two different ways: (1) all user reviews belonging to a single book were combined in a single document representation for that book, and (2) each book review was indexed and retrieved separately. The former book-centric review index reviews is used in Section 3; the latter review-centric index reviews-split is used in our social re-ranking approach described in Section 4.

We used the Indri 5.0 retrieval toolkit[4] for indexing and retrieval. We performed stopword filtering on all of our indexes using the SMART stopword list, and preliminary experiments showed that using the Krovetz stemmer resulted in the best performance. Topic representations were processed in the same manner.

2.3 Topics

As part of the INEX 2011 Social Book Search track two set of topics were released with requests for book recommendations based on textual description of the user's information need: a training set and a test set. Both topic sets were extracted from the LibraryThing forum. The training set consisted of 43 topics and also contained relevance judgments, which were crawled from the LibraryThing forum messages. Each book that was mentioned in the forum thread was deemed relevant, meaning these could possibly be incomplete or inaccurate. Despite these known limitations, we used the training set to optimize our retrieval algorithms in the different runs. The results we report in Sections 3 and 4 were obtained using this training set.

The test set containing 211 topics is the topic set used to rank and compare the different participants' systems at INEX. The results listed in Section 6 were obtained on this test set.

Each topic in the two sets are represented by several different fields, with some fields only occurring in the test set. In our experiments with the training and the test set, we restricted ourselves to automatic runs using the following three fields (partly based on a manual inspection of their usefulness for retrieval):

Title. The <title> field contains the title of the forum topic and typically provide a concise description of the information need. Runs that only use the topic title are referred to as title.

[4] Available at http://www.lemurproject.org/

Group. The LibraryThing forum is divided into different groups covering different topics. Runs that only use the <group> field (i.e., the name of the LibraryThing group as query) are referred to as group.

Narrative. The first message of each forum topic, typically posted by the topic creator, describes the information need in more detail. This often contains a description of the information need, some background information, and possibly a list of books the topic creator has already read or is not looking for. The <narrative> field typically contains the richest description of the topic and runs using only this field are referred to as narrative.

All Topic Fields. In addition to runs using these three fields individually, we also performed runs with all three fields combined (all-topic-fields).

The test and training sets contained several other fields that we did not experiment with due to temporal constraints, such as <similar> and <dissimilar>. However, we list some of our ideas in Section 7.1.

2.4 Experimental Setup

In all our retrieval experiments, we used the language modeling approach with Jelinek-Mercer (JM) smoothing as implemented in the Indri 5.0 toolkit. We preferred JM smoothing over Dirichlet smoothing, because previous work has shown that for longer, more verbose queries JM smoothing performs better than Dirichlet smoothing [2], which matches the richer topic descriptions provided in the training and test sets.

For the best possible performance, we optimized the λ parameter, which controls the influence of the collection language model, with higher values giving more influence to the collection language model. We varied λ in steps of 0.1, from 0.0 to 1.0 using the training set of topics. We optimized λ separately for each combination of indexes and topic sets. For each topic we retrieve up 1000 documents and we used NDCG[5] as our evaluation metric [3].

3 Content-Based Retrieval

For our first round of experiments focused on a standard content-based retrieval approach where we compared the different index and the different topic representations. We had six different indexes (all-doc-fields, metadata, content, controlled-metadata, tags, and reviews) and four different sets of topic representations (title, group, narrative, and all-topic-fields). We examined each of these pairwise combinations for a total of 24 different content-based retrieval runs. Table 1 shows the best NDCG results for each run on the training set with the optimal λ values.

We can see several interesting results in Table 1. First, we see that the best overall content-based run used all topic fields for the training topics, retrieved

[5] Please note that the official evaluation on the test set used NDCG@10 as an evaluation metric instead of NDCG; we were not aware of this at the time of performing our experiments.

Table 1. Results of the 24 different content-based retrieval runs on the training set using NDCG as evaluation metric. Best-performing runs for each topic representation are printed in bold. The boxed run is the best overall.

Document fields	Topic fields			
	title	narrative	group	all-topic-fields
metadata	0.2756	0.2660	0.0531	0.3373
content	0.0083	0.0091	0.0007	0.0096
controlled-metadata	0.0663	0.0481	0.0235	0.0887
tags	0.2848	0.2106	0.0691	0.3334
reviews	**0.3020**	0.2996	0.0773	0.3748
all-doc-fields	0.2644	**0.3445**	**0.0900**	**0.4436**

against the index containing all document fields (all-doc-fields). In fact, for three out of four topic sets, using all-doc-fields provides the best performance. The book-centric reviews index is close second with strong performance on all four topic sets. Finally, we observe that the content and controlled-metadata indexes result in the worst retrieval performance across all four topic sets.

When we compare the different topic sets, we see that the all-topic-fields set consistently produces the best performance, followed by the title and narrative topic sets. The group topic set generally produced the worst-performing runs.

4 Social Re-ranking

The inclusion of user-generated metadata in the Amazon/LibraryThing collection gives the track participants the opportunity to examine the effectiveness of using social features to re-rank or improve the initial content-based search results. One such a source of social data are the tags assigned by LibraryThing users to the books in the collection. The results in the previous section showed that even when treating these as a simple content-based representation of the collection using our tags index, we can achieve relatively good performance.

In this section, we turn our attention to the book reviews entered by Amazon's large user base. We mentioned in Section 2.1 that we indexed the user reviews from the <review> fields in two different ways: (1) all user reviews belonging to a single book were combined in a single document representation for that book (reviews), and (2) each book review was indexed and retrieved separately (reviews-split). The results of the content-based runs in the previous section showed that a book-centric approach to indexing reviews provided good performance.

Review-Centric Retrieval. However, all user reviews are not equal. Some reviewers provide more accurate, in-depth reviews than others, and in some cases reviews may be even be misleading or deceptive. This problem of spam reviews on online shopping websites such as Amazon.com is well-documented [4].

This suggests that indexing and retrieving reviews individually and then aggregating the individually retrieved reviews could be beneficial by matching the best, most topical reviews against our topics.

Our review-centric retrieval approach works as follows. First, we index all reviews separately in our reviews-split index. We then retrieve the top 1000 individual reviews for each topic (i.e., this is likely to be a mixed of different reviews for different books). This can result in several reviews covering the same book occurring in our result list, which then need to be aggregated into a single relevance score for each separate book. This problem is similar to the problem of *results fusion* in IR, where the results of *different* retrieval algorithms on the *same* collection are combined. This suggest the applicability of standard methods for results fusion as introduced by [5]. Of the six methods they investigated, we have selected the following three for aggregating the review-centric retrieval results.

- The CombMAX method takes the maximum relevance score of a document from among the different runs. In our case, this means that for each book in our results list, we take the score of the highest-retrieved individual review to be the relevance score for that book.
- The CombSUM method fuses runs by taking the sum of the relevance scores for each document separately. In our case, this means that for each book in our results list, we take the sum of the relevance scores for all reviews referring to that particular book.
- The CombMNZ method does the same as the CombSUM method, but boost the sum of relevance scores by the number of runs that actually retrieved the document. In our case, this means that for each book in our results list, we take the sum of the relevance scores for all reviews referring to that particular book, and multiply that by the number of reviews that were retrieved for that book.

Helpfulness of Reviews. One of the more popular aspects of user reviewing process on Amazon.com is that reviews can be marked as helpful or not helpful by other Amazon users. By using this information, we could ensure that the most helpful reviews have a better chance of being retrieved. We can use this information to improve the retrieval results by assigning higher weights to the most helpful reviews and thereby boosting the books associated with those reviews. The assumption behind this is that helpful reviews will be more accurate and on-topic than unhelpful reviews.

We estimate the helpfulness of a review by dividing the number of votes for helpfulness by the total number of votes for that review. For example, a review that 3 out of 5 people voted as being helpful would have a helpfulness score of 0.6. For each retrieved review i we then obtain a new relevance score $score_{weighted}(i)$ by multiplying that review's original relevance score $score_{org}(i)$ with its helpfulness score as follows:

$$score_{weighted}(i) = score_{org}(i) \times \frac{helpful\ vote\ count}{total\ vote\ count} \tag{1}$$

This will results in the most helpful reviews having a bigger influence on the final rankings and the less helpful reviews having a smaller influence. We combine this weighting method with the three fusion methods CombMAX, CombSUM, and CombMNZ to arrive at a weighted fusion approach.

Book Ratings. In addition, users can also assign individual ratings from zero to five stars to the book they are reviewing, suggesting an additional method of taking into account the quality of the books to be retrieved. We used these ratings to influence the relevance scores of the retrieved books. For each retrieved review i we obtain a new relevance score $score_{weighted}(i)$ by multiplying that review's original relevance score $score_{org}(i)$ with its normalized rating r as follows:

$$score_{weighted}(i) = score_{org}(i) \times \frac{r}{5} \tag{2}$$

This will results in the positive reviews having a bigger influence on the final rankings and the negative reviews having a smaller influence. An open question here is whether positive reviews are indeed a better source of book recommendations than negative reviews. We combine this weighting method with the three fusion methods CombMAX, CombSUM, and CombMNZ to arrive at a weighted fusion approach.

Table 2 shows the results of the different social ranking runs for the optimal λ values. The results of the runs using the book-centric reviews index are also included for convenience.

Table 2. Results of the 9 different social ranking runs with the reviews-split index on the training set using NDCG as evaluation metric. The results of the runs using the book-centric reviews index are also included for convenience. Best-performing runs for each topic representation are printed in bold. The boxed run is the best overall using the reviews-split index.

Runs	Topic fields			
	title	narrative	group	all-topic-fields
CombMAX	0.3117	**0.3222**	0.0892	0.3457
CombSUM	**0.3377**	0.3185	**0.0982**	**0.3640**
CombMNZ	0.3350	0.3193	**0.0982**	0.3462
CombMAX - Helpfulness	0.2603	0.2842	0.0722	0.3124
CombSUM - Helpfulness	0.2993	0.2957	0.0703	0.3204
CombMNZ - Helpfulness	0.3083	0.2983	0.0756	0.3203
CombMAX - Ratings	0.2882	0.2907	0.0804	0.3306
CombSUM - Ratings	0.3199	0.3091	0.0891	0.3332
CombMNZ - Ratings	0.3230	0.3080	0.0901	0.3320
reviews	0.3020	0.2996	0.0773	0.3748

What do the results of the social ranking approaches tell us? The best overall social ranking approach is the unweighted CombSUM method using all available topic fields, with a NDCG score of 0.3640. Looking at the unweighted fusion methods, we see that our results confirm the work of, among others [5] and [6], as the CombSUM and CombMNZ fusion methods tend to perform better than CombMAX. For the weighted fusion approaches where the weights are derived from information about review helpfulness and book ratings we see the same patterns for these three methods: CombSUM and CombMNZ outperform CombMAX.

Overall, however, the unweighted fusion methods outperform the two weighted fusion methods. This is *not* in line with previous research [7,8], where the optimal combination of weighted runs tends to outperform the unweighted variants. This could be due to the fact that the relevance assessments for the training set can be incomplete or inaccurate. Another possibility is that our weighting methods using helpfulness and ratings are not optimal. It may be that reviews that are helpful for users are not necessarily helpful for a retrieval algorithm. Analogously, increasing the influence of positive reviews over negative reviews may not be the ideal approach either. We do observe however that using weights based on book ratings seem to have a slight edge over weights derived from review helpfulness.

Finally, if we compare the book-centric and review-centric approaches, we see a mixed picture: while the best result using the reviews-split index is not as good as the best result using the reviews index, this is only true for one of the four topic sets. For the other topic sets where the retrieval algorithm has less text to work with the review-centric approach actually comes out on top.

5 Submitted Runs

We selected four automatic runs for submission to INEX[6] based on the results of our content-based and social retrieval runs. Two of these submitted runs were content-based runs, the other two were social ranking-based runs.

Run 1. title.all-doc-fields This run used the titles of the test topics[7] and ran this against the index containing all available document fields, because this index provided the best content-based results.

Run 2. all-topic-fields.all-doc-fields This run used all three topic fields combined and ran this against the index containing all available document fields. We submitted this run because this combination provided the best overall results on the training set.

Run 3. title.reviews-split.CombSUM This run used the titles of the test topics and ran this against the review-centric reviews-split index, using the unweighted CombSUM fusion method.

[6] Our participant ID was 54.

[7] While our experiments showed that using only the title topic set did not provide the best results, submitting at least one run using only the title topic set was required by the track organizers.

Run 4. all-topic-fields.reviews-split.CombSUM This run used all three topic fields combined and ran this against the review-centric reviews-split index, using the unweighted CombSUM fusion method.

6 Results

The runs submitted to the INEX Social Book Search track were examined using three different types of evaluations [1]. In all three evaluations the results were calculated using NDCG@10, P@10, MRR and MAP, with NDCG@10 being the main metric.

LibraryThing Judgments for All 211 Topics. The first evaluation was using the 211 test set topics where the relevance judgments derived from the books recommended on the LibraryThing discussion threads of the 211 topics. Table 3 shows the results of this evaluation.

Table 3. Results of the four submitted runs on the test set, evaluated using all 211 topics with relevance judgments extracted from the LibraryThing forum topics. The best run scores are printed in bold.

Runs	NDCG@10	P@10	MRR	MAP
title.all-doc-fields	0.1129	0.0801	0.1982	0.0868
all-topic-fields.all-doc-fields	0.2843	0.1910	0.4567	**0.2035**
title.reviews-split.CombSUM	0.2643	0.1858	0.4195	0.1661
all-topic-fields.reviews-split.CombSUM	**0.2991**	**0.1991**	**0.4731**	0.1945

We see that, surprisingly, the best-performing runs on all 211 topics was run 4 with an NCDG@10 of 0.2991. Run 4 used all available topic fields and the unweighted CombSUM fusion method on the review-centric reviews-split index. Run 2, with all available document and topic fields was a close second.

Amazon Mechanical Turk Judgments for 24 Topics. For the first type of evaluation the book recommendations came from LibraryThing users who actually read the book(s) they recommend. The second type of evaluation conducted by the track participants enlisted Amazon Mechanical Turk (AMT) workers for judging the relevance of the book recommendations for 24 of the 211 test topics. These 24 topics were divided so that they covered 12 fiction and 12 non-fiction book requests. The judgments were based on pools of the top 10 results of all official runs submitted to the track, evaluated using all 211 topics. Table 4 shows the results of this second type of evaluation.

We see that consistent with the results on the training set the best-performing run on the 24 selected topics was run 2 with an NCDG@10 of 0.5415. Run 2 used all available topic and document fields. Runs 3 and 4 were a close second and third.

Table 4. Results of the four submitted runs on the test set, evaluated using 24 selected topics with relevance judgments from Amazon Mechanical Turk. The best run scores are printed in bold.

Runs	NDCG@10	P@10	MRR	MAP
title.all-doc-fields	0.4508	0.4333	0.6600	0.2517
all-topic-fields.all-doc-fields	**0.5415**	0.4625	**0.8535**	**0.3223**
title.reviews-split.CombSUM	0.5207	**0.4708**	0.7779	0.2515
all-topic-fields.reviews-split.CombSUM	0.5009	0.4292	0.8049	0.2331

If we split the topics by fiction and non-fiction book requests, an interesting pattern emerges for our top two performing runs. Run 2, with all available document and topic fields, achieved an NDCG@10 score of 0.5770 on non-fiction topics, but only a score of 0.5060 on fiction topics. In contrast, our second-best performing run on the 24 AMT topics (title.reviews-split.CombSUM) performed better on fiction topics with an NDCG@10 of 0.5465 compared to a score of 0.4949 on the non-fiction topics. This suggests that the different approaches have different strengths. Topics that request recommendations for fiction books might benefit more from using the available reviews than non-fiction books, because the content and themes of such books are more difficult to capture using the different curated and user-generated types of metadata. Reviews seem to contribute more to effective retrieval here, whereas the content of non-fiction books is more easily described using the available document fields.

LibraryThing Judgments for the 24 AMT Topics. The third type of evaluation used the same 24 AMT topics from the second evaluation, but with the original LibraryThing relevance judgments. Table 5 shows the results of this third type of evaluation.

Table 5. Results of the four submitted runs on the test set, evaluated using 24 selected Amazon Mechanical Turk topics with relevance judgments extracted from the LibraryThing forum topics. The best run scores are printed in bold.

Runs	NDCG@10	P@10	MRR	MAP
title.all-doc-fields	0.0907	0.0680	0.1941	0.0607
all-topic-fields.all-doc-fields	**0.2977**	**0.1940**	**0.5225**	**0.2113**
title.reviews-split.CombSUM	0.2134	0.1720	0.3654	0.1261
all-topic-fields.reviews-split.CombSUM	0.2601	**0.1940**	0.4758	0.1515

We see that, again consistent with the results on the training set, the best-performing run on the 24 selected topics with LibraryThing judgments was run 2 with an NCDG@10 of 0.2977. Run 2 used all available topic and document fields. Run 4 was a close second and third.

We also see that for the same 24 topics, evaluation scores are much lower than for the second type of evaluation. This is probably due to the varying number of documents judged relevant for the two sets of relevance judgments. The AMT judgments were produced by pooling the first ten of each officially submitted run, thus ensuring that each of the result would be judged. For the LibraryThing judgments, 67 of the 211 topics have fewer than five judgments, which negatively influences the calculation of NDCG@10, P@10 and MAP scores.

7 Discussion and Conclusions

Both in the the training set and the test set good results were achieved by combining all topic and document fields. This shows support for the principle of polyrepresentation [9] which states that combining cognitively and structurally different representations of the information needs and documents will increase the likelihood of finding relevant documents. However, using only the split reviews as index gave in four cases in the test set even better results, which speaks against the principle of polyrepresentation.

We also examined the usefulness of user-generated metadata for book retrieval. Using tags and reviews in separate indexes showed good promise, demonstrating the value of user-generated metadata for book retrieval. In contrast, the effort that is put into curating controlled metadata was not reflected its retrieval performance. A possible explanation could be that user-generated data is much richer, describing the same book from different angles, whereas controlled metadata only reflects the angle of the library professional who assigned them.

We also experimented with a review-centric approach, where all reviews were indexed separately and fused together at a later stage. This approach yielded good results, both on the training and the test set. We attempted to boost the performance of this approach even further by using review helpfulness and book ratings as weights, but this only decreased performance. At first glance, this is surprising since a helpful review can be expected to be well-written and well-informed. The quality of a book as captured by the rating could also be expected to have an influence on the review usefulness for retrieval, as could have been expected. Our current weighting scheme was not able to adequately capture these features though.

However, experimental evidence suggests that using a review-centric approach is a more promising approach to requests for fiction books than using all available document and topic fields is. More research is needed to confirm this however. Our overall recommendation would therefore be to always use all available document fields and topic representations for book retrieval.

7.1 Future Work

Future work would include exploring additional social re-ranking methods. As we are dealing with a book recommendation task, it would be a logical next step to explore techniques from the field of recommender systems, such as collaborative

filtering (CF) algorithms. One example could be to use book ratings to calculate the neighborhood of most similar items for each retrieved book and use this to re-rank the result list. The lists of (dis)similar items in the topic representations could also be used for this.

References

1. Kazai, G., Koolen, M., Kamps, J., Doucet, A., Landoni, M.: Overview of the INEX 2011 Book and Social Search Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 1–29. Springer, Heidelberg (2012)
2. Zhai, C., Lafferty, J.: A Study of Smoothing Methods for Language Models Applied to Information Retrieval. ACM Transactions on Information Systems 22(2), 179–214 (2004)
3. Järvelin, K., Kekäläinen, J.: Cumulated Gain-based Evaluation of IR Techniques. ACM Transactions on Information Systems 20(4), 422–446 (2002)
4. Jindal, N., Liu, B.: Review Spam Detection. In: WWW 2007: Proceedings of the 16th International Conference on World Wide Web, pp. 1189–1190. ACM, New York (2007)
5. Fox, E.A., Shaw, J.A.: Combination of Multiple Searches. In: TREC-2 Working Notes, pp. 243–252 (1994)
6. Lee, J.H.: Analyses of Multiple Evidence Combination. SIGIR Forum 31(SI), 267–276 (1997)
7. Renda, M.E., Straccia, U.: Web Metasearch: Rank vs. Score-based Rank Aggregation Methods. In: SAC 2003: Proceedings of the 2003 ACM Symposium on Applied Computing, pp. 841–846. ACM, New York (2003)
8. Bogers, T., Van den Bosch, A.: Fusing Recommendations for Social Bookmarking Websites. International Journal of Electronic Commerce 15(3), 33–75 (2011)
9. Ingwersen, P.: Cognitive Perspectives of Information Retrieval Interaction: Elements of a Cognitive IR Theory. Journal of Documentation 52(1), 3–50 (1996)

Using Page Breaks for Book Structuring

Hervé Déjean

Xerox Research Centre Europe
6 Chemin de Maupertuis, F-38240 Meylan
Firstname.lastname@xrce.xerox.com

Abstract. We report on the XRCE participation to the Structure Extraction task of the INEX/ICDAR Book Structure Extraction 2011. We wanted to assess a simple method for structuring a book: using leading and trailing page whitespace. The detection of such large whitespace occurring at the top of leading pages and at the bottom of trailing pages is based on the detection of the type area zone. Evaluation shows as expected a very good precision. Since this approach aims at detecting high level book structures (parts, chapters), structures not marked a page break are not detected (thus a lower recall).

1 Introduction

Our previous participations to the INEX book structure task made use of several components in order to achieve the best possible result. In this participation, we only used one new component based on the detection of page breaks. Such breaks which correspond to a high-level structure of the book (part, chapter) generate whitespaces on top of pages (leading pages) or bottom of pages (trailing pages). This notion of *leading and trailing pages* is an extension of the usual leading and trailing white spaces which exist at the line level: white spaces occurring at the beginning and the end of a line.

Definition 1: We call *trailing page* a page where the content flow ends, which often results in a white space zone occurring on bottom of its type area.

Definition 2: We call *leading page* a page where a main logical document unit starts, which often results in a white space zone on top of its type area.

Definition 3: We call *page break* a delimitation produced by trailing or leading pages.

The detection of leading and trailing pages allows for the recognition of high-level logical units of a document. By completing this detection by the extraction of titles on these pages, the high-level organization of the document is extracted.

To reliably detect these whitespace zones, we use the typographical notion of **type area**. The type area corresponds to the zone of the page where running text is laid out. This zone corresponds most of the time to the zone delimited by the four margins. Once this zone is detected for a given page, whitespaces zones occurring at the top and bottom of the page can be reliable detected.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 57–67, 2012.

We will first present how type areas are found for each page of a book, but collecting information at the book level. For this a prototypical type area is computed, and then matched against the pages of the book. This method works well even with noisy page images. In a second part, the detection of leading and trailing pages is explained. The method is evaluated against the 2011 Book Structure collection, and shows very good precision.

2 Type Area Detection

2.1 Definition

The *type area* (or print space) corresponds to the effective area on the page. It is surrounded by the four page margins. Rules exist to define and locate the type area inside a page [1]. The only reference, in the document recognition domain, to type area detection was found in [2], where it is called page frame. [2] defines a page frame as "the smallest rectangle that encloses all the foreground elements of the document page". The purpose of this work is to delete marginal noise in a page image, and to apply OCR only on the relevant page frame. In his definition, the page frame corresponds to the bounding box (smallest rectangle) of some selected elements (foreground elements).

We propose a more abstract and typographical-based definition for the type area, which refers back to the document model used to lay-out the content of the document. Instead of defining the type area as a minimal zone (the smallest rectangle), we consider it as a *regular* page zone over the document which content is laid out on. Figure 1 shows the difference between both definitions. Figure 1(a) shows the type area zone as defined by [2], the smallest bounding box around content. Figure 1(c) shows the type area as defined by ourselves for a “full page”. Figure 1(b) and (d) show leading and trailing page, where their type areas include whitespace zones.

The main, but crucial, difference between both definitions is that our type area is not always full of content (text, image), but can also enclose white space. To compute this rectangle, additional information other than the current processed page is required. The type area is computed considering *all the pages* of the document. The advantages of this definition are:

- to provide a better description of the page layout, especially by integrating white space zones into the type area zone
- to be more consistent with regard to typographical concepts
- to allow for a robust and very fast method when working at document level

This area is indeed computed at document level, collecting information from all the pages of the document to build a prototypical type area and margins. Then this area is projected on each page of the document. We will now explain how this area is computed and projected on pages, the latter being the most difficult in case of noise.

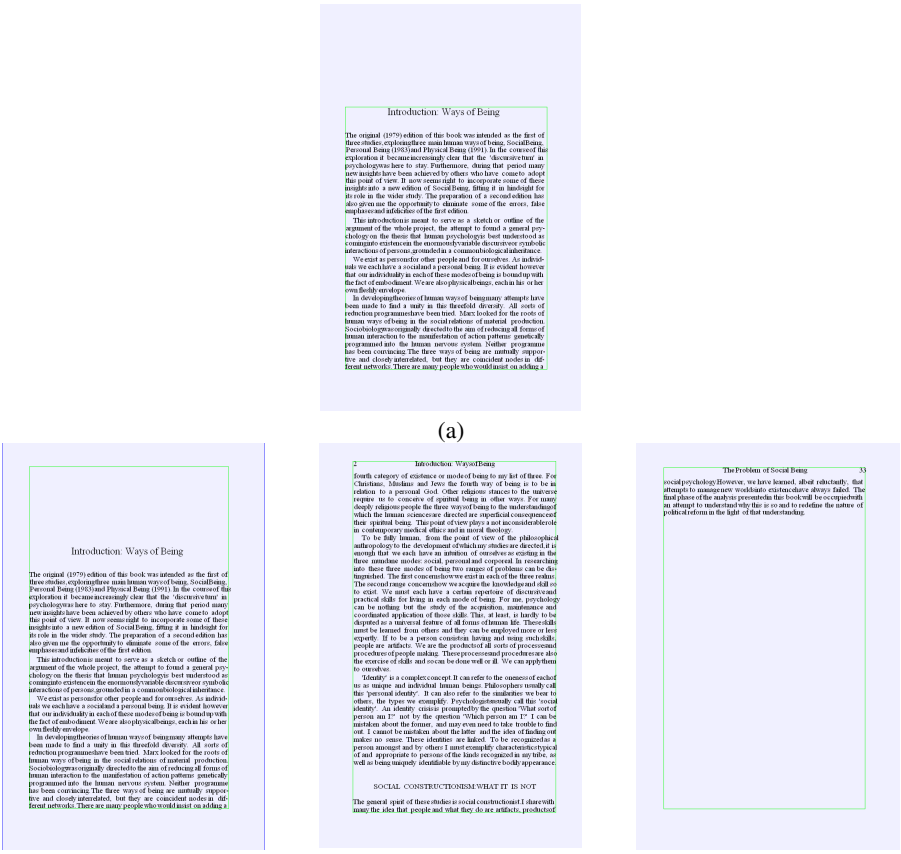


Fig. 1. Figure 1(a) shows the type area as defined by [2], the bounding box of non-noisy elements. Figures 1(b-c-d) shows examples of our type area: a regular surface, which easy allows for detecting leading (1-b), and trailing pages (1-d).

2.2 Method

A baseline approach to compute the type area dimensions, its width and height, of a document is to compute the frequency histogram of the content bounding boxes for all the pages of the document. The most frequent bounding box dimensions can be considered as the type area dimensions (assuming the simplified case of one sole type area per document). For this competition, the content bounding box computation makes use of the provided textual elements.

This simple approach frequently provides accurate type area dimensions for non noisy cases (digital-born documents). Noise may come from the previous steps: scanning or OCR (specially zoning) steps as discussed in [2] and shown Figure 2.

For these cases, a more elaborated method is required. An improvement consists in measuring independently the different values of the type are: its height, width and the generated 4 margins. The mains steps of the method are then the following ones:

- type area dimension generation (height, width)
- Margin histogram generation (for left, top, right, bottom margins)
- Combining type area dimension and margin dimension.
- Type area matching

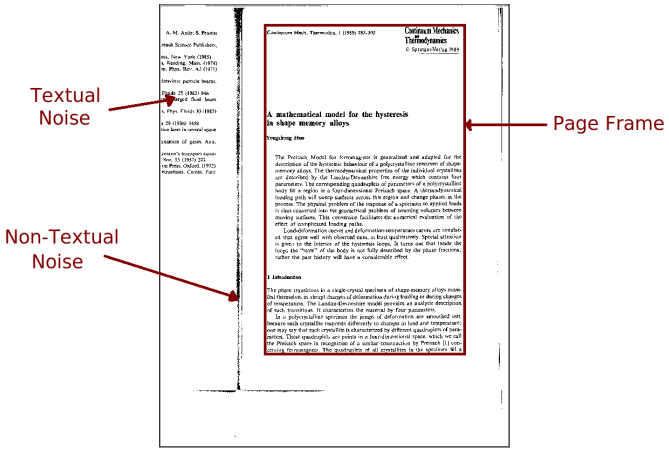


Fig. 2. (From [2]). A case of noisy page. In such a case, the content bounding box does not provide the correct type area dimensions and position.

Step 1: Type Area Dimensions Histogram

For each page, the bounding box of the page content is computed and its bounding box height and width frequencies are collected. Then the type area dimension couples are generated by generating all pairs (height, width). A score is associated to each couple corresponding to the sum of its height and width frequency.

Step 2: Margin Dimension Histogram Generation

Based on the content bounding box of each page of a given document, a frequency histogram for the left, top, right and bottom margin dimensions is computed (knowing the kind of margin, a single value is enough to characterize it). Similarly to the type area, all the possible 4-ples (left, top, right, bottom margin widths) are generated, and their scores correspond to the sum of each dimension frequency over the document. Optionally, odd and even pages are considered separately in order to easily cope with mirrored pages (in this case the left and right margins are not identical for odd and even pages).

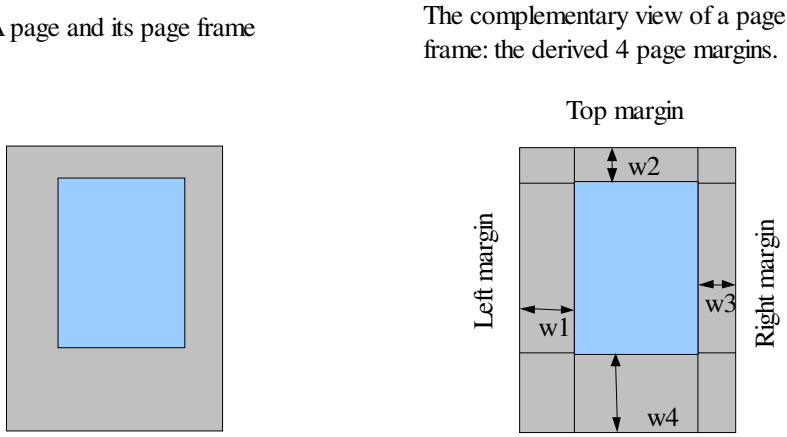
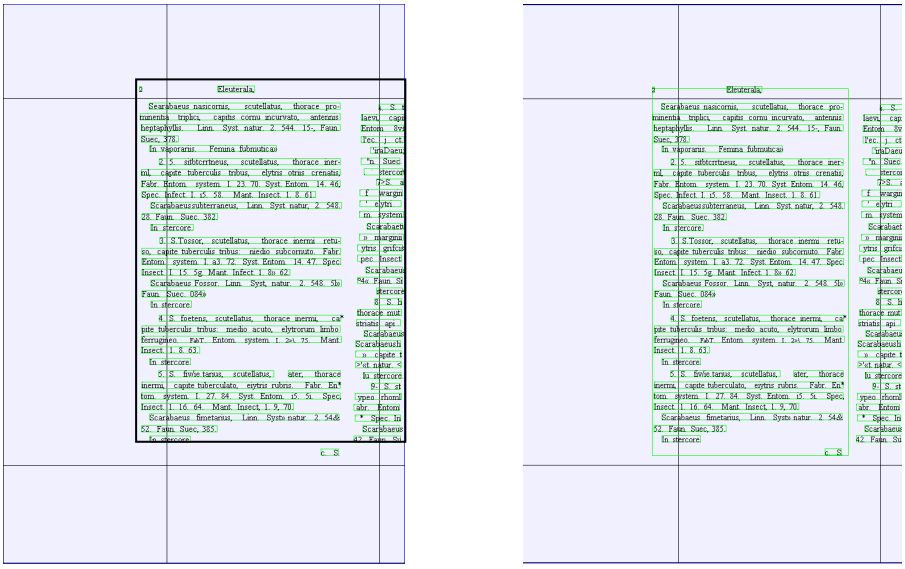


Fig. 3. A page structure: its type area and its margin zones

Step 3: Associating Type Area Dimensions and Margins Positions

The list of type area dimensions (height, width) is sorted in decrease order of score, and is validated against the list of margin dimensions sorted in decrease order of score.



(a) (b)

Fig. 4. (a) The content bounding box is indicated as well as the 4 prototypical margins. (b) After the best matching step, the correct type area is identified.

The purpose of this step is to associate type area dimension and margin dimension to fully characterize a type area by its dimension and position. Starting with the dimension couple (height, width) with the highest score, a couple is associated to a margin dimension: The dimensions of a type area must fit the central space created by the 4 margins. The identified margins are called hereafter **prototypical margins** (indicated by vertical and horizontal lines Figure 4).

Step 4: Matching a Type Area against a Page

The final and most important step is to match the found type area up with the document pages.

For this, the following heuristic is used to perform an efficient matching: Knowing the dimensions of the type area, we only need to identify a specific point in the page, the *anchor point,* corresponding to a specific corner of the type area to draw the type area rectangle. The difficulty is to identify a set of relevant points to be scored, and to identify the best point among them as *anchor point.* For this, we use the content of the page and the prototypical margins. First the content bounding box is drawn. Each of its corners is a potential anchor point. In order to cope with noise, other points are identified: In the current implementation, lines corresponding to frequent vertically aligned elements are taken into consideration.

Then we draw the 4 prototypical margins on the page, and compute the distance between each margin corner (intersection of two margins) and each of these relevant points. Each point is evaluated using a scoring function which considers two elements:

1. its distance to the margin corners
2. the number of elements (here the words) intersected by the type area if this point is taken as anchor point (number of intersected elements divided by the total number of elements).

The point with the best score is taken as anchor point and the type area is positioned in the page.

Table 1. The final scores of the points generated the content bounding box of the page shown Figure 4. The bottom left corner of the bounding box is selected as anchor point, and the type area is drawn accordingly.

Corner of bounding box as candidate anchor point	**Shortest distance to a margin corner**	**Intersected elements score**	**Final score**
A1: top left	37.1	0.01	0.37
A2: top right	33.6	0.36	12.08
A3: bottom left	**31.0**	**0.01**	**0.31**
A4: bottom right	31.0	0.36	11.10

A final test is performed: if a part of the type area is outside the page rectangle, the type area is invalidated, and no type area is associated with the page (this situation occurs for very noisy pages).

If a page is empty, no type area is associated to it.

3 Leading and Trailing Page Detection

The precise detection and positioning of page type areas allows for the categorization of several types of pages:

(a) empty page
(b) full page
(c) leading page
(d) trailing page
(e) noisy page

The described method associated and positioned at most one type area to a page. For some pages, too noisy or too empty, no type area is associated. These pages are considered as non reliable and ignored for the detection of page breaks (category e). The empty page corresponds to a page with no content.

The detection of trailing and leading pages (definition given Section 1) is simply based on a ratio between the type area height and the position of the content bounding box (see Figure 5). For detecting trailing pages, we compute a ratio *rt*, between the horizontal position of the last element of the page and the type area height. If the ratio *rt* is smaller than a given threshold Θ, the page is considered as a trailing page.

$$rt = \frac{Hbottom}{Hpf}$$

For detecting leading pages, we compute a ratio, *rl*, between the horizontal position of the top element and the type area height. If the ratio rl is higher than a given threshold Θ, the page is considered as leading page

$$rl = \frac{Htop}{Hpf}$$

This threshold Θ is the key parameter of the method. We use the same value for both trailing and leading pages.

Another case is to be considered: centered pages, where the content in vertically centered. This mainly corresponds to title pages. To identify such pages, the type area is divided into three zones, and pages containing only text in the centered zone are considered as a centered page.

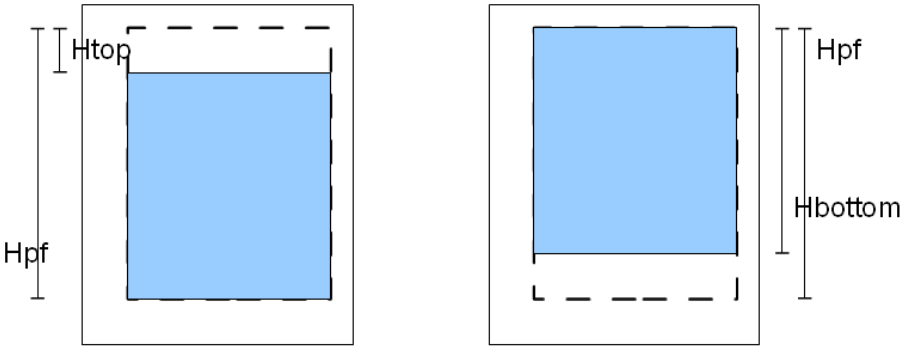


Fig. 5. Computation of leading and trailing pages

Experimentally, the best value for Θ is 0.90 (values between 0.50 and 0.95 have been tested with our development dataset). This value can be considered as high, but, considering that you can write on average 40-50 lines in your page, its value roughly corresponds to 3-5 empty lines. Below this threshold, the page is considered as a full page. Due to some noise in the positioning of the type area, and also due to some layout convention (such as no widowed line, orphaned word), it is difficult to increase the value of Θ in order to detect whitespaces corresponding to 1 or 2 empty lines. Decreasing this value makes the method more precise (micro-averaged precision), but recall is heavily impacted.

We use the following rules to detect the start of logical units based on trailing and leading pages:

- If a current page p is a leading page, then p is a page break.
- If a page p is a trailing page, then the next non empty page and non noisy page is considered as a page break.

The sequence of page breaks provides a segmentation of a book into high level units, mostly corresponding to the chapter level for the INEX collection.

4 Title Extractions

Once page breaks are found, titles corresponding to these breaks are extracted. Here a simple heuristics is used: page headers are first detected (using [Dejean et al.]), then, omitting headers, the first line is systematically considered as title. The second line is considered as part of the title is its length is smaller than the type area width (a ratio of 0.8 is used).

5 Two Runs

We submitted two runs: the first run corresponds to the method as explained previously (Section 2 to 4). We also combine this method with our method based on the detection of the table of contents (used and described in the previous

competitions). Both methods are applied, and the final solution is generated following these rules:

- if the TOC-based solution does not produce output, the page-break based output is taken
- if the page-break based solution does not generate output, the TOC-based output is taken
- if both solutions generate outputs, they are compared: if the number of entries in both outputs is similar, the Toc-based output is taken. Otherwise the page-break based output is taken.

The underlying idea, using this combination, was to increase the recall of the method by adding entries provided by the TOC-based method, and missed by the page-break method (a chapter can end at the end of a page, and thus having no corresponding trailing page).

6 Evaluation

We discuss here the so-called XRCE linked evaluation (Table 2). The other official INEX evaluation is available at the official web-site. This evaluation shows that our method is the most precise one (almost 10 points above the second one). The page break criterion is then, as expected, very reliable for such type of collections (mostly novels). Errors are due to breaks not present in the table of contents (prefaces, appendixes). See [4] for a general discussion on the issues for such a task. The second run, XRCE-2, corresponding to the combination of page-break and TOC-based methods, is disappointing, regarding the added complexity. This combination provides a slightly better F score, by improving recall (+2.6) and degrading precision (-3.4). Since a key value of the method is its precision, we consider this experience has disappointing.

The low recall of the method, compared to the best solutions, is easily explainable: the page-break method only segments high-level structures, and completely ignores low level structures, which are not marked with page breaks. The current groundtruth does not allow to easily (automatically) separate high level and low-level entries, thus an evaluation of the other methods for these specific entries was not conducted.

An immediate improvement regarding recall should be to consider leading and trailing columns instead of leading and tailing pages: the current implementation does not take care properly of multi-columns documents (except for multi-column layout which impose an equal length for the last page columns).

[5] is the closest work related to this page-break solution. It presents a solution based on a local context of 4 pages. The rules are as follows: chapter title detection throughout the document was conducted using a four-page sliding window. It is used to detect chapter transitions. The underlying idea is that the chapter begins after a blank, at least a blank at the top of page. The half page fill rate is the simple cue used to decide on chapter transition. The beginning of a chapter is detected by one of the two patterns below, where *i* is the page when a chapter starts:

- top of page *i-2* and bottom of page equally filled
- bottom of page *i-1* less filled than top of page
- top of page *i* less filled than bottom of page
- top of page *i+1* and bottom of page equally filled
- empty page *i-1*
- top of page *i* less filled than bottom of page
- top of page *i+1* and bottom of page equally filled

They use a four-page window in order to determine whitespaces, where we rely on the notion of type area. Evaluation shows that our method provided better results.

Table 2. The method shows the best precision

	P-link	R-link	F-link	Ttl-acc	RelLvl-acc
MDCS	64.5%	70.2%	65.1%	83.7%	79.2%
NANKAI-1	67.6%	67.4%	63.2%	74.4%	76.3%
NANKAI-2	66.0%	60.3%	59.8%	75.9%	75.5%
NANKAI-3	65.8%	60.3%	59.8%	75.9%	75.5%
NANKAI-4	67.6%	67.4%	63.2%	74.4%	76.3%
UNICAEN-1	65.2%	49.9%	50.7%	46.2%	61.4%
UNICAEN-2	65.2%	49.9%	50.7%	46.2%	80.4%
UNICAEN-3	32.5%	24.5%	24.4%	31.1%	64.0%
XRCE-1	**79.3%**	52.5%	57.6%	60.9%	78.6%
XRCE-2	75.9%	55.1%	58.1%	63.7%	77.9%

7 Conclusion

The purpose of this participation was to assess the notion of type area and page breaks for structuring books. Once type areas are computed, the detection of leading and trailing pages is only based on the unique threshold. Evaluations show that a value of 10% (a page is a break if at least 10% on its top or bottom is empty) provides best results, and is consistent over several collections. If this method allows for detecting important structures in a book, the remaining substructures (which are not linked to a page break) have to be tackled with others techniques.

As shown in the evaluation, the combination of methods (here for extracting the titles and for adding missing entries) is not easy, and is, in this context, useless. We hope for the next participation to rely on the precision of the presented method, and test a new method for detecting the low-level TOC entries, not covered by the latter method.

References

1. Tschichold, J.: The form of the book: essays on the morality of good design. Hartley & Marks, Point Roberts (1991)
2. Shafait, F., van Beusekom, J., Keysers, D., Breuel, T.M.: Document cleanup using frame detection. International Journal of Document Analysis and Recognition 11, 81–96 (2008)
3. Déjean, H., Meunier, J.-L.: A System for Converting PDF Documents into Structured XML Format. In: Bunke, H., Spitz, A.L. (eds.) DAS 2006. LNCS, vol. 3872, pp. 129–140. Springer, Heidelberg (2006)
4. Déjean, H., Meunier, J.-L.: Reflections on the INEX structure extraction competition, Boston. In: Document Analysis Systems, pp. 301–308 (2010)
5. Giguet, E., Baudrillart, A., Lucas, N.: Resurgence for the Book Structure Extraction Competition. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009 Workshop Pre-Proceedings (2009)

Social Recommendation and External Resources for Book Search

Romain Deveaud[1], Eric SanJuan[1], and Patrice Bellot[2]

[1] LIA - University of Avignon
339, Chemin des Meinajaries, F-84000 Avignon Cedex 9
{romain.deveaud,eric.sanjuan}@univ-avignon.fr
[2] LSIS - Aix-Marseille University
Domaine Universitaire de Saint Jérôme, F-13397 Marseille Cedex 20
patrice.bellot@lsis.org

Abstract. In this paper we describe our participation in the INEX 2011 Book Track and present our contributions. This year a brand new collection of documents issued from Amazon was introduced. It is composed of Amazon entries for real books, and their associated user reviews, ratings and tags.

We tried a traditional approach for retrieval with two query expansion approaches involving Wikipedia as an external source of information. We also took advantage of the social data with recommendation runs that use user ratings and reviews. Our query expansion approaches did not perform well this year, but modeling the popularity and the interestingness of books based on user opinion achieved encouraging results. We also provide in this paper an insight into the combination of several external resources for contextualizing tweets, as part of the Tweet Contextualization track (former QA track).

1 Introduction

Previous editions of the INEX Book Track focused on the retrieval of real out-of-copyright books [3]. These books were written almost a century ago and the collection consisted of the OCR content of over 50,000 books. It was a hard track because of vocabulary and writing style mismatches between the topics and the books themselves. Information Retrieval systems had difficulties to found relevant information, and assessors had difficulties judging the documents.

This year, for the books search task, the document collection changed. It is now composed of the Amazon pages of real books. IR systems must now search through bibliographic information, user reviews and ratings for each book, instead of searching through the whole content of the book. The topics were extracted from the LibraryThing[1] forums and represent real requests from real users.

[1] http://www.librarything.com/

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 68–79, 2012.

This year we experimented with query expansion approaches and recommendation methods. Like we already did for the INEX 2010 Book track, we used a language modeling approach to retrieval. We started by using Wikipedia as an external source of information, since many books have their dedicated Wikipedia article [4]. We associate a Wikipedia article to each topic and we select the most informative words from the articles in order to expand the query. For our recommendation runs, we used the reviews and the ratings attributed to books by Amazon users. We computed a "social relevance" probability for each book, considering the amount of reviews and the ratings. This probability was then interpolated with scores obtained by Maximum Likelihood Estimates computed on whole Amazon pages, or only on reviews and titles, depending on the run.

The rest of the paper is organized as follows. The following Section gives an insight into the document collection whereas Section 3 describes the our retrieval framework. Finally, we describe our runs in Section 4 and discuss some results in Sections 5 and 6.

2 The Amazon Collection

The document used for this year's Book Track is composed of Amazon pages of existing books. These pages consist of editorial information such as ISBN number, title, number of pages etc... However, in this collection the most important content resides in social data. Indeed Amazon is social-oriented, and user can comment and rate products they purchased or they own. Reviews are identified by the `<review>` fields and are unique for a single user: Amazon does not allow a forum-like discussion. They can also assign tags of their creation to a product. These tags are useful for refining the search of other users in the way that they are not fixed: they reflect the trends for a specific product. In the XML documents, they can be found in the `<tag>` fields. Apart from this user classification, Amazon provides its own category labels that are contained in the `<browseNode>` fields.

Table 1. Some facts about the Amazon collection

Number of pages (i.e. books)	2,781,400
Number of reviews	15,785,133
Number of pages that contain at least a review	1,915,336

3 Retrieval Model

3.1 Sequential Dependence Model

Like in 2010, we used a language modeling approach to retrieval [5]. We use Metzler and Croft's Markov Random Field (MRF) model [6] to integrate multi-word phrases in the query. Specifically, we use the Sequential Dependance Model

(SDM), which is a special case of the MRF. In this model three features are considered: single term features (standard unigram language model features, f_T), exact phrase features (words appearing in sequence, f_O) and unordered window features (requiring words to be close together, but not necessarily in an exact sequence order, f_U).

Documents are thus ranked according to the following scoring function:

$$score_{SDM}(Q, D) = \lambda_T \sum_{q \in Q} f_T(q, D) + \lambda_O \sum_{i=1}^{|Q|-1} f_O(q_i, q_{i+1}, D) + \lambda_U \sum_{i=1}^{|Q|-1} f_U(q_i, q_{i+1}, D)$$

where the features weights are set according to the author's recommendation ($\lambda_T = 0.85, \lambda_O = 0.1, \lambda_U = 0.05$). f_T, f_O and f_U are the log maximum likelihood estimates of query terms in document D, computed over the target collection with a Dirichlet smoothing.

3.2 External Resources Combination

As previously done last year [2], we exploited external resources in a Pseudo-Relevance Feedback (PRF) fashion to expand the query with informative terms. Given a resource $\mathcal{R}$, we form a subset $\mathcal{R}_Q$ of informative documents considering the initial query Q using pseudo-relevance feedback. To this end we first rank documents of $\mathcal{R}$ using the SDM ranking function. An entropy measure $H_{\mathcal{R}_Q}(t)$ is then computed for each term t over $\mathcal{R}_Q$ in order to weigh them according to their relative informativeness:

$$H_{\mathcal{R}_Q}(t) = -\sum_{w \in t} p(w|\mathcal{R}_Q) \cdot \log p(w|\mathcal{R}_Q)$$

These external weighted terms are finally used to expand the original query. The ranking function of documents over the target collection $\mathcal{C}$ is then defined as follows:

$$score(Q, D) = score_{SDM}(Q, D) + \frac{1}{|\mathcal{S}|} \sum_{\mathcal{R}_Q \in \mathcal{S}} \sum_{t \in \mathcal{R}_Q} H_{\mathcal{R}_Q}(t) \cdot f_T(t, D)$$

where $\mathcal{S}$ is the set of external resources.

For our official experiments with the Book Track we only considered Wikipedia as an external resource, but we also conducted unofficial experiments on the Tweet Contextualization track after the workshop. In order to extract a comprehensive context from a tweet, we used a larger set $\mathcal{S}$ of resources. It is composed of four general resources: Wikipedia as an encyclopedic source, the New

York Times and GigaWord corpora as sources of news data and the category B of the ClueWeb09 collection as a web source. The English GigaWord LDC corpus consists of $4,111,240$ newswire articles collected from four distinct international sources including the New York Times. The New York Times LDC corpus contains $1,855,658$ news articles published between 1987 and 2007. The Wikipedia collection is a recent dump from July 2011 of the online encyclopedia that contains $3,214,014$ documents. We removed the spammed documents from the category B of the ClueWeb09 according a standard list of spams for this collection[2]. We followed authors recommendations [1] and set the "spamminess" threshold parameter to 70, the resulting corpus is composed of $29,038,220$ web pages. We present the results in the dedicated sections below.

3.3 Wikipedia Thematic Graphs

In the previous methods we expand the query with words selected from pages directly related to the query. Here, we wanted to select broader, more general words that could stretch topic coverage. The main idea is to build a thematic graph of Wikipedia articles in order to generate a set of articles that (ideally) completely covers the topic.

For this purpose we use anchor texts and their associated hyperlinks in the first Wikipedia page associated to the query. We keep the term extraction process detailed in Section 3.2 for selecting a Wikipedia page highly relevant to the query. We extract informative words from this page using the exact same method as above. But we also extract all anchor texts in this page. Given $T^{\mathcal{W}}$ the set of words extracted by entropy from the Wikipedia article $\mathcal{W}$ and $A^{\mathcal{W}}$ its set of anchor texts. We then compute the intersection between the set $T^{\mathcal{W}}$ and each anchor text $A_i^{\mathcal{W}}$. The intersection is not null if at least one contextual informative word is present in the anchor text. We then consider that the Wikipedia article that is linked with the anchor text is thematically relevant to the first retrieved Wikipedia article. Then we sum the previously computed entropies for all words from $T^{\mathcal{W}}$ occurring in the anchor text, which gives a confidence score for anchor $A_i^{\mathcal{W}}$. The computation of this score can be formalized as follows:

$$s_P(A_i^{\mathcal{W}}) = \sum_{t \in T^{\mathcal{W}} \cap A_i^{\mathcal{W}}} H_{\mathcal{W}}(t)$$

This thematic link hypothesis between Wikipedia articles relies on the fact that anchor texts are well-written and reviewed by the community. Each contributor can edit or correct an article while moderators can prevent abuses. This behavior was previously noted by [8] within the frame of experiements on the semantic relations that exist between lexical units. This study shows that using Wikipedia, an open and collaborative resource, achieves better results than the use of ontologies or hand-crafted taxonomies in some cases. These reflections hence justify our use of anchor texts to model thematic links between Wikipedia articles, or every other collaborative resource.

[2] http://plg.uwaterloo.ca/~gvcormac/clueweb09spam/

We can iterate and construct a directed graph of Wikipedia articles linked together. Children node pages (or *sub-articles*) are weighted half that of their parents in order to minimize a potential *topic drift*. We avoid loops in the graph (i.e. a child node can not be linked to one of his elder) because it brings no additional information. It also could change weights between linked articles. Informative words are then extracted from the sub-articles and incorporated to our retrieval model like another external resource.

3.4 Social Opinion for Book Search

The test collection used this year for the Book Track contains Amazon pages of books. These pages are composed amongst others of editorial information, like the number of pages or the blurb, user ratings and user reviews. However, contrary to the previous years, the actual content of the books is not available. Hence, the task is to rank books according to the sparse informative content and the opinion of readers expressed in the reviews, considering that the user ratings are integers between 1 and 5.

Here, we wanted to model two social popularity assumptions: a product that has a lot of reviews must be relevant (or at least popular), and a high rated product must be relevant. Then, a product having a large number of good reviews really must be relevant. However in the collection there is often a small amount of ratings for a given book. The challenge was to determine whether each user rating is significant or not. To do so, we first define X_R^D a random set of "bad" ratings (1, 2 or 3 over 5 points) for book D. Then, we evaluate the statistical significant differences between X_R^D and $X_R^D \cup X_U^D$ using Welch's t-test, where X_U^D is the actual set of user rating for book D. Finally, we take the complement of the test p-value as the probability that reviewers like the book.

The underlying assumption is that significant differences occur under two different situations. First, when there is a small amount of user ratings (X_U^i) but they all are very good. For example this is the case of good but little-known books. Second, when there is a very large amount of user ratings but there are average. Hence this statistical test gives us a single estimate of both likability and popularity.

We use our SDM baseline defined in section 3.1 and incorporate the above recommendation estimate:

$$score_{recomm}(Q, D) = \lambda_D \; score_{SDM}(Q, D) + (1 - \lambda_D) \; t_D$$

where the λ_D parameter was set based on the observation over the test topics made available to participants for training purposes. Indeed we observed on these topics that the t_D had no influence on the ranking of documents after the hundredth result (average estimation). Hence we fix the smoothing parameter to:

$$\lambda_D = \frac{\arg\max_D score_{SDM}(Q, D) - score_{SDM}(Q, D)_{100}}{N_{Results}}$$

In practice, this approach is re-ranking of the results of the SDM retrieval model based on the popularity and the likability of the different books.

4 Runs

This year we submitted 6 runs for the Social Search for Best Books task only. We used Indri[3] for indexing and searching. We did not remove any stopword and used the standard Krovetz stemmer.

baseline-sdm. This run is the implementation of the SDM model described in Section 3.1. We use it as a strong baseline.

baseline-tags-browsenode. This is an attempt to produce an improved baseline that uses the Amazon classification as well as user tags. We search all single query terms in the specific XML fields (`<tag>` and `<browseNode>`). This part is then combined with the SDM model, which is weighted four times more than the "tag searching" part. We set these weights empirically after observations on the test topics. The Indri syntax for the query `schumann biography` would typically be:

```
#weight (
  0.2 #combine ( #1(schumann).tag #1(biography).tag
                 #1(schumann).browseNode #1(biography).browseNode )
  0.8 #weight ( 0.85 #combine( Schumann Biography )
                0.1  #combine( #1(schumann biography) )
                0.05 #combine( #uw8(schumann biography) ) )
        )
```

sdm-wiki. This run is the implementation of the external resources combination model described in Section 3.2, only applied to a single resource: Wikipedia. The Wikipedia API was queried on August, 2011. For each topic we extract the 20 top informative words based on their entropy measure from the top ranked article given by the Wikipedia API. We then reformulate the initial query by adding these words with their entropy as weights. The motivation to do this was that there are many books that have their dedicated Wikipedia article [4]. If we could select the proper article and extract informative words about a book topic or a book series, it could help retrieval.

sdm-wiki-anchors. This run is the implementation of the Wikipedia thematic graph approach described in Section 3.3. For each topic, we queried the Wikipedia API to retrieve the first ranked article. We then computed all thematic links between this first Wikipedia article (we call it $reference$) and all the others that are linked to it. We then extract the 20 top informative words from these linked articles in order to enrich the query with several thematically linked sources. In these experiments we only consider the top 5 linked articles with best $s_P(A_i^{\mathcal{W}})$ confidence score.

[3] http://www.lemurproject.org

sdm-reviews-combine. This run uses the social information contained in the user reviews, it is the implementation of the approach described in Section 3.4. First, a **baseline-sdm** is performed. We then extract the number of reviews and their ratings for each document previously retrieved. A probability that the book is popular is then computed with a Welch's t-test. This interestingness and popularity score is finally interpolated to the SDM score.

Recommendation. This run is similar to the previous one except that we compute a query likelihood estimate only on the `<title>` and on the `<content>` fields, instead of considering the whole document like the SDM does. Scores for the title and the reviews, and the popularity of the books are interpolated the same way as above. The sum of these three scores gives a recommendation score for each book based only on its title and on user opinions, without tanking into account any other editorial information.

5 Book Search Results and Discussion

The evaluation results shown below are based on the official INEX 2011 Social Books topic set, consisting of 211 topics from the LibraryThing discussion groups. There are two separate sets of relevance judgements. The first set is derived from the suggestions from members of the discussion groups, and is considered as the principal mean of evaluation for this task. The second set is based on judgements from Amazon Mechanical Turk for 24 out of the 211 topics. We present the results for the first set of relevance judgements in Table 2.

We observe that our **recommendation** approach performs the best amongst our other runs, while our two query expansion approaches with Wikipedia both fail. Our **baseline-sdm** run do not use any additional information except the user query (which is in fact the title of the corresponding LibraryThing thread), hence this is a good mean of comparison for other runs using social information for example. Despite that using an external encyclopedic resource like Wikipedia do not work for improving the initial query formulation, we see that a traditional pseudo-relevance feedback (PRF) approach achieved the best results overall this year. Indeed the approach of the University of Amsterdam (p4) was to expand the query with 50 terms extracted from the top 10 results, either performing over a full index or over an index that only include social tags (such as reviews, tags and ratings). The latter performed the best with their PRF approach, and it is coherent with the results of our **recommendation** run. Indeed in this run we only consider the content of the user reviews, which correspond to a limited version of the social index mentioned above. It also suggests that the baseline model is quite effective and selects relevant feedback documents, which is confirmed by the results computed with the Amazon Mechanical Turk judgements shown in Table 3.

In this table we see that the baselines perform very well compared to the others, and it confirms that a language modeling base system performs very well on this test collection. It is very good at retrieving relevant documents in the first

Table 2. Official results of the Best Books for Social Search task of the INEX 2011 Book track, using judgements derived from the LibraryThing discussion groups. Our runs are identified by the *p*62 prefix and are in boldface.

Run	nDCG@10	P@10	MRR	MAP
p4-inex2011SB.xml_social.fb.10.50	0.3101	0.2071	0.4811	0.2283
p54-run4.all-topic-fields.reviews-split.combSUM	0.2991	0.1991	0.4731	0.1945
p4-inex2011SB.xml_social	0.2913	0.1910	0.4661	0.2115
p4-inex2011SB.xml_full.fb.10.50	0.2853	0.1858	0.4453	0.2051
p54-run2.all-topic-fields.all-doc-fields	0.2843	0.1910	0.4567	0.2035
p62.recommendation	0.2710	0.1900	0.4250	0.1770
p54-run3.title.reviews-split.combSUM	0.2643	0.1858	0.4195	0.1661
p62.sdm-reviews-combine	0.2618	0.1749	0.4361	0.1755
p62.baseline-sdm	0.2536	0.1697	0.3962	0.1815
p62.baseline-tags-browsenode	0.2534	0.1687	0.3877	0.1884
p4-inex2011SB.xml_full	0.2523	0.1649	0.4062	0.1825
wiki-web-nyt-gw	0.2502	0.1673	0.4001	0.1857
p4-inex2011SB.xml_amazon	0.2411	0.1536	0.3939	0.1722
p62.sdm-wiki	0.1953	0.1332	0.3017	0.1404
p62.sdm-wiki-anchors	0.1724	0.1199	0.2720	0.1253
p4-inex2011SB.xml_lt	0.1592	0.1052	0.2695	0.1199
p18.UPF_QE_group_BTT02	0.1531	0.0995	0.2478	0.1223
p18.UPF_QE_genregroup_BTT02	0.1327	0.0934	0.2283	0.1001
p18.UPF_QEGr_BTT02_RM	0.1291	0.0872	0.2183	0.0973
p18.UPF_base_BTT02	0.1281	0.0863	0.2135	0.1018
p18.UPF_QE_genre_BTT02	0.1214	0.0844	0.2089	0.0910
p18.UPF_base_BT02	0.1202	0.0796	0.2039	0.1048
p54-run1.title.all-doc-fields	0.1129	0.0801	0.1982	0.0868

Table 3. Top runs of the Best Books for Social Search task of the INEX 2011 Book track, using judgements obtained by crowdsourcing (Amazon Mechanical Turk). Our runs are identified by the *p*62 prefix and are in boldface.

Run	nDCG@10	P@10	MRR	MAP
p62.baseline-sdm	0.6092	0.5875	0.7794	0.3896
p4-inex2011SB.xml_amazon	0.6055	0.5792	0.7940	0.3500
p62.baseline-tags-browsenode	0.6012	0.5708	0.7779	0.3996
p4-inex2011SB.xml_full	0.6011	0.5708	0.7798	0.3818
p4-inex2011SB.xml_full.fb.10.50	0.5929	0.5500	0.8075	0.3898
p62.sdm-reviews-combine	0.5654	0.5208	0.7584	0.2781
p4-inex2011SB.xml_social	0.5464	0.5167	0.7031	0.3486
p4-inex2011SB.xml_social.fb.10.50	0.5425	0.5042	0.7210	0.3261
p54-run2.all-topic-fields.all-doc-fields	0.5415	0.4625	0.8535	0.3223

ranks which is an essential quality for a system that performs PRF. Hence a query expansion approach can be very effective on this dataset, but feedback documents must come from the target collection and not from an external resource. It is however important to note that these judgements are coming from people that often are not experts or that do not have the experience of good readers.

Their assessments may then come from the suggestions of well-known search engines or directly from Amazon. This behavior could possibly explain the high performances of the baselines for the AMT judgements set.

To confirm this assessment, we tried to combine the four heterogenous resources mentioned in Section 3.2 and we reported the results on Table 2 under the unofficial run identified by ***wiki-web-nyt-gw***. Although the combination of multiple external resources does much better than using Wikipedia alone, it still does not beat our baseline. Hence can safely affirm that reformulating the query using a wide range of external sources of knowledge does not work when the target collection is mainly composed of recommendation or opinion-oriented text.

The other part of our contribution lies in the social opinion that we took into account in our ranking function. Indeed we are the only group that submitted runs that model the popularity and the likability of books based on user reviews and ratings. Royal School of Library and Information Science's group (p54) tried in their early experiments to define an helpfulness score for each review, aiming to give more weight to a review found truthful, and also tried to weigh books reviews according to their associated ratings. However these experiments showed that it didn't performed well compared to an approach where they sum the relevance score of all the reviews for a given book. The two runs we submitted that make use of social information (**recommendation** and **sdm-reviews-combine**) can both be viewed as a re-ranking of the baseline, and both of them improve its performance. The recommendation run only uses reviews content and the title of the book for the retrieval of books while the sdm-reviews-combine run uses the whole content of the Amazon/LibraryThing pages. The fact that the recommendation run performs best than the sdm-reviews-combine is coherent with the approach of Royal School of Library and Information Science described above. Additional information seems to be considered as noise while the real informative content is situated inside the reviews, but this may also be a smoothing issue. Indeed the size of the reviews are much larger than any other component in the documents ($\approx$ 156 words per review, while tags are only composed of 1 or 2 words), and defining specific smoothing parameter values for each field based on the average length of their length could perform better.

6 Contextualizing Tweets by Combining General Resources

Considering that the use of an external resource did not bring anything to social book search, we wanted to evaluate our resource combination approach on another track. This approach intuitively matches well against the Tweet Contextualization one (former QA track). Indeed its purpose is to extract relevant passages from Wikipedia in order to generate a readable summary (500 words maximum) giving insights into a topic of current interest. These topics are represented by tweets, which are in fact titles of New York Times articles. We use the exact same approach previously described in Section 3.2. Tweets are enriched

with additional information coming from the various external resources, and sentences are extracted from the target Wikipedia collection to form a contextual excerpt. The organizers provided a full baseline for participants that could not implement their own index of the Wikipedia collection. It is composed of a full state-of-the-art XML-element retrieval system which was already available for the previous edition of the INEX QA track [7]. We tried every combination of one, two, three and four resources, but we only report the approaches that use a single resource and the full combination of the four. Some official results are reported in Table 4 as well as those of our unofficial runs.

Table 4. Official results of the INEX 2011 Tweet Contextualization track. Our runs are unofficial and are in boldface, runs in italic are the official baselines.

Run	**Unigram**	**Bigram**	**With 2-gap**
ID12R_IRIT_default.run	0.8271	0.9012	0.9028
ID126R_Run1.run	0.7982	0.9031	0.9037
ID128R_Run2.run	0.8034	0.9091	0.9094
ID138R_Run1.run	0.8089	0.9150	0.9147
ID129R_Run2.run	0.8497	0.9252	0.9253
wiki-web-nyt-gw	0.8267	0.9273	0.9289
Baseline_sum.run	0.8363	0.9350	0.9362
gigaword	0.8409	0.9371	0.9383
ID18R_Run1.run	0.8642	0.9368	0.9386
nyt	0.8631	0.9437	0.9443
ID46R_JU_CSE_run1.run	0.8807	0.9453	0.9448
web	0.8522	0.9454	0.9466
wiki	0.8515	0.9454	0.9471
Baseline_mwt.run	0.9064	0.9777	0.9875

The evaluation metrics considers the absolute normalized log-difference between the result passages and the textual assessments. The main metric is **With 2-gap** and evaluates the frequency differences between "*pairs of consecutive lemmas, allowing the insertion between them of a maximum of two lemmas*". We see that despite the fact that the passage extraction method we use was the baseline provided by the organizers, using a single resource to reformulate the initial query (or tweet) does not beat the *Baseline_sum.run*. However we see that the GigaWord and the NYT corpora are the ones that harm retrieval the less, mainly because of their coverage of the news topics. Surprisingly, the use of Wikipedia as a single source of expansion (i.e. pseudo-relevance feedback) achieves the worst results of our unofficial runs. We did not have the time to further investigate, but this may be a first coverage indication of Wikipedia for the given topics. It also suggests that constructing a coherent summary based exclusively on Wikipedia for a given news topic is not an easy task. Despite the negative effect of single resources, we observe that the combination of the four resources performs better than the baseline. The improvement is statistically significant (t-test with

p-value < 0.05). The combination thus contextualizes effectively the information need from 3 different points of view corresponding to the 3 types of resources, namely: encyclopedic, news and web. This contextualization acts in the form of contextual features extracted from the different sources and used to reformulate the initial query (or tweet). These results are very promising and encouraging, and we aim at experimenting other means of contextualization with several kind of external data.

7 Conclusions

In this paper we presented our contributions for the INEX 2011 Book track. One main observation from this year's Book track was that the baselines based on a language modeling approach to retrieval were very hard to beat. This also helped the approaches that used pseudo-relevance feedback to perform well. We proposed a query expansion method that exploit four different resources as external sources of expansion terms. This method considers the most informative words of the best ranked articles in order to reformulate the query. It did not perform well overall and did not manage to beat our baseline. We also tried to build a limited thematic graph of Wikipedia articles in order to extract more expansion terms, but this approach was even less effective. This collection is mainly composed of user reviews that contain opinion-oriented text more than factual information, and using external information seems not to work here. However we tried to extend our method to the Tweet Contextualization track and saw that combining the four resources is effective and beats the baseline, while every single resource harms passage retrieval.

We also submitted two runs to the Book track that took advantage of the social information available in the Amazon collection. They exploit the number of reviews and the user ratings to compute popularity and likability scores that we interpolate with query likelihood probabilities. These approaches showed to be effective but still need some improvements, especially with the estimation of a "good" review. We aim to model the quality of a reviewer for the upcoming year, thus weighting the different reviews of a given book according to several criteria.

References

1. Cormack, G., Smucker, M., Clarke, C.: Efficient and effective spam filtering and re-ranking for large web datasets. Information Retrieval (2011)
2. Deveaud, R., Boudin, F., Bellot, P.: LIA at INEX 2010 Book Track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 118–127. Springer, Heidelberg (2011)
3. Kazai, G., Koolen, M., Kamps, J., Doucet, A., Landoni, M.: Overview of the INEX 2010 Book Track: Scaling Up the Evaluation Using Crowdsourcing. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 98–117. Springer, Heidelberg (2011)

4. Koolen, M., Kazai, G., Craswell, N.: Wikipedia pages as entry points for book search. In: Proceedings of the Second ACM International Conference on Web Search and Data Mining, WSDM 2009, pp. 44–53. ACM, New York (2009)
5. Metzler, D., Croft, W.B.: Combining the language model and inference network approaches to retrieval. Inf. Process. Manage. 40, 735–750 (2004)
6. Metzler, D., Croft, W.B.: A markov random field model for term dependencies. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2005, pp. 472–479. ACM, New York (2005)
7. SanJuan, E., Bellot, P., Moriceau, V., Tannier, X.: Overview of the INEX 2010 Question Answering Track (QA@INEX). In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 269–281. Springer, Heidelberg (2011)
8. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: Proceedings of the 21st National Conference on Artificial Intelligence, vol. 2, pp. 1419–1424 (2006)

The University of Massachusetts Amherst's Participation in the INEX 2011 Prove It Track

Henry A. Feild, Marc-Allen Cartright, and James Allan

Center for Intelligent Information Retrieval
University of Massachusetts Amherst, Amherst MA 01003, USA
{hfeild,irmarc,allan}@cs.umass.edu

Abstract. We describe the process that led to the our participation in the INEX 2011 Prove It task. We submitted the results of six book page retrieval systems over a collection of 50,000 books. Two of our runs use the sequential dependency model (a model that uses both unigrams and bigrams from a query) and the other four interpolate between language model scores at the passage level and sequential dependency model scores at the page level. In this report, we describe our observations of these and several other retrieval models applied to the Prove It task.

Keywords: INEX, Prove It, Book Retrieval, Sequential Dependency Model, Passage Retrieval.

1 Introduction

In this report we describe our submissions to the 2011 INEX Prove It task, where the goal is to rank book pages that are supportive, refutative, or relevant with respect to a given fact. We did not participate in the optional sub task of classifying each result as confirming or refuting the topic; in our submissions we labeled all retrieved documents as *confirming* the fact. To determine what retrieval systems to submit, we investigated several models. In the following sections, we detail those models and give a summary of the results that led to our submissions.

2 Indexing and Retrieval Models

We only considered indexing pages. The index used no other information about a page's corresponding book, chapter, or section and all tokens were stemmed using the Porter stemmer. We indexed a total of 6,164,793,369 token occurrences from 16,971,566 pages from 50,232 books using a modified version of the Galago retrieval system.[1]

We explored a number of models for page and passage retrieval, including relevance modeling, sequential dependence modeling, passage modeling, stop word removal, and mixtures thereof. We describe each below.

[1] http://galagosearch.org/

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 80–85, 2012.

Query Likelihood Language Modeling (QLM). This model scores each page by its likelihood of generating the query [4]. The model also smooths with a background model of the collection; for this, we used Dirichlet smoothing with the default smoothing parameter: $\mu = 1500$.

Relevance Modeling (RM). A form of pseudo relevance feedback, relevance modeling creates a language model from the top k pages retrieved for a query, expands the query with some number of the most likely terms from the model, and performs a second retrieval [2]. We investigated relevance modeling because, as with all pseudo relevance feedback methods, it allows the vocabulary of the original query to be expanded, hopefully capturing related terms. There are three parameters to set: the number of feedback pages to use (set to 10), the number of feedback term to use (also set to 10), and the weight to give the original query model and the relevance model for the second retrieval (set to 0.5). These are the default settings distributed with Galago.

Sequential Dependence Modeling (SDM). This model interpolates between document scores for three language models: unigram, bigram, and proximity of adjacent query term pairs [3]. Because of its use of bigrams, SDM captures portions of phrases that unigram models miss. The weight of each sub language model are parameters, and we used the defaults suggested by Metzler and Croft [3]: 0.85, 0.10, 0.05 for the unigram, bigram, and proximity models, respectively. In addition, we used Dirichlet smoothing for each language model and experimented with $\mu = 1500$ (the Galago default) and $\mu = 363$ (the average number of terms per page).

Passage Modeling (PM). This model first scores passages using QLM with Dirichlet smoothing (setting μ to the length of the passage), selects the highest passage score per page, and then interpolates between that score and the corresponding page's SDM score. In our implementation, the top 1,000 pages (*Pages*) and the top 10,000 passages (*Pass*)[2] are retrieved as two separate lists and then interpolated. If a passage is present in *Pass*, but the corresponding page is not in *Pages*, the page score is set to the minimum page score in *Pages*. Likewise, if a page is retrieved in *Pages* but no passages from that page are present in *Pass*, the lowest passage score in *Pass* is used as a proxy. The parameters of the PM model include the passage length l and the interpolation factor, λ, where the maximum passage score is weighted by λ and the page score is weighted by $1 - \lambda$. We experimented with several values of λ.

Stop Word Removal (Stop). When stopping is used, query terms found in a list of 119 stop words[3] are removed.

We considered several combinations of the above models, all using stemming. These include: QLM, RM, SDM, SDM+RM, PM, and each of these with and without stop words removed.

[2] We allow multiple passages per document to appear on this list; filtering the highest scoring passage per page is performed on this 10,000 passage subset.

[3] http://www.textfixer.com/resources/common-english-words.txt

Table 1. The INEX Prove It topic fields and examples

Field	Example 1	Example 2	Example 3
ID	2010000	2010012	2010015
Fact	In the battle of New Orleans on the 8th of January 1815, 2000 British troops were killed, wounded or imprisoned, while only 13 American troops were lost, 7 killed and 6 wounded.	The main function of telescope is to make distant objects look near.	Victor Emanuel enters Rome as king of united Italy.
Info need	All sections of books that detail the losses suffered either at the British or the American side are relevant. I am not interested in how the battle was fought, but just want to find out about the losses at the end of the battle.	Most of the book is relevant to Astronomy as its a handbook on astronomy.	Italy will celebrate next year its re-unification and I needed to check the facts and their dates. Italy had two other capitals, Turing and Florence, before it was possible to get Rome back from the Vatican State.
Query	New Orleans battle 1815 troops lost killed	Telescope	Rome capital
Subject	battle of New Orleans 1815	telescope	Rome becomes capital of united Italy
Task	My task is to find out the scale of losses on both the British and American side in the battle of New Orleans in 1815	We need to write a primer on Astronomy.	Find out the date when Rome became capital of reunited Italy.

3 Training Data

Of the 83 total topics available for the Prove It task, 21 have judgments to evaluate submissions from the 2010 INEX Prove It workshop. We used these as the basis for our training set.

Inevitably, new systems pull up unjudged book pages in the top ten ranks. To handle these cases, we developed a judgment system with which lab members, including the authors, annotated pages as being *supportive*, *refutative*, or *relevant* in the case that a page was on topic, but not distinctly and completely supportive or refutative. The system displayed all fields of a topic, making the annotator as informed as possible. The fields are listed in the first column of Table 1 along with three examples of the field contents. The *info need* field usually describes what should be considered relevant, and the accessors were asked to abide by this. Some topics were tricky to judge, as in the case of Example 3 in Table 1 (Topic 2010015), where the broad focus is clearly on Italy, but the specific information being sought is inconsistent across the fields. In cases such as these, annotators were asked to interpret the information need as best they could and judge all pages relative to that interpretation.

Using the procedure described above, we augmented our training set with 535 additional relevance judgments. This covers many of the unjudged documents the systems retrieved in their top 10 lists for each topic.

Table 2. The results of several systems over the 21 training topics

	NDCG@10	
System	Stopped	Unstopped
$QLM_{\mu=1500}$	0.811	0.811
RM	0.751	0.701
$SDM+RM_{\mu=1500}$	0.755	0.751
$SDM_{\mu=1500}$	0.834	0.854
$SDM_{\mu=363}$	0.828	0.854
$PM_{l=100,\lambda=0.25}$	0.856	0.859
$PM_{l=50,\lambda=0.25}$	0.863	0.873

4 Results

In this section we discuss the performance of the models listed in Section 2 on the training data and our submitted models on the INEX 2011 test data.

4.1 Results over Training Topics

We evaluated over all 21 training topics. Each model considered only the *fact* field of each topic; when using the *query* field, our best models only outperform the better systems from last year's track by a small margin [1]. The substantial difference in performance between using the the two fields appears to stem from the poor representation of the information need in most topics' *query* field. Consider Example 2 in Table 1: the query "telescope" does not adequately describe the information need, which is the assertion that the primary function of a telescope is to magnify distant objects.

Table 2 reports the normalized discounted cumulative gain at rank 10 (NDCG@10) of the systems with and without stopwords removed. We binarized the graded relevance judgments such that the *supportive*, *refutative*, and *relevant* labels are conflated. The relevance models do not perform as well as the others, though this is partially due to not having enough judgments. Even if the unjudged documents are assumed relevant, SDM outperforms RM in the unstopped case, and RM only marginally improves over SDM in the stopped case. Setting μ to the average page length was not helpful for SDM, however, we entered $SDM_{\mu=363}$ as a submission because without a comprehensive parameter sweep, setting μ to the average page length is more principled than the Galago default.

The PM models outperform the others, with a passage size of 50 terms taking the lead. To understand why we choose $\lambda = 0.25$,[4] see Figure 1 (this only shows the variation with stop words removed). For both 50 and 100 term passages, it is clear that a value of λ in the $[0.20, 0.30]$ range, and specifically 0.25, is optimal. This places much of the final page score on SDM, but still gives a substantial amount of weight to the maximum QLM passage score.

[4] Our submissions' names suggest we used $\lambda = 0.025$, however this was a typo.

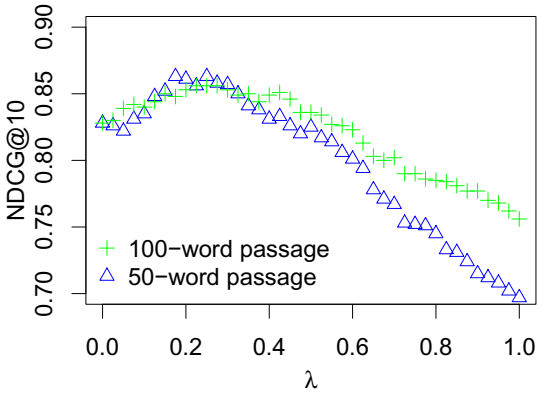


Fig. 1. A sweep over the λ parameter for the passage model. Smaller λ values mean more weight is given to the SDM score of the page, while higher values mean more weight is given to the highest scoring passage (using QLM). All queries were stopped.

SDM captures pieces of phrases in a fact, and these seem to be important given the results. PM adds the notion of tight proximity—a high passage score ideally applies to passages that are topical hot spots. By setting $\lambda = 0.25$, the model ranks pages that seem relevant overall and also contain topical hot spots higher than those that do not, which means the page's content is more important than the content of any single passage. Said differently, a page with many medium scoring passages will be ranked higher than a page with one high scoring passage. We performed a manual inspection of the retrieved documents and found that pages having only one high scoring passage are often non-relevant. The passage may make reference to an aspect of the topic, but provides no in depth information. Perhaps due to the nature of books, relevant sections tend to discuss topics over several paragraphs and even pages. Thus, the behavior of PM when $\lambda = 0.25$ is consistent with our observations of relevance within the data set.

4.2 Results over Test Topics

INEX participants provided a limited number of judgments for nine topics. These judgments cover only about 20% of the the top ten pages retrieved across the 18 submitted runs. The mean average precision (MAP), mean reciprocal rank (MRR), precision at 10 (P@10) and NDCG@10 are reported for each of our submissions in Table 3. The limited number of judgments is apparent in the lower NDCG@10 figures. The results suggest that removing stop words is detrimental, which is consistent with our findings with the training data. The two best performing runs are SDM with $\mu = 363$ and PM with 100 word passages

Table 3. The results of our submissions on the nine INEX 2011 test queries. Best results are shown in bold.

System	Stopped	MAP	MRR	P@10	NDCG@10
$SDM_{\mu=363}$	no	**0.2039**	0.3890	**0.1556**	**0.2768**
$SDM_{\mu=363}$	yes	0.1752	0.3220	**0.1556**	0.2437
$PM_{l=50,\lambda=0.25}$	no	0.2037	0.3889	**0.1556**	**0.2768**
$PM_{l=50,\lambda=0.25}$	yes	0.1743	0.3223	**0.1556**	0.2437
$PM_{l=100,\lambda=0.25}$	no	0.2035	**0.3894**	**0.1556**	**0.2768**
$PM_{l=100,\lambda=0.25}$	yes	0.1740	0.3236	**0.1556**	0.2447

and $\lambda = 0.25$, however, the 50-word passage model was not far behind. Overall, our models performed very well, but more judgments are necessary to fully understand the differences among them.

5 Summary

We considered several systems to retrieve supportive and refutative book pages for a given fact as part of the 2011 INEX Prove It task. We found that sequential dependence modeling (SDM) and passage-page interpolation (PM) perform best. Based on the behavior of these two systems and our observations of relevance from a manual inspection, relevant book pages tend to discuss the relevant material across many paragraphs. While PM attempts to model this to some degree, we believe that this phenomenon can be modeled in more powerful ways, which we leave to future work.

Acknowledgments. This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF CLUE IIS-0844226 and in part by NSF grant #IIS-0910884. 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. Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.): INEX 2010. LNCS, vol. 6932. Springer, Heidelberg (2011)
2. 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, pp. 120–127. ACM (2001)
3. Metzler, D., Croft, W.B.: A markov random field model for term dependencies. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2005, pp. 472–479. ACM, New York (2005)
4. Ponte, J., Croft, W.: A language modeling approach to information retrieval. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 275–281. ACM (1998)

The Book Structure Extraction Competition with the Resurgence Full Content Software at Caen University

Emmanuel Giguet and Nadine Lucas

GREYC Cnrs. Caen Basse Normandie University
BP 5186 F-14032 CAEN Cedex France
name.surname@unicaen.fr

Abstract. The GREYC participated in the Structure Extraction Competition, part of the INEX/ICDAR Book track, for the third time, with the Resurgence software. We used a minimal strategy primarily based on full-content top-down document representation with two then three levels, part, chapter and section. The main idea is to use a model describing relationships for elements in the document structure. Frontiers between high-level units are detected. The periphery center relationship is calculated on the entire document and then reflected on each page. The weak points of the approach are that level hierarchy is implicit, and dependent on named levels. It does not fit with the chapter and section levels reflected in the ground-truth. The strong points are that it deals with the entire document; it handles books without ToCs, and extracts titles that are not represented in the ToC (e. g. preface); it is tolerant to OCR errors and language independent; it is simple and fast. A test on sections was run after the competition to help understand the evaluation issues with more than two levels.

1 Introduction

The GREYC laboratory participated for the third time in the Book Structure Extraction Competition part of the INEX ICDAR evaluations in 2011 [1]. The extraction software Resurgence used at Caen University does not rely on the ToC pages but on the full content of the books. The experiment was conducted from pdf documents to ensure the control of the entire process. The document content is extracted using the pdf2xml software [2]. The original Resurgence software processes small documents, academic articles (mainly in pdf format) and news articles (mainly in HTML format) in various information extraction tasks and text parsing tasks [3].

In 2009, Resurgence handled only the chapter level [4] and in 2010 it handled part and chapter levels [5]. Surprisingly, better results were obtained when parts and chapters were evaluated irrespective of level. Since GREYC was the only participant in 2010, the experiment was reiterated in 2011 in runs 1 and 2 for part and chapter levels. We studied the effect of a complex hierarchy on the ground truth and on evaluation. In run 3, a test was made on three levels including sections with numbered series.

In the following, we explain our method on the 2011 ICDAR book corpus challenge. Results are compared in the two evaluation grids, ICDAR and link-based (Xerox) in section 3. In section 4 we discuss ways to correct our system and better handle sections.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 86–97, 2012.

After the competition, we tested the section level again with more success as expected. In the last section, we point at some inconsistencies or difficulties in the ground-truth constitution and make proposals for future competitions with an enhanced annotation tool.

2 A Differential Book Structure Extraction Method

2.1 Challenges

The size of the book corpus is the first challenge. Resurgence was modified in order to load the necessary pages only. The objective was to allow processing on usual laptop computers.

The fact that the corpus was OCR documents also challenged our original program that detects the structure of electronic academic articles. A new branch in Resurgence had to be written in order to deal with scanned documents. The document parsing principles were tested on two levels of the book hierarchy at a time, part (meaning here a book part including a number of chapters) and chapter. Experiments on sections in run 3 with a new method are also reported, and will be further explained in section 4 with a new corrected run outside the competition.

2.2 Strategy

The strategy in Resurgence is based on document positional representation. and does not rely on the table of contents (ToC). This means that the whole document is considered first. Then document constituents are considered top-down (by successive subdivision). with focus on the middle part (main body) of the book. The document is thus the unit that can be broken down ultimately to pages.

The main idea is to use a model describing relationships for elements in the document structure. The model is a periphery-center dichotomy. The periphery center relationship is calculated on the entire document and reflected on each page. The algorithm aims at retrieving the book main content bounded by annex material like preface and post-face with different layout. It ultimately retrieves the page body in a page, surrounded by margins [4].

We adopted the principle to get systematically down the book structure hierarchy one level at a time. For this experiment, we focused on part (if any) and chapter title detection, so that the program detects two levels. i. e. part titles and chapter titles in runs 1 and 2 as in 2010. The transition page between two parts or between two chapters is characterized in a sliding window of four pages as detailed in [5].

In run 3 some elaboration on title detection using “longitudinal” series at a given level was tempted as detailed below. In run 3 we also included three levels, chapter, part and section detection.

2.3 Title Extraction Strategy

Title extraction is conducted in four steps for all three levels. First, selection of would-be numbered titles; second, reconstruction of series names through creation of an equivalence table for each series; third, series validation through numbers; fourth, starting point detection.

2.3.1 Selection of Candidate Titles

A regular expression detects characters patterns:

a) sharing the same layout;
b) placed on the same line;
c) beginning with a capitalized word followed by a number (Arabic. roman or ordinal).

Note that in practice. extracted series may contain for example:

- CHAPTEE.
- THE.
- BOOK.
- Chapter

2.3.2 Series Names Reconstruction

Series names candidates are checked throughout the document. A global test checks if there are at least two successive series name candidates in the book for the same level (as derived from position and layout).

The "word" before the number, also called prefix, is kept in memory, if its frequency is above 1. The idea is to detect series names candidates as prefixes, such as *Chaptee, Book*, without being blocked by a strong expectation on a given wording. This is to avoid both OCR errors and misses when series wording varies from conventional use, with *Poem* or *Sermon* instead of *Chapter,* or *Book* instead of *Part*.

Thus, a series comprising some OCR errors like CHAPTER I ... CHAPTEK V … CHAPTEE XI is considered as a good text segment candidate provided the Levenshtein distance between two wordings is small (below 20%). The prefix variants will be considered as equivalent to the most frequent wording, thus CHAPTEK and CHAPTEE will be equivalent to CHAPTER. Note that in practice some extracted series may still be deemed incorrect, if there are more OCR errors than correct titles. This has no importance for the structuration task. A correction rule could be applied on the entire collection for search engines tasks, later on.

2.3.3 Series Name Validation

Once the series name is fixed, the numbers are checked with some tolerance. The idea is to find one or several grossly growing series in number with an equivalent series title, considered as a prefix. In the example below, some numbers are missing (typically first chapters are more difficult to detect). Some others have been sliced by return commands, so the series is awkward.

- CHAPTER: III IV V VI VII VIII IX X XII XIII XIV XV XVI XVII XVIII XIX XX XXI XXII XXIII XXIV XXV XXVI XXVII XXVIIL XXIX XXX XXXI XXXII XXXIII XXXIV II III IV V VI VII VIII IX X XL XII XIII XIV XV XVI XVII XVIII XIX XX XXI XXIV XXV XXV\

I XXVII XXVIII XXIX XXX XXXI XXXII II III IV V VI VII VIII IX X XL XII XIII XIV XV XVI XVII XVIII XIX XX XXL XXII XXIV XXV XXVI XXVII XXVIII II III IV V VI VII VIII IX X XII XIII XIV XV XVI XVIII XIX XX XXI XXII XXIII XXIV XXV XXVI XXVI\

I XXVIII XXIX XXX XXXI XXXII XXXIII XXXIV XXXV XXXVI XXXVII II III IV V VI VII VIII IX X XI XII XIII XV XVI XVII XVIII XIX XX XXL XXII XXIII XXIV XXV XXVI XXVII XXVIII XXIX XXX XXXI XXXII XXXIV XXXV XXXVI XXXVII XXXIX XL

However, increasing series are found no withstanding some holes or redundancies. One or more such series will be considered correct as a plausible level prefix, here chapter level series. The same will apply to a shorter series at another level, book parts.

- BOOK: II III IV IV V

On the contrary, some wordings selected as prefix at stage 1 will be forgotten, because they are not followed by a grossly growing series of numbers. It might have been a title such as *The second world war*.

- THE:

Last, a series of numbers without any increase will also be forgotten.

- Chapter: XX XVII XIX

2.3.4 Starting Point Detection

In order to find often overlooked chapters, mainly first chapters or sections, the starting point for titles series was established at the beginning of the main body, that is, after the ToC if any. Thus, a procedure to detect would-be ToCs was applied.

2.4 Calibrating the System

On the practical side, the team was interested in handling voluminous documents, such as textbooks and cultural heritage books. Working on the whole document requires the ability to detect and deal with possible heterogeneous layouts in different parts of the document (preface. main body. appendices). Layout changes can impact page formatting (e.g.. margin sizes. column numbers) as well as text formatting (e.g.. font sizes. text alignments) [6].

The standard page structure recognition has been improved by a better recognition of the shape of the body, which is not strictly rectangular in scanned books [5]. Line detection, standard line height and standard space height detection were also improved. They are important in our approach, because the standard line is the background against which salient features such as large blanks and title lines can be detected. The improvements in line computation improved the results in chapter detection as explained in [5].

However, the hierarchy consolidation was not implemented.

2.5 Experiment

The corpus provided in 2011 was similar in size to the 2009 one. It comprised 998 books (as compared with 1114 books in 2010 and 1000 in 2009, some empty) [1, 10].

The GREYC 2011 program detected only part and chapter titles in run 1 and 2. The top-down strategy and the highest levels in the book hierarchy were favoured because this is the most useful step when filtering large book collections, in text mining tasks for instance. Moreover, most if not all known techniques start from the lower levels [7]. Reasonable results can be obtained for those levels with existing programs once the relevant parts or chapters have been retrieved.

There was only one run to test section detection, run 3. However, due to a bug in document numbers, it ran astray. Run 4 was added after the competition to test the strategy explained in 2.2, at the section level as well. It will be discussed separately.

3 Results

3.1 General Results

The official results for 2011 are reproduced in Table 1, against a ground-truth of 513 books. GREYC missed one book of the ground-truth. It is at the fourth and last rank. The entire corpus was handled, with 60 misses. The very bad results in run 3 were due to a bug in document numbers.

Table 2 shows the F-link measure, with the same ranking.

Table 1. F-measure evaluation 2011 on 2011 ground-truth (513 books)

RunID	Participant	F-measure (complete entries)
MDCS	Microsoft Development Center Serbia	40.75%
Nankai-run1	Nankai University. China	33.06%
Nankai-run4	Nankai University	33.06%
Nankai-run2	Nankai University	32.46%
Nankai-run3	Nankai University	32.43%
XRCE-run1	Xerox Research Centre Europe	20.38%
XRCE-run2	Xerox Research Centre Europe	18.07%
GREYC-run2	GREYC University of Caen. France	8.99%
GREYC-run1	GREYC University of Caen. France	8.03%
GREYC-run3	GREYC University of Caen. France	3.30%

Table 2. F-link evaluation 2011 on 2011 ground-truth (513 books)

RunID	Participant	F-link
MDCS	Microsoft Development Center Serbia	65.1%
Nankai-run1	Nankai University, China	63.2%
Nankai-run4	Nankai University, China	63.2%
Nankai-run2	Nankai University	59.8%
Nankai-run3	Nankai University	59.8%
XRCE-run2	Xerox Research Centre Europe	58.1%
XRCE-run1	Xerox Research Centre Europe	57.6%
GREYC-run1	GREYC University of Caen, France	50.7%
GREYC-run2	GREYC University of Caen, France	50.7%
GREYC-run3	GREYC University of Caen, France	24.4%

3.2 Greyc Results Evolution

These results are compared with the GREYC official evaluation in 2009 best run and with 2010 in Table 3.

Table 3. Official evaluation 2009 to 2010 on the 2009 ground-truth (527 books)

Results 2009	**Precision**	**Recall**	**F-Measure**
Titles	19.83%	13.60%	13.63%
Levels	16.48%	12.08%	11.85%
Links	1.04%	0.14%	0.23%
Complete entries	0.40%	0.05%	0.08%
Entries disregarding depth	1.04%	0.14%	0.23%
Results 2010			
Titles	18.03%	12.53%	12.35%
Levels	13.29%	9.60%	9.34%
Links	14.89%	7.84%	7.86%
Complete entries	14.89%	10.17%	10.37%
Entries disregarding depth	10.89%	7.84%	4.86%

Table 4. GREYC 2010 and 2011 evaluation with Xerox linked-based metrics

	XRCE Link-based Measure			
	Links			**Title accuracy** (for valid links)
	Precision	**Recall**	**F1**	
GREYC 2010	63.9%	39.5%	42.1%	47.6%
GREYC 2011 - run2	65.2%	49.9%	50.7%	46.2%
GREYC 2011 – run3	32.5%	24.5%	24.4%	31.1%

Table 4 shows the evaluation based on links and initially provided by Xerox Research Center Europe (XRCE).

The 2011 results slightly outperform the 2010 results as expected for chapter detection. This is mainly explained by improvements in the system calibration. Little gain is obtained from part detection, as in 2010. This is due to the fact that most book parts are not signalled in the ground-truth. Even if it were, the number of parts is low (and even often null in individual books), as compared to the total number of titled sections to be found throughout the collection. But the main interest in this year evaluation was to assess the effect of multilevel description with series. The failure of run 3 was a bad blow. It was found that chapter and sections are the two levels on which annotation focuses, being mainly based on ToCs.

4 Discussion

GREYC was the only candidate in 2010, so comparison with others was not possible. It was worth re-evaluating results on roughly the same corpus, and the same method, through runs 1 and 2. Moreover, we tested a new method in run 3, based on numbered series, and including sections. It is level independent but not quite lexicon-free. It was corrected in post run 4, to evaluate the benefits of this strategy.

The ground-truth annotation was not easy since we had to browse entire books, which took an enormous time with slow response delays to “turn” pages. We worked with a Mac, which could be a plea. It was not possible to establish two levels and save them explicitly as such through the menu. Therefore, the reference cannot be deeper than two levels, try as we may. As far as we saw, chapters and sections were the only levels used by other participants. Parts including chapters would as a consequence be judged as false when detected, as well as titled sub-sections in chapters.

4.1 Reflections on the Experiment

4.1.1 Extra Run

GREYC corrected a bug concerning document id numbers in Run 3 including section level and using series. The corrected run is called Corrected Run-4 and it obtained significantly better results.

Table 5 shows results given for the official best GREYC run for two levels (part and chapter) in run 2, and the best results for three levels (part. chapter and section) with document number correction in the post competition GREYC-Corrected Run 4. Fusion of position clues with series validation proved efficient.

Table 5. Comparison of two-level and three-level results for GREYC 2011

	F1 Link	F1 Inex Link
GREYC-run 2	50.7%	10.8%
GREYC corrected run 4	58.8%	20.1%

However, the need to propagate these principles to all the sub-levels of the book hierarchy (such as sub-sections) was not felt. This is because the subsections are seldom accompanied by a prefix, which is part of the recognition pattern used by GREYC to extract title series. As a consequence, many numbered but un-titled sections and subsections will go unnoticed. A different strategy has to be found for deep subdivisions. Moreover, the subsections are seldom kept in ToCs and the ground-truth also ignores them.

4.1.2 Comparison

On the scientific side, some strong points of the Resurgence program were ascertained. They are based on relative position and differential principles. The advantages are the following:

- The program deals with the entire document body, not on the table of contents;
- It handles books without table of contents (ToC), and titles that are not represented in the ToC (e. g. preface). It would be most welcome if the annotated corpus could be checked directly inside the book when looking for errors;
- It is dependent on typographical position, which is very stable in the corpus, despite heterogeneous domains and styles;
- It is not dependent on lexicon, or very little in run 3. Hence it is tolerant to OCR errors and it is language independent;
- Last, it is simple and fast.

The advantage of using the book body is clear when comparing two datasets, books without ToC and books with ToC [6, 9]. The difference is clearer in the GREYC case with the link-based measure.

Table 6. Comparison of 2009 results on two books datasets after [6]

	whole dataset (precision / recall)	no-ToC dataset (precision/ recall)
MDCS	65.9 / 70.3	0.7 / 0.7
XRCE	69.7 / 65.7	30.7 / 17.5
NOOPSIS	46.4 / 38.0	0.0 / 0.0
GREYC-1C 2009	59.7 / 34.2	48.2 / 27.6

Another advantage is robustness. Since no list of memorized forms is used, but position and distribution instead, fairly common strings are extracted, such as CHAPTER or SECTION, but also uncommon ones, such as PSALM or SONNET. When chapters have no numbering and no explicit mention such as *chapter*, they are found as well, for instance a plain title stating "Christmas Day".

Resurgence took advantage on numbering of titles series through many steps in 2011: since numbers are an important source of OCR errors, a tolerant pattern recogniser is used. This approach reflects an original breakthrough to improve robustness and proves very useful to generate ToCs to help navigate digitized books when none was provided in the printed version (20% of the corpus).

4.2 Reflections on Evaluation Measures

Concerning evaluation rules. the very small increment in quantified results did not reflect our qualitative assessment of a significant improvement in numbered series.

Generally speaking, the ground-truth is still very coarse and it mostly relies on automated results depending on the ToC [9. 1]. If the ToC is the reference, it is an error to extract prefaces, for instance, because they generally do not figure in ToCs. In the same way, most ToCs do not reflect the whole hierarchy of sections and subsections, but skip lower levels. The participants using the book body as main reference are penalized if they extract the whole hierarchy of titles as it appears in the book, when the ToC represents only higher levels.

For all participants, accuracy on titles seems to be a thorny question, because there is a huge difference in title accuracy as calculated by INEX organizers from the retrieval of the wording, and title accuracy as calculated by XRCE from the links [1, 7]. In the INEX08-like measure on accuracy for title and level provided by XRCE, the figures decrease while precision and recall grow.

A test was made to evaluate level accuracy, since proceeding one level at a time allowed a relevance check on this measure. In 2009 GREYC calculated only chapters and the level accuracy was high, 73.2. in the GREYC results, after correction on the document id bug. Scores in level accuracy in 2010 were calculated with part and chapter level information and then without part and chapter level information to check consistency (Table 7).

Table 7. GREYC link-based evaluation with and without level information against the 2009 ground-truth as compared with 2011 evaluation and ground-truth

	XRCE Link-based Measure				**Inex08 like Accuracy**	
	Links			**Accuracy** for valid links		
	Precision	**Recall**	**F1**	*Title*	*Title*	*Level*
GREYC-1C 2009	59.7	34.2	38.0	13.9	42.1	73.2
GREYC 2010	64.4	38.9	41.5	47.6	22.3	64.2
GREYC 2010 without level info	64.4	38.9	41.5	47.6	22.3	**77.9**
GREYC 2011 run 2	65.2	49.9	50.7	46.2	21.9	**80.4**

In 2011, title accuracy was lower but level accuracy was slightly better. GREYC reached the best official relative level accuracy among all participants with a 80.4% score, followed by MDC at 79.2%, as shown in Table 8.

Since GREYC was the only candidate working from the actual book body layout and not after the ToC, results suffered from the fact that ToC when present — in 80% of the cases — is used as the baseline reference in the ground-truth [1]. However, there are significant differences between ToC and book titles as reported in [5, 6].

Table 8. 2011 alternative link-based evaluation against the 2011 cleaned ground-truth (513 books), compared with Inex-like accuracy depending on title recognition

	F-link	Titl-acc	RelLevel-accuracy	F~Inex Link	Inex Titl-acc	Level-acc
MDCS	65.1%	83.7%	79.2%	47.6%	69.1%	79.6%
NANKAI-1	63.2%	74.4%	76.3%	40.9%	54.2%	77.2%
NANKAI-2	59.8%	75.9%	75.5%	40.1%	56.6%	76.4%
NANKAI-3	59.8%	75.9%	75.5%	40.1%	56.5%	76.3%
NANKAI-4	63.2%	74.4%	76.3%	40.9%	54.2%	77.2%
UNICAEN-1	50.7%	46.2%	61.4%	10.8%	21.9%	61.3%
UNICAEN-2	50.7%	46.2%	80.4%	10.8%	21.9%	80.4%
UNICAEN-3	24.4%	31.1%	64.0%	4.2%	11.2%	63.9%
XRCE-1	57.6%	60.9%	78.6%	24.8%	43.8%	78.6%
XRCE-2	58.1%	63.7%	77.9%	23.5%	40.1%	77.9%

Table 9. GREYC 2011 runs in the two measures (level accuracy in bold)

	F-link	Titl-acc	RelLvl-acc	F~Inex Link	Inex Titl-acc	Level-acc
UNICAEN-1	50.7%	46.2%	61.4%	10.8%	21.9%	61.3%
UNICAEN-2	50.7%	46.2%	**80.4%**	10.8%	21.9%	**80.4%**
UNICAEN-3	24.4%	31.1%	64.0%	4.2%	11.2%	63.9%
UNICAEN-Corrected 4	56.5%	56.8%		20.1%	33.1%	**78.4%**

The scores for corrected run 4 were calculated using the download package [8] but the new item Relative Level accuracy was not included.

5 Proposals

The bias introduced by a semi-automatically constructed ground-truth was salient as can be seen in the example above, where split words or added *pp.* at the end of the entry illustrate poor quality against human judgment. Manually corrected annotation is still to be checked to improve the ground-truth quality. As mentioned in [1] quantitative effort is also needed, but it is time-consuming. Crowdsourcing was considered a better solution to minimize annotator's discrepancies [10].

However, it might not be realistic to expect a clean unique reference for a large book collection. It might be better to handle parameters according to the final aim of the book processing, such as navigation or information filtering. Thus known automatic biases might be countered or even valued in the performance measure according to real use.

It would be very useful to provide results by normalized title depth (level) as suggested by [5, 7], because providing complete and accurate results for one or more levels would be more satisfying than missing some items at all levels. It is important to get coherent and comparable text spans for many tasks, such as indexing, helping navigation or pre-processing for text mining.

The reason why the beginning and end of the titles are overrepresented in the evaluation scores is not clear and a more straightforward edit distance for extracted titles should be provided.

One simple idea used in the 2011 evaluation was to consider equally results for titles matching with either the ToC or the book body, with or without a prefix such as *Chapter* [1].

Despite shortcomings, mostly due to early stage development, the book structure extraction competition was very interesting. The corpus provided for the INEX /ICDAR Book track is the best available corpus offering full books at document level [1, 9, 10]. Although it comprises mostly XIXth century printed books, it is very valuable, for it provides various types of layout. Besides, this corpus meets our requirements for electronic use of patrimonial assets. The ground-truth is manually corrected, so that the dataset is easier to work with than the dataset provided by [11].

Some efforts should be exerted to improve the interface used to annotate books, so that the whole title hierarchy can be clearly and conveniently marked. Accordingly, accurate level measures reflecting the human judgement could trigger better automatic recognition.

References

1. Doucet, A., Kazai, G., Meunier, J.-L.: ICDAR 2011 Book Structure Extraction Competition. In: 11th International Conference on Document Analysis and Recognition (ICDAR 2011), pp. 1501–1505 (2011)
2. Giguet, E., Lucas, N., Chircu, C.: Le projet Resurgence: Recouvrement de la structure logique des documents électroniques. In: JEP-TALN-RECITAL 2008 Avignon (2008)
3. Déjean, H., Giguet, E.: pdf2xml open source software, `http://sourceforge.net/projects/pdf2xml/` (last update February 25, 2011; last visited February 2012)
4. Giguet, E., Lucas, N.: The Book Structure Extraction Competition with the Resurgence Software at Caen University. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 170–178. Springer, Heidelberg (2010)
5. Giguet, E., Lucas, N.: The Book Structure Extraction Competition with the Resurgence Software for Part and Chapter Detection at Caen University. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 128–139. Springer, Heidelberg (2011)

6. Déjean, H., Meunier, J.-L.: Document: a useful level for facing noisy data. In: 4th Workshop on Analytics for Noisy Unstructured Text Data (AND 2010), Toronto, Canada, pp. 3–10 (2010)
7. Déjean, H., Meunier, J.-L.: Reflections on the INEX structure extraction competition. In: 9th IAPR International Workshop on Document Analysis Systems (DAS 2010), pp. 301–308. ACM, New York (2010), doi:10.1145/1815330.1815369
8. Source forge, https://sourceforge.net/projects/inexse/
9. Doucet, A., Kazai, G., Dresevic, B., Uzelac, A., Radakovic, B., Todic, N.: Setting up a Competition Framework for the Evaluation of Structure Extraction from OCR-ed Books. International Journal of Document Analysis and Recognition (IJDAR) 14(1), 45–52 (2010)
10. Kazai, G., Koolen, M., Kamps, J., Doucet, A., Landoni, M.: Overview of the INEX 2010 Book Track: Scaling Up the Evaluation Using Crowdsourcing. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 98–117. Springer, Heidelberg (2011)
11. Vincent, L.: Google Book Search: Document understanding on a massive scale. In: 9th International Conference on Document Analysis and Recognition (ICDAR 2007), pp. 819–823. IEEE (2007)

TOC Structure Extraction from OCR-ed Books

Caihua Liu[1,⋆], Jiajun Chen[2,⋆],
Xiaofeng Zhang[2,⋆], Jie Liu[1,⋆⋆], and Yalou Huang[2]

[1] College of Information Technical Science, Nankai University, Tianjin, China 300071
[2] College of Software, Nankai University, Tianjin, China 300071
{liucaihua,chenjiajun,zhangxiaofeng}@mail.nankai.edu.cn,
{jliu,ylhuang}@nankai.edu.cn

Abstract. This paper addresses the task of extracting the table of contents (TOC) from OCR-ed books. Since the OCR process misses a lot of layout and structural information, it is incapable of enabling navigation experience. A TOC is needed to provide a convenient and quick way to locate the content of interest. In this paper, we propose a hybrid method to extract TOC, which is composed of rule-based method and SVM-based method. The rule-based method mainly focuses on discovering the TOC from the books *with* TOC pages while the SVM-based method is employed to handle with the books *without* TOC pages. Experimental results indicate that the proposed methods obtain comparable performance against the other participants of the ICDAR 2011 Book structure extraction competition.

Keywords: table of contents, book structure extraction, xml extraction.

1 Introduction

Nowadays many libraries focus on converting the whole libraries by digitizing books on an industrial scale and this project is referred as '*digital libraries*'. One of the most important tasks in digital libraries is extracting the TOC. A table of contents (TOC) is a list of TOC entries each of which consists of three elements: title, page number and level of the title. The intention of extracting TOC is to provide a convenient and quick way to locate content of interest. To extract the TOC of the books, we are faced with several challenges. First, the books are various in forms, since the books come from different fields and there are kinds of layout formats. A large variety of books increases the difficulty of utilizing a uniform method to well extract the TOC. Taking the poems for example, some poems contain TOC pages while some not. The alignment of poems may be left-aligned, middle-aligned and right-aligned. Second, due to the limitation of OCR technologies, there are a certain number of mistakes. OCR mistakes also cause trouble in extracting TOC, especially when some keywords such as 'chapters', 'sections', etc., are mistakenly recognized.

⋆ The first three authors make equal contributions.
⋆⋆ Corresponding author.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 98–108, 2012.

Many methods have been proposed to extract TOC, most of which are published in INEX[1] workshop. Since 80% of these books contain table of contents, MDCS [1] and Noopsis [2] took the books with table of content into consideration. While the University of Caen [3] utilized a four pages window to detect the large whitespace, which is considered as the beginning or ending of chapters. XRCE [4] segmented TOC pages into TOC entries and used the references to obtain page numbers. XRCE also proposed a method trailing whitespace.

In this paper, we propose a hybrid method to extract TOC, since there are two types of data. 80% of the books contain TOC pages and the remaining do not. This two situations are considered via rule-based method and SVM-based method respectively. For books containing TOC pages, some rules are designed to extract these TOC entries. The rules designed are compatible with the patterns of most books, which is also demonstrated in the experiments. For books without TOC pages, a SVM model is trained to judge whether one paragraph is a title or not. A set of features is devised for representing each paragraph in the book. These features also do not depend on knowledge of TOC pages. Using these features and the machine learning method we can extract TOC entries, whether the book has an TOC page or not. To better organize these TOC entries, the level and the page number of each TOC entry locating are also extracted, besides these TOC entries themselves.

The paper is organized as follows. In section 2, we give a description of the previous works about the extraction of TOC. An introduction about the books and the format of these data is presented in section 3. The main idea of our works to extract TOC entries is shown in section 4. In section 5, we locate the target page for each TOC entry. We assign levels for TOC entries in section 6. Finally, experiments and a short conclusion are displayed.

2 Related Works

In the application of digital libraries, there are four main technologies, information collecting, organizing, retrieving and security. Organizing data with XML is the normal scheme, especially when we are faced with large scales of data. A information retrieval workshop named INEX has been organized to retrieve information from XML data. In 2008, BSE(Book Structure extraction) was added to INEX, whose purpose is to evaluate the performance of automatic TOC structure extraction from OCR-ed books.

MDCS [1] and Noopsis[2] focus on books containing TOC pages. Except for locating the TOC entries, they make no use of the rest of the books. MDCS employees three steps to extract TOC entries. Firstly, they recognize TOC pages. Secondly, they assign a physical page number for every page. Finally, they extract the TOC entries via a supervised method relying on pattern occurrences detected. MDCS's method depends on the TOC pages and it can not work for books without TOC pages. University of Caen's [3] method did not rely on the content page, the key hypothesis of which is that the large whitespace is the

[1] http://www.inex.otago.ac.nz/

beginning of one chapter and the ending of another chapter. So they utilized a four pages window to detect the whitespace. However, it works well on high-level title but the lower title can not be recognized well. Xerox Research Center Europe [4] used four methods to extract the TOC entries. First two are based on TOC pages and index pages. The third method is similar to MDCS, which also defines some patterns while the last method trails the whitespace like University of Caen does. The method proposed by us is more efficient than others, since we directly extract TOC entries by analyzing the TOC pages for books with TOC pages. Due to the diversity of books without TOC pages, it's difficult to find a uniform rule or pattern to extract the TOC entries. To these issues, we perform an automatic learning method for extracting TOC entries.

3 The Architecture of the Hybrid Extracting Method

In this section, we will give a description of the architecture of our method. Since 80% of the books contain TOC pages while the remaining do not, we consider these two situations respectively. One more crucial reason is that the TOC pages contain a lot of hidden structural contents. The well using of the TOC pages can help improve the extracting performance. In addition, for books with TOC pages, directly extracting TOC from the TOC pages performs better than extracting TOC from main text.

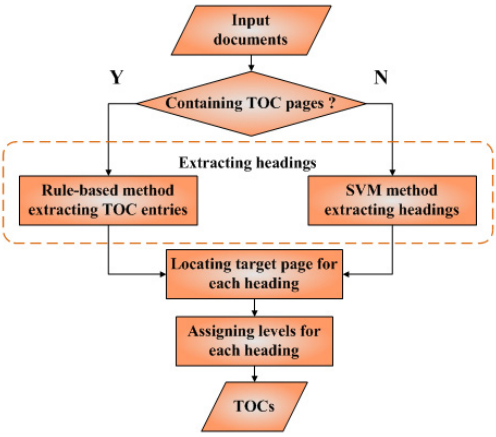


Fig. 1. The flow chart of our hybrid extracting system

As previously stated, each TOC entry contains three parts: title, page number and level of the title. As shown in Figure 1, the extracting process is conducted with the following steps. (1) A judgement is conducted to separate the books into two parts via whether containing TOC pages. (2) Extracting each TOC

entry and obtaining the title, page number. Since there are two types of data, we extract the TOC from them respectively. For books containing TOC pages, a rule-based method is proposed to extract these TOC entries from the TOC pages. For books without TOC pages, a SVM method is introduced to achieve the purpose of extracting TOC. (3) Locating the target page for each TOC entry. (4) Assigning levels for these extracted TOC entries. A simple relationship between these TOC entries can be obtained by this step.

After these steps, each TOC entry has been extracted. we also obtain the target pages and the organizational structure of these TOC entries. Then the TOC is outputted as the predefined format. Until now, all of the works to extract TOC has been accomplished. And the specific methods to conduct this three steps is stated in the following sections.

4 Extracting TOC Entries

In this section, we focus on the two methods to extract TOC entries. The following two sections will give a specific introduction of these two methods respectively.

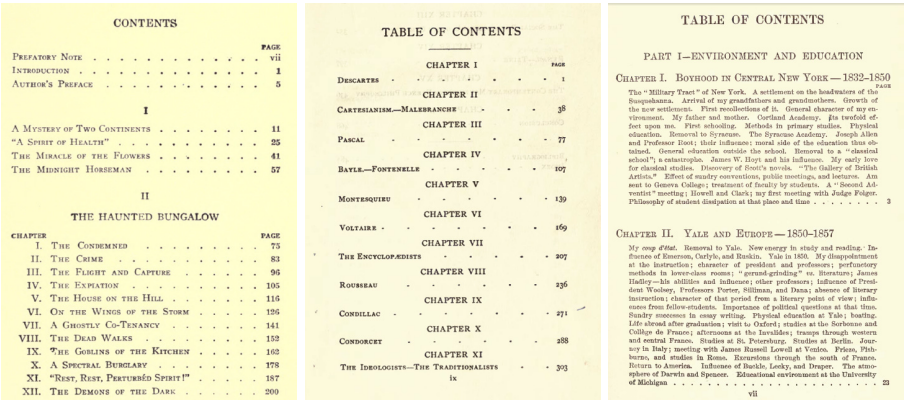
4.1 Extracting with TOC Pages

We extract the TOC from the original TOC page in the book with TOC pages. This task is divided into two steps: locating the TOC pages of the book and extracting TOC entries from the TOC pages.

Locating TOC Pages. If a book contains TOC pages, we extract the TOC entries from the TOC pages directly. Naturally, the TOC pages start with key words such like '*Contents*' and '*Index*', and the TOC page contains many lines ending with numbers. We can use these features to locate the beginning of contents pages. Since most books have headers, the pages are ascertained to be TOC pages, if there are key words like '*Contents*' and '*Index*' appearing at the beginning of the page, or if there is a considerable number of lines ending with numbers. Naturally, TOC pages usually appear in the front part of a book, as a result, we only need to consider the first half of the book to accelerate the process.

Extracting Contents Entries. TOC entries in books vary greatly in different books. As is shown in Figure 2, some entries like Figure 2(a) occupy only one line, while some entries Figure 2(b) occupy several lines. Some entries Figure 2(c) are divided into multi-lines, of which the first line is text, the second line is logical page number and the third line is introduction of the section.

To extract each TOC entry, we need to obtain the beginning and the ending of each TOC entry. The following rules are conducted to identify the beginning of TOC entries.



CONTENTS

PAGE
PREFATORY NOTE vii
INTRODUCTION 1
AUTHOR'S PREFACE 5

I

A MYSTERY OF TWO CONTINENTS 11
"A SPIRIT OF HEALTH" 25
THE MIRACLE OF THE FLOWERS 41
THE MIDNIGHT HORSEMAN 57

II

THE HAUNTED BUNGALOW

CHAPTER PAGE
I. THE CONDEMNED 75
II. THE CRIME 83
III. THE FLIGHT AND CAPTURE 96
IV. THE EXPIATION 105
V. THE HOUSE ON THE HILL 116
VI. ON THE WINGS OF THE STORM 126
VII. A GHOSTLY CO-TENANCY 141
VIII. THE DEAD WALKS 152
IX. THE GOBLINS OF THE KITCHEN 162
X. A SPECTRAL BURGLARY 178
XI. "REST, REST, PERTURBÉD SPIRIT!" 187
XII. THE DEMONS OF THE DARK 200

TABLE OF CONTENTS

CHAPTER I PAGE
DESCARTES 1
CHAPTER II
CARTESIANISM.—MALEBRANCHE 38
CHAPTER III
PASCAL 77
CHAPTER IV
BAYLE.—FONTENELLE 107
CHAPTER V
MONTESQUIEU 139
CHAPTER VI
VOLTAIRE 169
CHAPTER VII
THE ENCYCLOPÆDISTS 207
CHAPTER VIII
ROUSSEAU 236
CHAPTER IX
CONDILLAC 271
CHAPTER X
CONDORCET 288
CHAPTER XI
THE IDEOLOGISTS—THE TRADITIONALISTS . . . 303
ix

TABLE OF CONTENTS

PART I—ENVIRONMENT AND EDUCATION

CHAPTER I. BOYHOOD IN CENTRAL NEW YORK—1832–1850
PAGE
The "Military Tract" of New York. A settlement on the headwaters of the Susquehanna. Arrival of my grandfathers and grandmothers. Growth of the new settlement. First recollections of it. General character of my environment. My father and mother. Cortland Academy. Its twofold effect upon me. First schooling. Methods in primary studies. Physical education. Removal to Syracuse. The Syracuse Academy. Joseph Allen and Professor Hoot; their influence; moral side of the education thus obtained. General education outside the school. Removal to a "classical school"; a catastrophe. James W. Hoyt and his influence. My early love for classical studies. Discovery of Scott's novels. "The Gallery of British Artists." Effect of sundry conventions, public meetings, and lectures. Am sent to Geneva College; treatment of faculty by students. A "Second Adventist" meeting; Howell and Clark; my first meeting with Judge Folger. Philosophy of student dissipation at that place and time 3

CHAPTER II. YALE AND EUROPE—1850–1857

My coup d'état. Removal to Yale. New energy in study and reading. Influence of Emerson, Carlyle, and Ruskin. Yale in 1850. My disparagement at the instruction; character of president and professors; perfunctory methods in lower-class rooms; "ground-grinding" vs. literature; James Hadley—his abilities and influence; other professors; influence of President Woolsey, Professors Porter, Silliman, and Dana; absence of literary instruction; character of that period from a literary point of view; influences from fellow-students. Importance of political questions at that time. Sundry successes in essay writing. Physical education at Yale; boating. Life abroad after graduation; visit to Oxford; studies at the Sorbonne and Collège de France; afternoons at the Invalides; tramps through western and central France. Studies at St. Petersburg. Studies at Berlin. Journey in Italy; meeting with James Russell Lowell at Venice. Prices, Pitchburns, and studies in Rome. Excursions through the south of France. Return to America. Influence of Buckle, Lecky, and Draper. The atmosphere of Darwin and Spencer. Educational environment at the University of Michigan 23

vii

Fig. 2. A variety of Contents structure. (a) Each TOC entry occupy one line; (b) Each TOC entry occupy several lines; (c) Each TOC entry occupy multi-lines.

1. if current line starts with key words like '*Chapter*', '*Part*', '*Volume*' or '*Book*' etc.
2. if current line starts with numbers or Roman numerals.
3. if last line ends with numbers or Roman numerals.

The start of a new TOC entry is the end of last TOC entry, so we can easily construct the following rules to identity the end of TOC entries.

1. if next line starts with key words like '*Chapter*', '*Part*', '*Volume*' or '*Book*' etc.
2. if next line starts with numbers or Roman numerals.
3. if current line ends with numbers or Roman numerals.

The above rules can handle most of the situations, however, some TOC entries such as multi-lines can not be well extracted using only those rules. To these issues, a new rule is added. If the last line does not have key words like '*Chapter*' and Roman numerals etc. that obviously separate contents items and the formats of current line and last line are very different, delete line information collected before.

The new rule treats current line as the start of a new TOC entry, and delete stored lines formerly should be treated as part of current TOC entry. The difficulty of this rule lies in the quantitative description of differences between two lines. Our approach only consider the relative_font_size.

$$relative\_font\_size = \frac{currentlinefontsize - averagefontsize}{maximumfontsize - avergefontsize} \quad (1)$$

Where the value of font_size is the average height of words in one line, and the height is computed by the position of words. If the '*relative_font_size*' of the two lines are very different and greater than some pre-set thresholds, these two lines will not be treated as one TOC entries.

4.2 Extracting without TOC Pages

Considering of establishing a uniform model for all the two types of books, we conduct a automatic method to label training data in the favor of TOC. It is expected that the SVM[2] model can handle both this two types data well. However, the method performs worse than the rule based method, so the SVM method is only utilized on books without TOC pages. It comes into the following steps: (1) extracting the features of each paragraph and labeling them(2) training the RBF-SVM to classify every paragraph.

Features. Through observing data set, some obvious features can be employed to identify a heading, as shown in table 1. Though we get the eight features, it happens that some common paragraphs have one or more features. For example, the page header is much more similar to the heading, so this will confuse the classifier. Commonly the page header always has the same content with the title, while it has a lot of duplications. So we use post-process method to delete the duplication and make the first page header that has the same content with others as the title. Another example, the title page (this page only contains a title, and in the next page it also starts with this title in the top) has also some of the features, what's more the effect of the features is more obvious than the title at times. So we must do some efforts to solve the problem. According to the title page, we set a threshold to judge whether a page is title page. It is means that if most of the paragraphs in the page are recognized as title, we think it is a title page.

Table 1. Features designed for books without TOC pages

Feature ID	Discription
1	Proportion of Capital Letters
2	Font Size
3	Left End position of a Paragraph
4	Right End Position of a Paragraph
5	Space between Paragraph
6	Line Number of a Paragraph
7	Average Number of words in Each Line of a Paragraph
8	The y-coordinate of a Paragraph Start

Recognizing the TOC Entries. We use SVM to identify whether one paragraph is TOC entry. Since many normal paragraphs are predicted as positive, a further analysis is made on these data. There are four situations to consider: the title we expect, page header, the misrecognized paragraph and the spot or handwritten note in the book which can be OCRed. In order to get a much higher performance, a post-process is conducted. And the following principles are devised to delete some of the positive ones.

[2] http://www.cs.cornell.edu/people/tj/svm_light.html

1. If there are less capital letters in a paragraph than a threshold.
2. If a paragraph only contains a letter, and it is not Roman number as well.
3. If a paragraph is similar to others,then keep the first one and delete the others.
4. If there are more than two positive paragraphs(paragraphs predicted by SVM as headings).

5 Locating Target Page

After extracting TOC entries, we need to ascertain where the entries actually locate for navigation purpose. The page numbers shown in TOC are the logical numbers. While the physical number shows the actual page number in the whole document. Hence the matching of physical page number and logical page number is expected to help users navigate over the whole document.

To match the physical and the logical page numbers, the logical numbers for every page are needed to be extracted first. Commonly, the logical number appears in the headers or footers. However, logical pages extracted in this way are not prefect enough, as some pages may indeed do not have logical page numbers or maybe an OCR error makes the logical page numbers not recognized correctly, so we need to deal with those omissions and errors. So a remedy is conducted to obtain the complete page numbers. First, fill the vacancies of pages without page numbers using the following method. If the physical page i and j ($j > i$) have logical page $L(i)$ and $L(j)$ respectively, and logical page numbers of pages between i and j are absent, at the same time, if $L(j) - L(i) = j - i$, fill the vacancies of logical page numbers for those page between i and j.

Whereas, there are still some pages with physical numbers which can not find logical number. For these physical pages we set them to 0. And then the logical page numbers of the extracted contents items are replaced to physical page numbers. For these physical pages labeled '0', We first find the the maximum logical page number smaller then current logical page number, and match with physical page number. Then from this maximum logical page number and forward we use text match to find the first page that contains the text of current TOC entry, and use the physical page number of this page as the physical page of current TOC entry.

6 Ranking Levels for TOC Entries

The final step is to rank those extracted TOC entries. Via the analysis of the data, we find that most of the content entries contain the key term '*Book*','*Volume*','*Part*', '*Chapter*' and so on. While information like arabic numerals and roman numerals can also be utilized to assign the level for content entries. So we pre-define five levels to arrange the levels of every contents entry.

1. First level: containing key words '*Part*', '*Volume*' and '*Book*' etc.
2. Second level: containing keywords '*Chapter*' and '*Chap*' etc.
3. Third level: containing keywords '*Section*' and '*Sect*' etc.
4. Forth level: containing Arabic numerals and Roman numerals or keywords like '*(a)*' and '*a*'.
5. To be ascertained level: other TOC entries that do not have above mentioned features while its level depends on their neighbors' contents, for example, previous rank.

A specific statistics on randomly selecting 100 books from the ICDAR 2011 dataset is shown in Figure 3. 70% of TOC entries contain keywords 'chapter' etc. and books with keyword 'Section' only occupy a small proportion of 100 books. More obvious is that 90% of the books correspond to our definition of levels. We first scan the whole content entries and assign levels for every entry

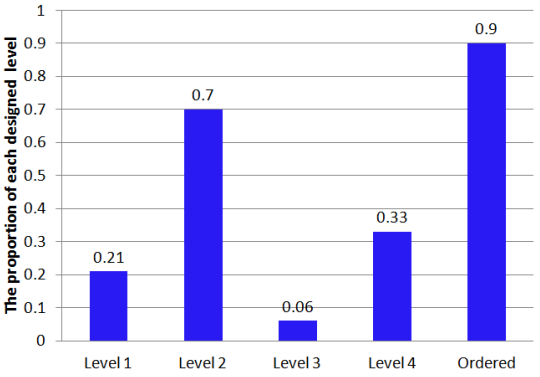


Fig. 3. The percent of each level and 'Ordered' means that the level of TOC corresponds to the level we defined

by the rule pre-defined above. Most of these entries are all assigned levels, only these entries without any characteristics left. A statistics has been conducted by us and it demonstrates that these left entries have a higher probability of the same level as the previous one, therefore, the levels for these left entries are assigned via this idea.

7 Experiments

In order to measure the performance of our method, we conduct experiment on two datasets. One is the 1000 books provided by the Book structure extraction competition, while the other is ICDAR 2009 competition dataset. The pdf and DjVuXML format of the books are both provided in these two datasets.

To give a full evaluation of the performance, three evaluate criterions are considered on five aspects. The evaluation measures are: precision, recall and

F-measure. And the 5 aspects are: (1) Titles, which evaluates whether the titles we obtained are sufficiently similar to the titles in the ground truth; (2) Links, which is correctly recognized if the TOC entries recognized by our method link to the same physical page in the ground truth. (3) Levels, which means whether established level of the title is at the same depth in the ground truth. (4) Complete except depth, which represents the title and the link are both right. (5) Complete entries, which is considered right only when all of these three items are right.

7.1 Experiments on the Whole Dataset

The experiments listed in this section are the public experimental results published by the official organizing committee of ICDAR. The performance of our method on the five aspects are reported in table 2. The performance comparison between our method with other participants of ICDAR is presented in Figure 4. It can be seen that MDCS outperforms others, but it is a TOC based method and it can not deal with books without table well. We rank second in the completion, however, we are capable of processing those books without contents. To address these issues, our method is comparable to others.

Table 2. The performance of our method on five evaluation aspects conducting on ICDAR 2011 dataset

Items	Precision	Recall	F-Measure
Titles	47.99%	45.70%	45.20%
Links	44.03%	41.44%	41.43%
Level	37.91%	36.84%	36.08%
Entries disregarding depth	44.03%	41.44%	41.43%
Complete entries	34.80%	33.28%	33.06%

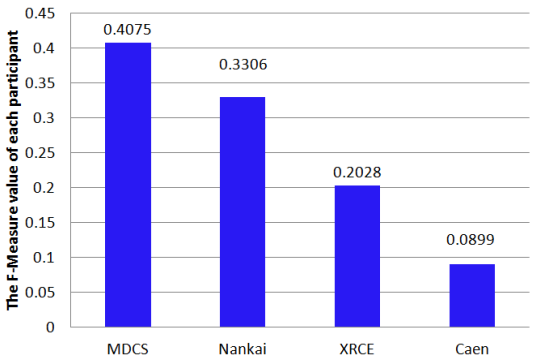


Fig. 4. The public results of ICDAR 2011 book structure extraction competition. It is conducted on the whole dataset and we rank second.

7.2 Experiments on Books without Table of Contents

To evaluate the ability of processing books without TOC pages, we conduct experiments on books without TOC pages in 2009 and 2011 ICDAR dataset respectively. The training data for SVM is obtained from the books with TOC pages and we try to learn how TOC entries look like. Since the number of normal context is much larger than the number of TOC entries, so we use all of the TOC entries and randomly select the same number of normal text.

Table 3. The performance of our method on five evaluation aspects conducting on ICDAR 2009 dataset

Items	Precision	Recall	F-Measure
Titles	14.85%	23.64%	14.38%
Links	10.88%	16.17%	10.71%
Level	11.78%	19.62%	11.40%
Entries disregarding depth	10.88%	16.17%	10.71%
Complete entries	8.47%	13.07%	8.43%

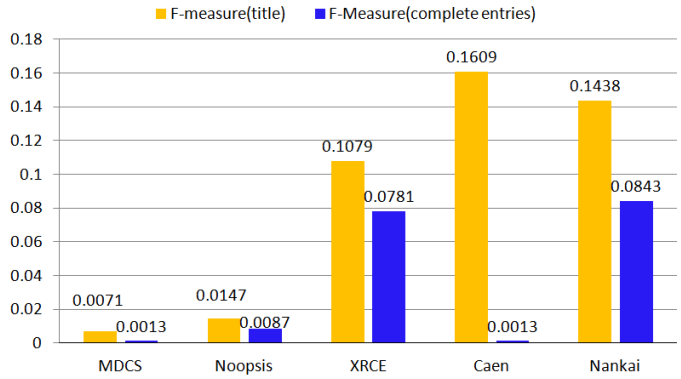


Fig. 5. Experiment on books without TOC pages and it is conducted on 93 books of ICDAR 2009 dataset

The F-Measure of complete entries on the 2011 ICDAR dataset is 5.20%. Owing to no result of books without table of contents is publicly published, we also conduct our experiment with the 2009 ICDAR dataset to make a comparison. The result of five fold-cross validation on 2009 dataset is shown in Table 3. All of these five evaluate aspects are considered to give a full description of our method. Figure 5 shows the comparison with other methods public published in ICDAR 2009. It can be seen that our method outperforms others'.

8 Conclusion

This paper presents the task of extracting TOC entries for navigation purpose. Due to the missing of layout and structural information caused by the OCR process, how to extract TOC entries from OCR-ed books becomes a challenging problem. We proposed an effective method to solve this problem. For books containing contents page, a rule based method is conducted. For books without contents page, we utilize a machine learning method to classify the title. Besides, recognition of the title, the matching of physical and logical page is conducted to help users navigate. To get more specific information about the book, the partition of the level of title is also performed. The experiments show that our method considering these three aspects is usable and effective. A uniform model to effectively address the problem is expected as a future work.

Acknowledgments. This research was supported supported by the Fundamental Research Funds for the Central Universities and the National Natural Science Foundation of China under Grant No. 61105049.

References

1. Dresevic, B., Uzelac, A., Radakovic, B., Todic, N.: Book Layout Analysis: TOC Structure Extraction Engine. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2008. LNCS, vol. 5631, pp. 164–171. Springer, Heidelberg (2009)
2. Doucet, A., Kazai, G., Dresevic, B., Uzelac, A., Radakovic, B., Todic, N.: Setting up a Competition Framework for the Evaluation of Structure Extraction from OCR-ed Books. International Journal of Document Analysis and Recognition (IJDAR) 14(1), 45–52 (2010)
3. Giguet, E., Lucas, N.: The Book Structure Extraction Competition with the Resurgence Software at Caen University. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 170–178. Springer, Heidelberg (2010)
4. Déjean, H., Meunier, J.-L.: XRCE Participation to the 2009 Book Structure Task. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 160–169. Springer, Heidelberg (2010)

OUC's Participation in the 2011 INEX Book Track

Michael Preminger and Ragnar Nordlie

Oslo and Akershus University College of Applied Science

Abstract. In this article we describe the Oslo University College's participation in the INEX 2011 Book track. In 2010, the OUC submitted retrieval results for the "Prove It" task with traditional relevance detection combined with some rudimental detection of confirmation. In line with our belief that proving or refuting facts are different semantic aware actions of speech, we have this year attempted to incorporate some rudimentary semantic support based on the WordNet database.

1 Introduction

In recent years large organizations like national libraries, as well as multinational organizations like Microsoft and Google have been investing labor, time and money in digitizing books. Beyond the preservation aspects of such digitization endeavors, they call on finding ways to exploit the newly available materials, and an important aspect of exploitation is book and passage retrieval.

The INEX Book Track[1], which has been running since 2007, is an effort aiming to develop methods for retrieval in digitized books. One important aspect here is to test the limits of traditional methods of retrieval, designed for retrieval within "documents" (such as news-wire), when applied to digitized books. One wishes to compare these methods to book-specific retrieval methods.

One important mission of such retrieval is supporting the generation of new knowledge based on existing knowledge. The generation of new knowledge is closely related to access to – as well as faith in – existing knowledge. One important component of the latter is claims about facts. This year's "Prove It" task may be seen as challenging the most fundamental aspect of generating new knowledge, namely the establishment (or refutal) of factual claims encountered during research.

On the surface, this may be seen as simple retrieval, but proving a fact is more than finding relevant documents. This type of retrieval requires from a passage to "make a statement about" rather than "be relevant to" a claim, which traditional retrieval is about. The questions we posed in 2010 were:

- what is the difference between simply being relevant to a claim and expressing support for a claim
- how do we modify traditional retrieval to reveal support or refutal of a claim?

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 109–117, 2012.

We also made the claim that "Prove It" sorts within the (not very well-defined) category "semantic-aware retrieval", which, for the time being will be defined by us as retrieval that goes beyond simple string matching, and is aware of the meaning (semantics) of text.

Those question, being rhetorical in part, may be augmented by the questions

- How can one detect the meaning of texts (words, sentences and passages) and incorporate those in the retrieval process to attain semantic-aware retrieval

and consequently

- can one exploit technologies developed within the semantic web to improve semantic-aware retrieval

The latter is not directly addressed in this paper, but we claim that the techniques used here point in this direction.

2 The "Prove It" Task

2.1 Task Definition and User Scenario

The prove-it task is still at its infancy, and may be subject to some modifications in the future. Quoting the user scenario as formulated by the organizers

> The scenario underlying this task is that of a user searching for specific information in a library of books that can provide evidence to confirm or refute a given factual statement. Users expect to be pointed directly at book pages that can help them to confirm or refute the claim of the topic. Users are assumed to view the ranked list of retrieved book pages starting from the top of the list and moving down, examining each result. No browsing is considered (only the returned book pages are viewed by users).

This user scenario is a natural point of departure as it is in the tradition of information retrieval and facilitates the development of the task by using existing knowledge. As a future strategy, it may be argued that this user scenario is gradually modified, as ranking in the context of proving is a highly complex process, and, in the context where Prove-it algorithms are most likely to be used, arguably superfluous.

2.2 What Is a Proof?

What constitutes a proof is well defined in fields like mathematics and computer science. In connection with a claim or a statement of fact, it is less obvious what demands a passage of text should satisfy in order to be considered proof of the claim. Obviously, we are looking for a passage which expresses a relevant truth about the claim, but what are the characteristics which signal a sufficient degree

of relevance and truthfulness? We might want to identify a trustworthy passage, which in its turn might be identified by considering the source of the passage, the degree to which the passage agreed with other passages treating the same claim or fact, or the centrality of the claim to the main content of the text. We might want to identify a concentrated passage, a passage where the largest amount of elements contained in the claim were represented or where they were by some measure most heavily represented. We might look for a definitional passage, which typographically or linguistically showed the characteristics of a definition. Or we might try to identify a "proof" by linguistic characteristics, mostly semantic, which might be of different kinds: certain typical words might be relatively consistently used to speak about a fact or claim in a "proving" manner, writing in a "proving" mode might entail using terms on a certain level of specificity, etc. These latter aspects are orthogonal to the statement or claim itself in the sense that they (at least ideally) apply equally to whatever claim being the subject of proving / confirming.

2.3 Semantic Approaches to Proof

A statement considered as a "proof" (or confirmation) may be characterized semantically by several indicators:

- the phenomenon to be supported may be introduced or denoted by specific terms, for instance verbs indicating a definition: "is", "constitutes", "comprises" etc.
- terms describing the phenomenon may belong to a specific semantic category
- nouns describing the phenomenon may be on a certain level of specificity
- verbs describing the phenomenon may denote a certain type of action or state

Deciding which specificity level or which semantic categories will depend on the semantic content and the relationship between the terms of the original claim. Without recourse to the necessary semantic analysis, we assume that in general, terms indicating a proof / confirmation will be on a relatively high level of specificity. It will in some way constitute a treatment of one or more aspects of the claim at a certain level of detail, which we expect to be reflected in the terminology which is applied.

As an initial exploration of these potential indicators of proof, without access to semantic analysis of the claim statements, we are investigating whether terms, in our case nouns, found on a page indicated as a potential source of proof diverges in a significant way from other text in terms of level of specificity. We determine the level of noun specificity through their place in the WordNet([2]) term hierarchies.

As stated further down the paper, this is an initial use of this type of semantics in retrieval, and the only thing we can hope for is that it gives us an indication about whether proceeding in this path is viable.

2.4 Ranking According to "Proof Efficiency"?

In this paper we are still following the two-step strategy of first finding pages relevant to the claim, and from those pages trying to identify pages that are likely to prove the claim[1]. The first step is naturally done using current strategies for ranked retrieval. The second stage identifies *among relevant documents* those which prove / confirm the statement. Rank order is not necessarily preserved in this process: if document A comprises a better string-wise match with the claim than does document B, document B can still be more efficient at proving the claim than document A is. Not all elements that make a document relevant also make it a good prover

Another issue is the context in which prove-it is used. One example is the writing of a paper. A writer is (again, arguably) more likely to evaluate a greater number of sources for proof of a claim than he or she would in a context of pure fact finding. Additionally, different contexts would arguably invite different proof emphases. All this advocates for use of other strategies of presenting proving results than ranked lists.

3 Indexing and Retrieval Strategies

The point of departure of the strategies discussed here is that confirming or refuting a statement is a simple action of speech that does not require from the book (the context of the retrieved page) to be *about* the topic covering the fact. In this way the "Prove It" task is different than e.g. the one referred to in [3] This means that we do not need the index we build for search purposes to be context-faithful (pages need not be indexed in a relevant book context). It is the formulation of the statement in the book or page that matters.

3.1 Indexing

In line with the above, indexing should facilitate two main aspects at retrieval time: identifying relevant pages and finding which of these is likely to prove a claim. The first aspect is catered for creating a simple index of all the words in the corpus, page by page. The pages are treated as separate documents regardless of the book in which they appear. The second aspect is catered for by calculating the average specificity of each page and tagging each page by one of a number of specificity tags. The latter are determined as described in Section 3.2

3.2 Calculating Specificity

At this stage of the research, the aspect of finding pages likely to prove a claim is catered for by statistically measuring the average specificity of words that occur in the page. We do this by calculating the specificity of each word and then averaging the measure of specificity of all the words in a page, as described

[1] We see refutal as a totally different type of task and will not address it in this paper.

below. To accomplish that, we have augmented the WordNet database (ref) by a Direct Acyclic Graph (DAG) of all the nouns, which lets us calculate a relative specificity of each word by its average position in this graph. Words closer to the common root of the graph (measured as a number of steps) are less specific, whereas words closer to the leaves are more specific. For each word in a trajectory, the specificity S is calculated as

$$S = \frac{P}{L},$$

where P is the position of the words in the trajectory (number of steps away from the root) and L is the length of the trajectory from root to leaf. Since this is a graph and not a tree, each word (a string of characters), even a leaf, may belong to more than one trajectory depending on the number of senses / synsets it participates in, and the number of parallel synsets it is a descendent of. Since we generally cannot know which sense of a word a certain occurrence stands for, we assign to each word (string of characters) the average of its specificities. Each page is then assigned the average of the specificities of its constituent words. Words not in the graph are assigned the "neutral" value of 0.5.

The pages are then categorized into predefined intervals of average specificity. We were working with an interval resolution of 5%, where pages between x and $x + 5\%$ are categorized together for each $x = 5\%, 10\%, 15\%...$. Each interval has its own tag for indexing purposes. These tags then facilitate weighting pages differently at retrieval time when retrieving candidates of confirming pages.

4 Runs and Results

We look at results in two different sub-scenarios. Instant - to what extent the system supports "instant proving" of documents. In this sub-scenario the first document that proves the statement is taken as the statement's proof, and no further pages are visited. This mode is well represented by the MRR (Mean reciprocal rank) measure. Thorough - more pages are visited to establish the proof of the statement. This is well represented by the MAP measure, and precision-recall curves. The NDCG (official measure of the Track) expresses both sub scenarios.

The way we measure the effect of specificity is that we, at retrieval time, boost up pages with different rates of specificity (as measured and tagged in 3.2) weighting them up by different factors. We operate with two range-modes:

- A narrow specificity interval (5 percent points between $x\%$ and $x + 5\%$) (eq) "spec_2x_eq_55" means that pages with a specificity between 55 and 60 are weighted twice as much as other pages.
- A one-sided specificity interval greater than or equal to an interval point (ge). "spec_5x_ge_55" means that the pages with a specificity equal to or greater than 55 are given five time the weight of lower-specificity pages at retrieval time.

In this section we present two types of runs:

- Calibration runs, runs that are meant to find good parameter candidates for specificity and document weighting at retrieval time.
- Full scale performance runs

4.1 Calibration Runs

The calibration runs are runs performed against an index of this subset of the pages only containing the pages appearing in the recall base of at least one of the topics (the pages in the applicable *.qrel file). The performance runs are runs against the entire page corpus.

The purpose of the calibration runs is to more sensitively (and more effortlessly) measure the effect of the parameters and combination of them on several performance indicators, before applying the best performing parameters to the full-scale performance runs. An index is constructed, containing only the pages appearing in the "qrel" files, giving the algorithms fewer non-relevant pages to deal with. A number of pre-runs not reported here have indicated an effective range of specificity (just above the neutral 0.5 rate) that perform better than both lower and higher measures.

In figure 1 we can see that the intermediate two-sided ranges, 55% and 60%, generally perform better than the 50% 65% ranges. Narrow (two-sided) ranges perform better than one-sided. Based on these results, a specificity rate of 60% gives the best reciprocal mean rank measure, meaning that the document performing best is in average second or third in the ranking list. A slightly lower specificity rate (55%) seems to better support the sub scenario where the user looks at a number of pages before accepting a statement as confirmed (as expressed by the map and ndcg measures).

4.2 Performance Runs

In figure 2 we present full-scale runs made against the full-scale index (17M pages), using the best parameters of the calibration runs.

The results presented here are an attempt at relating this year's results to our 2010 results [4]. Figure 3 shows the results of weighting pages featuring 3 percents or more confirmatory words at retrieval time, weighted double, quintuple (5x) and decuple (10x) the baseline[2]. We do spot a slight improvement in the 2011 results, but it is hard to say whether it is significant.

5 Discussion, Limitation and Further Research

At the same time that the book world becomes more and more digital,as old books are being digitized and new books are increasingly published digitally,

[2] For these, as well as all other plots, We were using the indri combine / weight operation (a combination of weighted expression) with no changes to the default setting (regarding smoothing, a.s.o).

50 ndcg			50 ndcg@10			50 map			50 rr		
	spec_10x_ge_50	0,358		spec_2x_ge_50	0,0797		spec_10x_ge_50	0,1075		spec_5x_ge_50	0,3551
	spec_5x_ge_50	0,3573		spec_5x_ge_50	0,0779		spec_5x_ge_50	0,1067		spec_10x_ge_50	0,353
	spec_2x_ge_50	0,3543		spec_10x_ge_50	0,0746		spec_2x_ge_50	0,1024		spec_2x_ge_50	0,3475
	spec_2x_eq_50	0,3395		spec_2x_eq_50	0,0716		spec_2x_eq_50	0,0886		spec_2x_eq_50	0,2579
	spec_5x_eq_50	0,3279		spec_5x_eq_50	0,0579		spec_5x_eq_50	0,0806		spec_5x_eq_50	0,2046
	spec_10x_eq_50	0,3232		spec_10x_eq_50	0,0568		spec_10x_eq_50	0,0774		spec_10x_eq_50	0,1969
55 ndcg			55 ndcg@10			55 map			55 rr		
	spec_10x_eq_55	0,3882		spec_2x_eq_55	0,1346		spec_10x_eq_55	0,1482		spec_2x_ge_55	0,4123
	spec_5x_eq_55	0,3853		spec_5x_eq_55	0,1216		spec_5x_eq_55	0,1445		spec_5x_ge_55	0,4063
	spec_2x_eq_55	0,3733		spec_10x_eq_55	0,1174		spec_10x_ge_55	0,1243		spec_10x_ge_55	0,4003
	spec_10x_ge_55	0,3711		spec_10x_ge_55	0,0946		spec_2x_eq_55	0,1235		spec_2x_eq_55	0,3656
	spec_5x_ge_55	0,3693		spec_2x_ge_55	0,0938		spec_5x_ge_55	0,1217		spec_5x_eq_55	0,327
	spec_2x_ge_55	0,3621		spec_5x_ge_55	0,0911		spec_2x_ge_55	0,1119		spec_10x_eq_55	0,3105
60 ndcg			60 ndcg@10			60 map			60 rr		
	spec_2x_eq_60	0,3552		spec_10x_eq_60	0,1112		spec_5x_eq_60	0,1023		spec_2x_eq_60	0,4444
	spec_5x_eq_60	0,3552		spec_5x_eq_60	0,1101		spec_2x_eq_60	0,1019		spec_5x_eq_60	0,4391
	spec_10x_eq_60	0,3535		spec_2x_eq_60	0,1042		spec_10x_eq_60	0,1014		spec_10x_eq_60	0,4376
	spec_2x_ge_60	0,3484		spec_2x_ge_60	0,0893		spec_2x_ge_60	0,0959		spec_2x_ge_60	0,4011
	spec_5x_ge_60	0,3448		spec_10x_ge_60	0,0874		spec_5x_ge_60	0,0936		spec_10x_ge_60	0,3906
	spec_10x_ge_60	0,3423		spec_5x_ge_60	0,0846		spec_10x_ge_60	0,0921		spec_5x_ge_60	0,3893
65 ndcg			65 ndcg@10			65 map			65 rr		
	spec_2x_eq_65	0,3409		spec_2x_eq_65	0,0719		spec_2x_eq_65	0,0918		spec_2x_ge_65	0,2476
	spec_2x_ge_65	0,3375		spec_2x_ge_65	0,0719		spec_5x_eq_65	0,0888		spec_10x_ge_65	0,2454
	spec_5x_eq_65	0,3359		spec_10x_ge_65	0,0627		spec_2x_ge_65	0,0883		spec_2x_eq_65	0,2415
	spec_10x_eq_65	0,3331		spec_5x_eq_65	0,0531		spec_10x_eq_65	0,0862		spec_10x_eq_65	0,1775
	spec_5x_ge_65	0,3273		spec_10x_eq_65	0,0521		spec_5x_ge_65	0,0818		spec_5x_eq_65	0,1747
	spec_10x_ge_65	0,2527		spec_5x_ge_65	0,0516		spec_10x_ge_65	0,0656		spec_5x_ge_65	0,1607

Fig. 1. Calibration runs: NDCG, MAP and Mean reciprocal rank results for runs using different parameter values

	ncdg	ndcg@10	map	rr
spec_10x_eq_55	0,1708	0,1349	0,0398	0,3266
spec_10x_eq_60	0,0802	0,1097	0,0195	0,4102
spec_2x_eq_55	0,1384	0,1372	0,0278	0,3521
spec_2x_eq_60	0,0877	0,1305	0,0196	0,3934
spec_5x_eq_55	0,1655	0,1408	0,0371	0,3184
spec_5x_eq_60	0,0794	0,1121	0,0192	0,4047

Fig. 2. Performance runs: NDCG, MAP and Mean reciprocal rank results for runs against a full scale index

	ncdg	map	rr
to_g_10xover3.eval	0,1177	0,0251	0,3039
to_g_2xover3.eval:	0,1267	0,0263	0,3053
to_g_5xover3.eval:	0,1149	0,0241	0,2923

Fig. 3. The 2010 results: NDCG, MAP and Mean reciprocal rank results for runs against a full scale index

information not published in book format becomes more and more "semantic" in the sense that data pieces (as opposed to exclusively documents in the web's first years) are linked together and made available. These two parallel development entail great opportunities in the exploitation of book material for different purposes, of which the topic of this paper is one example.

This paper provides an example of the possibilities and the challenges. Whereas "WordNet specificity", here representing content independent linguistic semantic, is one simple example of information that can be used to systematically extract semantics from written content, other much larger and much more complicated sources of semantics, the semantic web and linked data, are waiting to be used in a similar (or related) way. To explore these possibilities we will need to experiment with more modern texts than what our present test collection contains.

To judge by the results of the runs presented here, this path of research, though promising, still requires a lot of modification and calibration.

Exploring the semantics of a page in a basically statistical manner may be seen as a superposition of independent components. Counting occurrences of special words is one component on which we superimpose the detection of noun specificity. The treatment using WordNet represents further progress from the 2010 experiments, but is still rudimentary. Nouns are currently the only word-class we are treating, using only level of specificity. trying to detect classes nouns using the lateral structure of synsets may be another path to follow. It is also conceivable that treating of other word classes, primarily verbs, might contribute to the treatment. Verbs are more complicated than nouns in WordNet and such treatment will be more demanding.

Utilizing digital books poses new challenges on information retrieval. The mere size of the book text poses both storage, performance and content related challenges as compared to texts of more moderate size. But the challenges are even greater if books are to be exploited not only for finding facts, but also to support exploitation of knowledge, identifying and analyzing ideas, a.s.o.

This article represents work in progress. We explore techniques gradually in an increasing degree of complexity, trying to adapt and calibrate them.

Even though such activities may be developed and refined using techniques from e.g. Question Answering[5], we suspect that employing semantics-aware retrieval [6,7], which is closely connected to the development of the Semantic Web [8] would be a more viable (and powerful) path to follow.

One obstacle particular to this research is the test collection. Modern ontologies code facts that are closely connected to the modern world. For example the Yago2 [9] ontology, that codes general facts automatically extracted from Wikipedia, may be complicated to apply to an out-of-copyright book collection emerging from academic specialized environments. But this is certainly a path to follow.

6 Conclusion

This article is a further step in a discussion about semantics-aware retrieval in the context of the INEX book track. Proving (or confirmation or support) of factual statements is discussed in light of some rudimental retrieval experiments incorporating semantics. We also discuss the task of proving statement, raising the question whether it is classifiable as a semantics-aware retrieval task. Results are highly inconclusive.

References

1. Kazai, G., Koolen, M., Kamps, J., Doucet, A., Landoni, M.: Overview of the INEX 2010 Book Track: Scaling Up the Evaluation Using Crowdsourcing. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 98–117. Springer, Heidelberg (2011)
2. Fellbaum, C.: WordNet: an electronic lexical database. MIT Press, Cambridge (1998)
3. Cartright, M.A., Feild, H., Allan, J.: Evidence finding using a collection of books. In: BooksOnline 2011 Proceedings of the 4th ACM Workshop on Online Books, Complementary Social Media and Crowdsourcing, Amherst, MA, pp. 11–18 (2011)
4. Preminger, M., Nordlie, R.: OUC's Participation in the 2010 INEX Book Track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 164–170. Springer, Heidelberg (2011)
5. Voorhees, E.M.: The trec question answering track. Natural Language Engineering 7, 361–378 (2001)
6. Finin, T., Mayfield, J., Joshi, A., Cost, R.S., Fink, C.: Information retrieval and the semantic web. In: Proc. 38th Int. Conf. on System Sciences, Digital Documents Track (The Semantic Web: The Goal of Web Intelligence) (2005)
7. Mayfield, J., Finin, T.: Information retrieval on the semantic web: Integrating inference and retrieval. In: SIGIR Workshop on the Semantic Web, Toronto (2003)
8. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001)
9. Hoffart, J., Suchanek, F., Berberich, K., Weikum, G.: Yago2: A spatially and temporally enhanced knowledge base from wikipedia. Special Issue of the Artificial Intelligence Journal (2012)

Overview of the INEX 2011 Data-Centric Track

Qiuyue Wang[1,2], Georgina Ramírez[3], Maarten Marx[4],
Martin Theobald[5], and Jaap Kamps[6]

[1] School of Information, Renmin University of China, P.R. China
[2] Key Lab. of Data Engineering and Knowledge Engineering, MOE, P.R. China
qiuyuew@ruc.edu.cn
[3] Universitat Pompeu Fabra, Spain
georgina.ramirez@upf.edu
[4] University of Amsterdam, The Netherlands
maartenmarx@uva.nl
[5] Max-Planck-Institut für Informatik, Germany
martin.theobald@mpi-inf.mpg.de
[6] University of Amsterdam, The Netherlands
kamps@uva.nl

Abstract. This paper presents an overview of the INEX 2011 Data-Centric Track. Having the *ad hoc search task* running its second year, we introduced a new task, *faceted search task*, whose goal is to provide the infrastructure to investigate and evaluate different techniques and strategies of recommending facet-values to aid the user to navigate through a large set of query results and quickly identify the results of interest. The same IMDB collection as last year was used for both tasks. A total of 9 active participants contributed a total of 60 topics for both tasks and submitted 35 ad hoc search runs and 13 faceted search runs. A total of 38 ad hoc search topics were assessed, which included 18 subtopics for 13 faceted search topics. We discuss the setup for both tasks and the results obtained by their participants.

1 Introduction

As the de facto standard for data exchange on the web, XML is widely used in all kinds of applications. XML data used in different applications can be categorized into two broad classes: one is document-centric XML, where the structure is simple and long text fields predominate, e.g. electronic articles, books and so on, and the other is data-centric XML, where the structure is very rich and carries important information about objects and their relationships, e.g. e-Commerce data or data published from databases. The INEX 2011 Data Centric Track is investigating retrieval techniques and related issues over a strongly structured collection of XML documents, the IMDB data collection. With richly structured XML data, we may ask how well such structural information could be utilized to improve the effectiveness of search systems.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 118–137, 2012.

The INEX 2011 Data-Centric Track features two tasks: the *ad hoc search task* and the *faceted search task*. The *ad hoc search task* consists of informational requests to be answered by the entities contained in the IMDB collection (movies, actors, directors, etc.); the *faceted search task* asks for a restricted list of facet-values that will optimally guide the searcher towards relevant information in a ranked list of results, which is especially useful when searchers' information needs are vague or complex.

There were 49 institutes or groups interested in participating in the track, from which 8 (Kasetsart University, Benemérita Universidad Autónoma de Puebla, University of Amsterdam, IRIT, University of Konstanz, Chemnitz University of Technology, Max-Planck Institute for Informatics, Universitat Pompeu Fabra) submitted 45 valid ad hoc search topics and 15 faceted search topics. A total of 9 participants (Kasetsart University, Benemérita Universidad Autónoma de Puebla, University of Amsterdam, IRIT, Chemnitz University of Technology, Max-Planck Institute for Informatics, Universitat Pompeu Fabra, Renmin University of China, Peking University) submitted 35 ad hoc search runs and 13 faceted search runs. A total of 38 ad hoc topics were assessed, which included 18 subtopics for 13 faceted search topics.

2 Data Collection

The track uses the cleaned IMDB data collection used in INEX 2010 Data-Centric Track [1]. It was generated from the plain text files published on the IMDB web site on April 10, 2010. There are two kinds of objects in the collection, movies and persons involved in movies, e.g. actors/actresses, directors, producers and so on. Each object is richly structured. For example, each movie has title, rating, directors, actors, plot, keywords, genres, release dates, trivia, etc.; and each person has name, birth date, biography, filmography, etc. Each XML document contains information about one object, i.e. a movie or person, with structures conforming to the movie.dtd or person.dtd [1]. In total, the IMDB data collection contains 4,418,081 XML documents, including 1,594,513 movies, 1,872,471 actors, 129,137 directors who did not act in any movie, 178,117 producers who did not direct nor act in any movie, and 643,843 other people involved in movies who did not produce nor direct nor act in any movie.

3 Ad-Hoc Search Task

The task is to return a ranked list of results, i.e. objects, or equivalently documents in the IMDB collection, estimated relevant to the user's information need.

3.1 Topics

Each participating group was asked to create a set of candidate topics, representative of a range of real user needs. Each group had to submit a total of 3 topics, one for each of the categories below:

- **Known-Item:** Topics that ask for a particular object (movie or person). Example: "I am searching for the version of the movie 'Titanic' in which the two major characters are called Jack and Rose respectively". For these topics the relevant answer is a single (or a few) document(s). We will ask participants to submit the file name(s) of the relevant document(s).
- **List:** Topics that ask for a list of objects (movies or persons). For example: "Find movies about drugs that are based on a true story", "Find movies about the era of ancient Rome".
- **Informational:** Topics that ask for information about any topic/movie/person contained in the collection. For example: "Find information about the making of The Lord of the Rings and similar movies", "I want to know more about Ingmar Bergman and the movies she played in".

All the data fields in the IMDB collection can be categorized into three types: categorical (e.g. genre, keyword, director), numerical (e.g. rating, release_date, year), and free-text (e.g. title, plot, trivia, quote). All submitted topics had to involve, at least, one free-text field. The list of all the fields along with their types is given in Appendix 1. We asked participants to submit challenging topics, i.e. topics that could not be easily solved by a current search engine or DB system. Both Content Only (CO) and Content And Structure (CAS) variants of the information need were requested. TopX provided by Martin Theobald was used to facilitate topic development.

After cleaning some duplicates and incorrectly-formed topics, there were a total of 25 valid topics (11 list, 7 known-item, 7 informational). An example of topic is shown in Fig. 1.

```
<topic id="2011105" guid="20">
  <task>AdHoc</task>
  <type>Known-Item</type>
  <title>king kong jack black</title>
  <castitle>//movie[about(.//title, king kong) and about(.//actor, jack black)]</castitle>
  <description>I am searching for the version of the movie "King Kong" with the actor
Jack Black.</description>
  <narrative>Cause i've heard that this is the best King Kong movie, I am searching for the
version of the movie "King Kong", with the actor Jack Black.</narrative>
</topic>
```

Fig. 1. INEX 2011 Data Centric Track Ad Hoc Search Topic 2011105

3.2 Submission Format

Each participant could submit up to 3 runs. Each run could contain a maximum of 1000 results per topic, ordered by decreasing value of relevance. The results of one run had to be contained in one submission file (i.e. up to 3 files could be submitted in total). For relevance assessment and evaluation of the results we required submission files to be in the familiar TREC format:

<qid> Q0 <file> <rank> <rsv> <run_id>

Here:

- The first column is the topic number.
- The second column is the query number within that topic. This is currently unused and should always be Q0.
- The third column is the file name (without .xml) from which a result is retrieved.
- The fourth column is the rank of the result.
- The fifth column shows the score (integer or floating point) that generated the ranking. This score MUST be in descending (non-increasing) order and is important to include so that we can handle tied scores (for a given run) in a uniform fashion (the evaluation routines rank documents from these scores, not from ranks).
- The sixth column is called the "run tag" and should be a unique identifier that identifies the group and the method that produced the run. The run tags must contain 12 or fewer letters and numbers, with NO punctuation, to facilitate labeling graphs with the tags.

An example submission is:

2011001 Q0 9996 1 0.9999 2011UniXRun1
2011001 Q0 9997 2 0.9998 2011UniXRun1
2011001 Q0 person_9989 3 0.9997 2011UniXRun1

Here are three results for topic “2011001”. The first result is the movie from the file 9996.xml. The second result is the movie from the file 9997.xml, and the third result is the person from the file person_9989.xml.

4 Faceted Search Task

Given a vague or broad query, the search system may return a large number of results. Faceted search is a way to help users navigate through the large set of query results to quickly identify the results of interest. It presents the user a list of facet-values to choose from along with the ranked list of results. By choosing from the suggested facet-values, the user can refine the query and thus narrow down the list of candidate results. Then, the system may present a new list of facet-values for the user to further refine the query. The interactive process continues until the user finds the items of interest. The key issue in faceted search is to recommend appropriate facet-values for the user to refine the query and thus quickly identify what he/she really wants in the large set of results. The task aims to investigate and evaluate different techniques and strategies of recommending facet-values to the user at each step in a search session.

4.1 Topics

Each participating group was asked to create a set of candidate topics representative of real user needs. Each topic consists of a general topic as well as a subtopic that

refines the general topic by specifying a particular interest of it. The general topic had to result in more than 1000 results, while the subtopics had to be restrictive enough to be satisfied by 10 to 50 results.

Each group had to submit 4 topics: two from the set of general topics given by the organizers, and two proposed by the participants themselves. The given set of general topics was: {"trained animals", "dogme", "food", "asian cinema", "art house", "silent movies", "second world war", "animation", "nouvelle vague", "wuxia"}.

After removing incorrectly-formed topics, we got a total of 15 general topics along with their 20 subtopics (2 subtopics for "Food", 3 subtopics for "Cannes" and 3 subtopics for "Vietnam"). An example of topic is shown in Fig. 2. The general topic is specified in the <general> field of the <topic> element, while the other fields of <topic>, e.g. <title> and <castitle>, are used to specify the subtopic, which is the searcher's real intention when submitting this general topic to the search system. The participants running the faceted search task could only view the 15 general topics, while the corresponding 20 subtopics were added to the set of topics for the ad hoc search task. The relevance results for these subtopics were used as the relevance results for their corresponding general topics. Thus, altogether we got 45 topics for the ad hoc search task and 15 topics for the faceted search task.

<topic id="**2011202**" guid="**28**">
<task>**Faceted**</task>
<general>**animation**</general>
<title>**animation fairy-tale**</title>
<castitle>**//movie[about(.//genre, animation) and about(.//plot, fairy-tale)]**</castitle>
<description>**I am searching for all animation movies based on a fairy-tale.**</description>
<narrative>**I like fairy-tales and their animations remakes.**</narrative>
</topic>

Fig. 2. INEX 2011 Data Centric Track Faceted Search Topic 2011202

4.2 Submission Format

Each participant had to submit up to 3 runs. A run consists of two files: one is the result file containing a ranked list of maximum 2000 results per topic in the ad hoc search task format, and the other is the recommended facet-value file, which can be a static facet-value file or a dynamic faceted search module.

(1) Facet-value File. It contains a hierarchy of recommended facet-values for each topic, in which each node is a facet-value and all of its children constitute the newly recommended facet-value list as the searcher selects this facet-value to refine the query. The maximum number of children for each node is restricted to be 20. The submission format is in an XML format conforming to the following DTD.

```
<!ELEMENT run    (topic+)>
<!ATTLIST run    rid    ID      #REQUIRED>
<!ELEMENT topic  (fv+)>
<!ATTLIST topic  tid    ID      #REQUIRED>
<!ELEMENT fv     (fv*)>
<!ATTLIST fv     f      CDATA   #REQUIRED
                 v      CDATA   #REQUIRED>
```

Here:

- The root element is <run>, which has an ID type attribute, *rid*, representing the unique identifier of the run. It must be identical with that in the result file of the same run.
- The <run> contains one or more <topic>s. The ID type attribute, *tid*, in each <topic> gives the topic number.
- Each <topic> has a hierarchy of <fv>s. Each <fv> shows a facet-value pair, with *f* attribute being the facet and *v* attribute being the value. The facet is expressed as an XPath expression. The set of all the possible facets represented as XPath expressions in the IMDB data collection can be found in Appendix 1. We allow only categorical or numerical fields to be possible facets. Free-text fields are not considered. Each facet-value pair represents a facet-value condition to refine the query. For example, <fv f="/movie/overview/directors/director" v="Yimou Zhang"> represents the condition /movie/overview/directors/director="Yimou Zhang".
- The <fv>s can be nested to form a hierarchy of facet-values.

An example submission is:

```
<run rid="2011UniXRun1">
  <topic tid="2011001">
    <fv f="/movie/overview/directors/director" v="Yimou Zhang">
      <fv f="/movie/cast/actors/actor/name" v="Li Gong">
        <fv f="/movie/overview/releasedates/releasedate" v="2002"/>
        <fv f="/movie/overview/releasedates/releasedate" v="2003"/>
      </fv>
      <fv f="/movie/cast/actors/actor/name" v="Ziyi Zhang">
        <fv f="/movie/overview/releasedates/releasedate" v="2005"/>
      </fv>
    </fv>
    …
  </topic>
  <topic tid="2011002">
    ...
  </topic>
…
</run>
```

Here for the topic "2011001", the faceted search system first recommends the facet-value condition /movie/overview/directors/director="Yimou Zhang" among other facet-value

conditions, which are on the same level of the hierarchy. If the user selects this condition to refine the query, the system will recommend a new list of facet-value conditions, which are /movie/cast/actors/actor/name="Li Gong" and /movie/cast/actors/actor/name="Ziyi Zhang", for the user to choose from to further refine the query. If the user then selects /movie/cast/actors/actor/name="Li Gong", the system will recommend /movie/overview/releasedates/releasedate="2002" and /movie/overview/releasedates/releasedate ="2003". Note that the facet-value conditions that are selected to refine the query form a path in the tree, e.g. /movie/overview/directors/director="Yimou Zhang" → /movie/cast/actors/actor/name = "Li Gong" → /movie/overview/releasedates/releasedate ="2003". It is required that no facet-value condition occurs twice on any path.

(2) Faceted Search Module. Instead of submitting a static hierarchy of facet-values, participants are given the freedom to dynamically generate lists of recommended facet-values and even change the ranking order of the candidate result list at each step in the search session. This is achieved by submitting a self-implemented dynamically linkable module, called Faceted Search Module (FSM). It implements the *FacetedSearchInterface* defined as the following:

```
public interface FacetedSearchInterface {
   public String[] openQuery(String topicID, String[] resultList);
   public String[] selectFV(String facet, String value, String[] selectedFV);
   public String[] refineQuery(String facet, String value, String[] selectedFV);
   public String[] expandFacet(String facet, String[] selectedFV);
   public void closeQuery(String topicID);
}

public class FacetedSearch implements FacetedSearchInterface {
    // to be implemented by the participant
}
```

The User Simulation System (USS) used in evaluation will interact with the FSM to simulate a faceted search session. The USS starts to evaluate a run by instantiating a FacetedSearch object. For each topic to be evaluated, the USS first invokes openQuery() method to initialize the object with the topic id and initial result list for this topic. The result list is actually the list of retrieved file names (without .xml) in the third column of the result file. The method would return a list of recommended facet-values for the initial result list. A facet-value is encoded into a String in the format "<facet>::<value>", for example, "/movie/overview/directors/director::Yimou Zhang".

After opening a query, the USS then simulates a user's behavior in a faceted search system based on some user model as described in Section 5. When the simulated user selects a facet-value to refine the query, the selectFV() method would be called to return a new list of recommended facet-values; and the refineQuery() method would be called to return a list of candidate results in the initial result list that satisfy all the selected facet-value conditions. The inputs to both methods are the currently selected facet and value, as well as a list of previously selected facet-values. A facet-vaue pair is encoded into a String in the format shown above.

If the user could not find a relevant facet-value to refine the query in the recommended list, he/she could probably expand the facet-value list by choosing a facet among all possible facets, examine all its possible values and then select one to refine the query. In such a case, the USS invokes the expandFacet() method with the name of the facet to be expanded as well as a list of previously selected facet-values as input and the list of all possible values of this facet as output. Observe that in the specification of FacetedSearchInterface, we do not restrict facet-value comparisons to be of equality, but can be of any other possible semantics since the interpretation of facet-value conditions is capsulated into the implementation of FacetedSearchInterface. Thus, given the same facet, different systems may give different sets of all possible values depending on if they will cluster and how they will cluster some values.

When the search session of a query ends, the closeQuery() method is invoked. The FacetedSearch object will be used as a persistent object over the entire evaluation of a run. That is, different topics in the same run will be evaluated using the same FacetedSearch object. But different runs may have different implementations of the FacetedSearch class.

5 Assessments and Evaluations

In total 35 ad hoc search runs and 13 faceted search runs were submitted by 9 active participants. Assessment was done using the same assessment tool as that used in INEX 2010 Data-Centric Track provided by Shlomo Geva. A total of 38 ad hoc topics among 45 ones were assessed by those groups that submitted runs. Among the assessed topics, there are 9 list type topics, 6 known-item type topics, 5 informational type topics, and 18 subtopics for 13 faceted search topics.

Table 1 shows the mapping between the subtopics in the ad hoc search task and the general topics in the faceted search task. The relevance results of subtopics are treated as the intended results for their corresponding general topics. Note that some general topics, e.g. 2011205, 2011207 and 2011210, have more than one intention/subtopic. For these general topics, we take the subtopics that have the least number of relevance results. For example, compared with topic 2011120 and 2011142, topic 2011141 has the least number of relevance results, whose relevance results are then chosen as the relevance results for topic 2011205. The chosen subtopics are underlined in Table 1. Since the subtopics 2011121 and 2011139 were not assessed, we have no relevance results for topics 2011206 and 2011215 in the faceted search task.

Table 1. Mapping between the faceted search topics and subtopics in ad hoc task

General Topics	Subtopics
2011201	2011111
2011202	2011114
2011203	2011118
2011204	2011119
2011205	2011120,2011141,2011142
2011206	2011121
2011207	2011112,2011140
2011208	2011129
2011209	2011130
2011210	2011135,2011144,2011145
2011211	2011143
2011212	2011136
2011213	2011137
2011214	2011138
2011215	2011139

The TREC MAP metric, as well as P@5, P@10, P@20 and so on, was used to measure the performance of all ad hoc runs at whole document retrieval.

For the faceted search task, since it is the first year, we used the following two types of evaluation approaches and metrics to gain better understanding to the problem.

- **NDCG of Facet-values:** The relevance of the hierarchy of recommended facet-values is evaluated based on the relevance of the data covered by these facet-values, measured by NDCG. The details of this evaluation methodology are given in [2].
- **Interaction Cost:** The effectiveness of a faceted search system is evaluated by measuring the interaction cost or the amount of effort spent by a user in meeting his/her information needs. To avoid an expensive user study and to make the evaluation repeatable, we applied user simulation methodology like that used in [3, 4] to measure the costs.

We can use two metrics to measure the user's interaction cost. One is the number of results, facets or facet-values that the user examined before he/she encounters the first relevant result, which is similar to the Reciprocal Rank metric in traditional IR. Here we assume that the effort spent on examining each facet or facet-value is the same as that spent on examining each result. The other is the number of actions that the user performs in the search session. We only consider the click actions.

As in [3, 4], we assume that the user will end the search session when he/she encounters the first relevant result, and the user can recognize the relevant results from the list of results, and can distinguish the relevant facets or facet-values that match at least one relevant result from the list of facets or facet-values.

The user begins by examining the first page of the result list for the current query. It is assumed that each page displays at most 10 results. If the user finds relevant results on the first page, the user selects the first one and ends the session. If no relevant result is found, the user then examines the list of recommended facet-values. If there are relevant facet-values, the user then clicks on the first relevant facet-value in the list to refine the query, and the system returns the new lists of results and facet-values for the refined query. If none of the recommended facet-values are relevant, the user chooses the first relevant facet in the list of all possible facets to expand and select the first relevant value in this facet's value list to refine the query. If the user does not find any relevant facet to expand, the user begins to scan through the result list and stops at the first relevant result encountered. Fig. 3 shows the flowchart of the user interaction model and cost model used in the evaluation. Notation used in Fig. 3 is given in Table 2.

Table 2. Notation used in Fig. 3

Symbol	Meaning
q	The current query
R_q	The result list of query q
FV_q	The list of recommended facet-values for query q
F_q	The list of all possible facets for query q
loc(x,y)	A function returns the position of item x in the list y
cost	The number of results, facet-values or facets examined by the user
actionCount	The number of click actions performed by the user

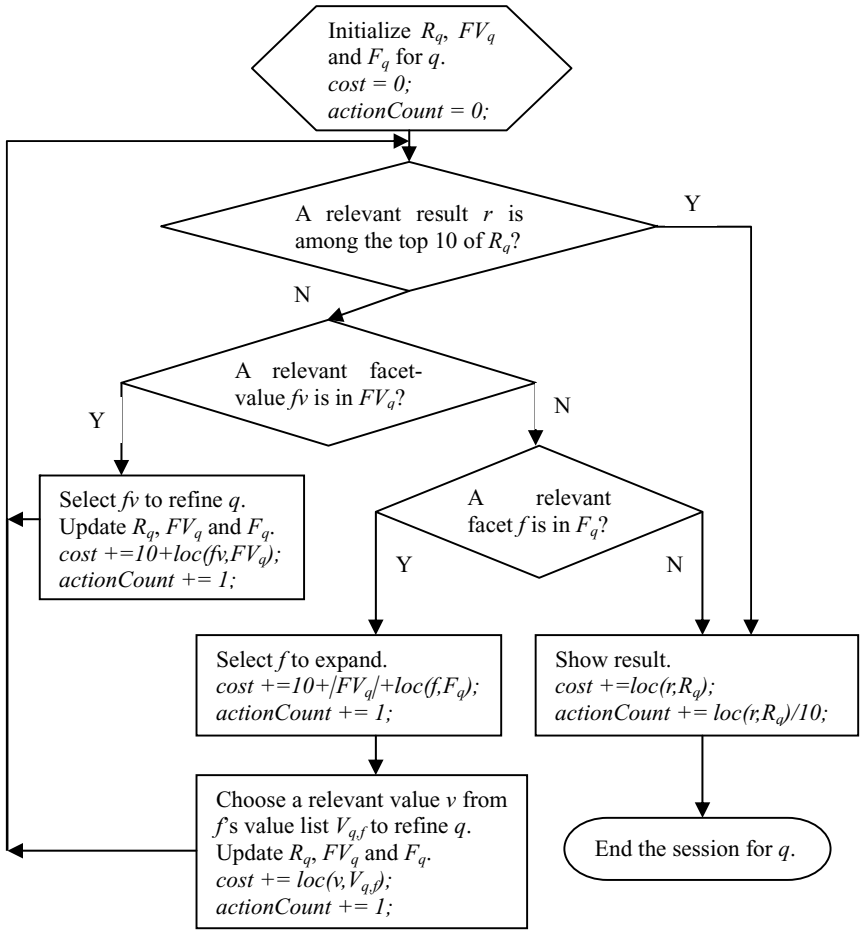


Fig. 3. Flowchart of the Simulated User Interaction Model with Faceted Search System

6 Results

6.1 Ad Hoc Search Results

As mentioned above, a total of 35 runs from 9 different institutes were submitted to the ad hoc search task. This section presents the evaluation results for these runs. Results were computed over the 38 topics assessed by the participants using the TREC evaluation tool. The topic set is a mixture of informational, known-item, list, and faceted (sub)topics. We use MAP as the main measure since it averages reasonably well over such a mix of topic types.

Table 3 shows an overview of the 10 best performing runs for this track. Over all topics, the best scoring run is from the University of Amsterdam with a MAP of 0.3969. Second best scoring team is Renmin University of China (0.3829). Third best scoring team is Kasetsart University (0.3479) with the highest score on mean reciprocal rank (1/rank). Fourth best team is Peking University (0.3113) with the highest precisions at 10 and 30. Fifth best team is Universitat Pompeu Fabra, with a MAP of 0.2696 but the highest score for precision at 20.

Table 3. Best performing runs (only showing one run per group) based on MAP over all ad hoc topics

Run	map	1/rank	P@10	P@20	P@30
p4-UAms2011adhoc	**0.3969**	0.6991	0.4263	0.3921	0.3579
p2-ruc11AS2	0.3829	0.6441	0.4132	0.3842	0.3684
p16-kas16-MEXIR-2-EXT-NSW	0.3479	**0.6999**	0.4316	0.3645	0.3298
p77-PKUSIGMA01CLOUD	0.3113	0.5801	**0.4421**	0.4066	**0.3851**
p18-UPFbaseCO2i015	0.2696	0.5723	0.4342	**0.4171**	0.3825
p30-2011CUTxRun2	0.2099	0.6104	0.3684	0.3211	0.2965
p48-MPII-TOPX-2.0-co	0.1964	0.5698	0.3684	0.3395	0.3289
p47-FCC-BUAP-R1	0.1479	0.5120	0.3474	0.2763	0.2412
p12-IRIT_focus_mergeddtd_04	0.0801	0.2317	0.2026	0.1724	0.1702

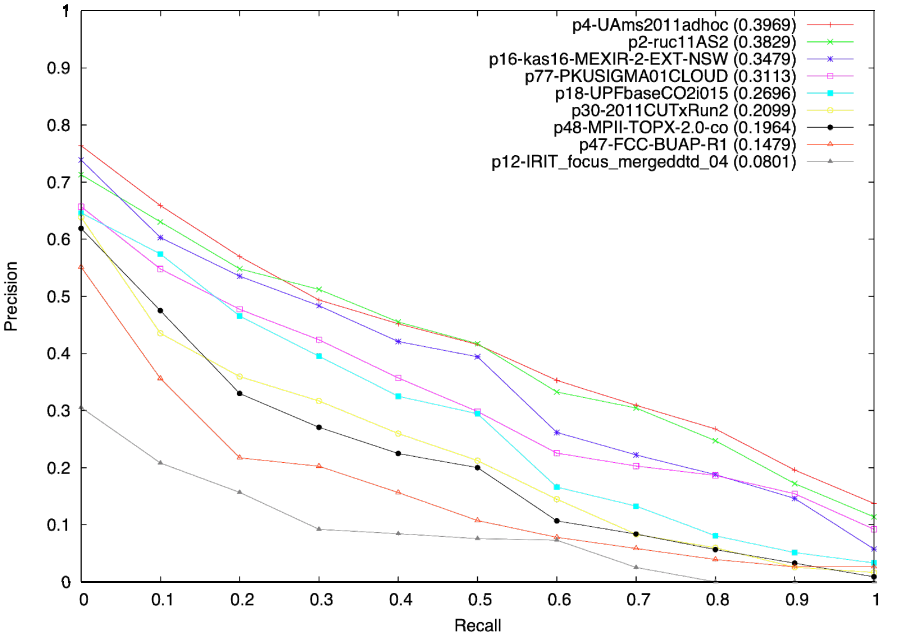


Fig. 4. Best run by each participating institute measured with MAP

Interpolated precision against recall is plotted in Fig. 4, showing quite solid performance for the better scoring runs.

Breakdown over Topic Types

In this section, we will analyze the effectiveness of the runs for each of the four topic types. Let us first analyze the topics and resulting judgments in more details. Table 4 lists the topics per topic type, and Table 5 lists statistics about the number of relevant entities.

Table 4. Breakdown over Topic Types

Topic Type	Topics created	Topics Judged	Topics with relevance
Informational	7	5	5
Known-Item	7	6	6
List	11	9	8
Faceted subtopics	20	18	18
All	45	38	37

Table 5. Relevance per Topic Type

Topic Type	Topics	Min	Max	Median	Mean	Std.	Total
Informational	5	6	327	40	125.8	150.4	629
Known-Item	6	1	416	2	71.3	168.9	428
List	8	5	299	32	98.6	118.1	789
Faceted subtopics	18	23	452	148	168.3	123.8	3,029
All	37	1	452	72	168.3	134.0	4,875

What we see in Table 4 is that we have 5 (informational) to 18 (faceted sub-) topics judged for each type. Given the small number of topics per type, one should be careful with drawing final conclusions based on the analysis, since the particular choice of topics may have had a considerable influence on the outcome.

While all topics have been judged “as is” without special instructions for each of the topic types, the statistics of the relevance judgments in Table 5 is confirming the differences between these topic types. The known-item topics have a median of 2 relevant documents, the list topics have a median of 32 relevant documents, and the informational topics have a median of 40. The faceted (sub)topics, which were based on a general seed topic, have even a median of 148 relevant documents. For all topic types the distribution over topics is skewed, and notable exceptions exist, e.g. a known-item topic with 416 relevant documents.

Table 6 shows the results over only the informational topics. We see that Kasetsart (0.3564), Chemnitz (0.3449), and BUAP (0.3219) now have the best scores, and that there are less differences in scores amongst the top 5 or 6 teams. Over all 34 submissions the system rank correlation (Kendall’s tau) with the ranking over all topics is moderate with 0.512.

Table 7 shows the results over only the known-item topics, now evaluated by the mean reciprocal rank (1/rank). We observe that Amsterdam (0.9167), Renmin (also 0.9167), and MPI (0.7222) now have the best scores. Hence the best teams over all topics also score well over the known-item topics. This is no surprise since the known-item topics tend to lead to relatively higher scores, and hence have a relatively large impact. Over all 34 submissions the system rank correlation based on MAP is 0.572.

Table 6. Best performing runs (only showing one run per group) based on MAP over the 5 informational ad hoc topics

run	map	1/rank	P@10	P@20	P@30
p16-kas16-MEXIR-2-EXT-NSW	**0.3564**	0.8000	0.5000	0.4200	0.3600
p30-2011CUTxRun2	0.3449	0.7067	0.5000	**0.4700**	0.4333
p47-FCC-BUAP-R1	0.3219	**1.0000**	**0.5600**	0.4300	0.4133
p2-ruc11AMS	0.3189	0.6500	0.4200	0.4500	**0.4600**
p4-UAms2011adhoc	0.3079	0.6750	0.3800	0.3100	0.2600
p18-UPFbaseCO2i015	0.2576	0.6346	0.4600	0.4400	0.3800
p77-PKUSIGMA02CLOUD	0.2118	0.5015	0.4400	0.4200	0.3133
p48-MPII-TOPX-2.0-co	0.0900	0.3890	0.2600	0.1800	0.2000
p12-IRIT_focus_mergeddtd_04	0.0366	0.3022	0.2200	0.1100	0.0733

Table 7. Best performing runs (only showing one run per group) based on 1/rank over the 6 known-item ad hoc topics

run	map	1/rank	P@10	P@20	P@30
p4-UAms2011adhoc	**0.8112**	**0.9167**	**0.3167**	**0.2417**	**0.2167**
p2-ruc11AS2	0.7264	**0.9167**	**0.3167**	**0.2417**	**0.2167**
p48-MPII-TOPX-2.0-co	0.2916	0.7222	0.2333	0.1833	0.1778
p18-UPFbaseCO2i015	0.3752	0.7104	0.2500	0.2083	0.1944
p16-kas16-MEXIR-2-EXT-NSW	0.4745	0.6667	0.0833	0.0417	0.0278
p77-PKUSIGMA01CLOUD	0.5492	0.6389	**0.3167**	**0.2417**	**0.2167**
p30-2011CUTxRun2	0.3100	0.5730	0.2667	0.1750	0.1667
p47-FCC-BUAP-R1	0.2500	0.3333	0.0333	0.0167	0.0111
p12-IRIT_large_nodtd_06	0.0221	0.0487	0.0167	0.0333	0.0222

Table 8 shows the results over the list topics, now again evaluated by MAP. We see the best scores for Kasetsart (0.4251), Amsterdam (0.3454), and Peking University (0.3332). The run from Kasetsart outperforms all other runs on all measures for the list topics. Over all 34 submissions the system rank correlation is 0.672.

Table 8. Best performing runs (only showing one run per group) based on MAP over the 8 list ad hoc topics

run	map	1/rank	P@10	P@20	P@30
p16-kas16-MEXIR-2-EXT-NSW	**0.4251**	**0.7778**	**0.4778**	**0.3833**	**0.3741**
p4-UAms2011adhoc	0.3454	0.6674	0.4222	0.3500	0.3222
p77-PKUSIGMA02CLOUD	0.3332	0.5432	0.3889	0.3667	0.3481
p2-ruc11AS2	0.3264	0.6488	0.4111	0.3333	0.2963
p48-MPII-TOPX-2.0-co	0.2578	0.4926	0.3000	0.3333	0.3259
p18-UPFbaseCO2i015	0.2242	0.5756	0.3556	0.3278	0.2741
p12-IRIT_focus_mergeddtd_04	0.1532	0.2542	0.2333	0.2111	0.2148
p30-2011CUTxRun3	0.0847	0.5027	0.1889	0.1611	0.1667
p47-FCC-BUAP-R1	0.0798	0.3902	0.2889	0.2500	0.2259

Table 9 shows the results over the faceted search subtopics (each topic covering only a single aspect). We see the best performance in the runs from Renmin (0.3258), Amsterdam (0.3093), and Peking University (0.3026), with Peking University having clearly the best precision scores. Given that 18 of the 37 topics are in this category, the ranking corresponds reasonably to the ranking over all topics. Over all 34 submissions the system rank correlation is high with 0.818.

Table 9. Best performing runs (only showing one run per group) based on MAP over the 18 facted ad hoc topics

run	map	1/rank	P@10	P@20	P@30
p2-ruc11AS2	**0.3258**	0.5585	0.4722	0.4778	0.4722
p4-UAms2011adhoc	0.3093	0.6492	0.4778	0.4861	0.4500
p77-PKUSIGMA02CLOUD	0.3026	**0.7400**	**0.5722**	**0.5361**	**0.5315**
p16-kas16-MEXIR-2-EXT-NSW	0.2647	0.6443	0.5056	0.4472	0.4000
p18-UPFbaseCO2i015	0.2605	0.5072	0.5278	0.5250	0.5000
p30-2011CUTxRun2	0.2130	0.6941	0.4611	0.4083	0.3741
p48-MPII-TOPX-2.0-co	0.1635	0.6078	0.4778	0.4389	0.4167
p47-FCC-BUAP-R1	0.0995	0.4969	0.4222	0.3333	0.2778
p12-IRIT_focus_mergeddtd_04	0.0810	0.2754	0.2500	0.2278	0.2296

6.2 Faceted Search Results

In the faceted search task, 5 groups submitted 12 valid runs: University of Amsterdam (Jaap), Max-Plank Institute, University of Amsterdam (Maarten), Universitat Pompeu Fabra, and Renmin University of China. All runs are in the format of static hierarchy of facet-values except that one run from Renmin is in the format of a self-implemented faceted search module. So we only present the evaluation results for the 11 static runs. Most of the runs are based on the reference result file provided by Anne Schuth, who generated the reference result file using XPath and Lucene. Two

runs from Amsterdam (Jaap) are based on a result file generated by Indri and one run from Max-Plank Institute is based on the result file generated by TopX.

Among 15 general topics, there are 13 ones that have relevance results. Table 10 shows, for each topic, the number of relevant results, and the rank of the first relevant result in the three result lists generated by Indri, Lucene and TopX respectively. The rank of the first relevant result is in fact the cost that users sequentially scan through the result list to find the first relevant answer without using the faceted-search facility. We call it raw cost, which is actually equal to 1/RR. "-" means that the result file contains no relevant result for this topic. It can be observed that the Indri result file contains relevant results for all topics and ranks them quite high. The TopX result file ranks the first relevant results for 7 topics highest among the three result files, but it fails in containing relevant results for 3 topics. The Lucene reference result file contains no relevant results for 4 topics.

We use two metrics to evaluate the effectiveness of recommended facet-values by each run. One is the interaction cost based on a simple user simulation model, and the other is the NDCG of facet-values [2].

Table 10. Raw costs (1/RR) of faceted search topics on 3 different result files

Topic ID	Number of relevant results	Raw cost of Indri result file	Raw cost of Lucene result file	Raw cost of TopX result file
2011201	48	**45**	-	97
2011202	327	**11**	19	85
2011203	138	**114**	451	-
2011204	342	**306**	989	-
2011205	141	9	316	**1**
2011207	23	69	850	**44**
2011208	285	2	11	**1**
2011209	76	**1**	2	**1**
2011210	23	217	-	**49**
2011211	72	61	45	**40**
2011212	156	1110	-	**344**
2011213	35	828	-	-
2011214	176	**4**	44	16

As described in Section 5, the interaction cost is defined as the number of results, facets or facet-values that the user examined before he/she encounters the first relevant result. This cost can be compared with the raw cost, which is the number of results sequentially examined in the result list without using faceted search facility, to see if the faceted search facility is effective or not. We name their difference as the *Gain* of faceted search. To compare systems across multiple topics, we define the *Normalized Gain (NG)* and *Average Normalized Gain (ANG)* as the following. Note that *NG* is a number between 0 and 1.

$$NG = \max(0, (rawCost - Cost) / rawCost) \quad (1)$$

$$ANG = \frac{1}{|Q|}\sum_{i=1}^{|Q|} NG_i \quad (2)$$

Table 11 shows the evaluation results for all the 11 runs in terms of *NG* and *ANG*. Two runs from Amsterdam (Jaap), p4-UAms2011indri-c-cnt and p4-UAms2011indri-cNO-scr2, are based on the Indri result file, and p48-MPII-TOPX-2.0-facet-entropy (TopX) from Max-Plank is based on the TopX result file. All the other 8 runs are based on the Lucene result file. Because the Indri result file is superior to the TopX and Lucene result files, the two runs based on it also perform better than other runs, and the best one is p4-UAms2011indri-cNO-scr2 (0.35). Among all the 8 runs based on the Lucene result file, p2-2011Simple1Run1 (0.33) from Renmin performs best in terms of ANG. It is followed by p4-UAms2011Lucene-cNO-lth (0.24) from Amsterdam (Jaap), p18-2011UPFfixGDAh2 (0.21) from Universitat Pompeu Fabra and p4-2011IlpsNumdoc (0.20) from Amsterdam (Maarten).

The NDCG scores calculated using the method described in [2] for all 11 static runs are listed in Table 12. For *p* we chose 10 (we thus consider the top 10 documents per facet-value) and we also limited the number of facet-values to be evaluated to 10. Note that we did not evaluate the runs using NRDCG.

Table 11. Evaluation results of all static runs in terms of NGs and ANG

run / NG	p4-UAms2011indri-c-cnt	p4-UAms2011indri-cNO-scr2	p4-UAms2011lucene-cNO-lth	p48-MPII-TOPX-2.0-facet-entropy (TopX)	p48-MPII-TOPX-2.0-facet-entropy (Lucene)	p4-2011IlpsFtScore	p4-2011IlpsNumdoc	p18-2011UPFfixG7DAnh	p18-2011UPFfixGDAh	p18-2011UPFfixGDAh2	p2-2011Simple1Run1
201	0.64	0.60	-	0	-	-	-	-	-	-	-
202	0	0	0.21	0	0	0	0.21	0.11	0.11	0.11	0.21
203	0	0	0	-	0	0.83	0.82	0	0	0	0.86
204	0.63	0.75	0.94	-	0	0	0.90	0	0	0.98	0.91
205	0	0	0.81	0	0.75	0.79	0.72	0.95	0.95	0.95	0.81
207	0	0.77	0	0	0	0	0	0	0	0	0.94
208	0	0	0	0	0	0	0	0	0	0	0
209	0	0	0	0	0	0	0	0	0	0	0
210	0.75	0.74	-	0	-	-	-	-	-	-	-
211	0.18	0	0.53	0	0	0	0	0.71	0.71	0.71	0.60
212	0.89	0.88	-	0	-	-	-	-	-	-	-
213	0.76	0.76	-	-	-	-	-	-	-	-	-
214	0	0	0.64	-	0	0	0	0	0.09	0	0
ANG	0.30	**0.35**	0.24	0	0.06	0.12	0.20	0.14	0.14	0.21	**0.33**

Table 12. Evaluation results for the 11 statics runs in terms of NDCG. Results are per topic and the mean over all topics. Highest scores per topic are highlighted.

run / NDCG	p4-UAms2011indri-c-cnt	p4-UAms2011indri-cNO-scr2	p4-UAms2011lucene-cNO-lth	p48-MPII-TOPX-2.0-facet-entropy (TopX)	p48-MPII-TOPX-2.0-facet-entropy (Lucene)	p4-2011IIpsFtScore	p4-2011IIpsNumdoc	p18-2011UPFfixG7DAnh	p18-2011UPFfixGDAh	p18-2011UPFfixGDAh2	p2-2011Simple1Run1
201	**0.03**	**0.03**	0	0	0	0	0	0	0	0	0
202	0	0	0	0	0	0	0	0	0	0	0
203	0	0	0	0	0	0	0	0	0	0	0
204	0	0	0	0	0	0	0	0	0	0	0
205	0	**0.43**	0.21	0	0	0	0.13	0	0	0	0.07
207	0	**0.16**	0	0	0	0	0	0	0	0	0
208	0	**0.45**	0	0	0	0	0	0	0	0	0
209	0	0	0	0	0	0	**0.24**	0	0	0	0
210	0	0	0	0	0	0	0	0	0	0	0
211	0	0	0	0	0	0	0	0	0	0	0
212	0	0	0	0	0	0	0	0	0	0	0
213	0	0	0	0	0	0	0	0	0	0	0
214	**0.18**	**0.18**	0.09	0	0	0	0	0	0	0	0
mean	0.02	**0.10**	0.02	0	0	0	0.03	0	0	0	0.01

Note that the NDCG calculation used the union of relevance judgments in case there were multiple subtopics for a topic. Statistics for the relevance judgments used for the NDCG evaluation are listed in Table 13.

Table 13. Relevance judgments for faceted search topics

Topic Type	Topics	Min	Max	Median	Mean	Std.	Total
Faceted	13	35	774	156	233	229.3	3029

7 Conclusions and Future Work

We presented an overview of the INEX 2011 Data-Centric Track. This track has successfully run its second year and has introduced a new task, the *faceted search* task. The IMDB collection has now a good set of assessed topics that can be further used for research on richly structured data. Our plan for next year is to extend this collection with related ones such as DBpedia and Wikipedia in order to reproduce a more realistic scenario for the newly introduced faceted search task.

Acknowledgements. Thanks are given to the participants who submitted the topics, runs, and performed the assessment process. Special thanks go to Shlomo Geva for porting the assessment tools, to Anne Schuth, Yu Sun and Yantao Gan for evaluating the faceted search runs, and to Ralf Schenkel for administering the web site.

References

1. Trotman, A., Wang, Q.: Overview of the INEX 2010 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 171–181. Springer, Heidelberg (2011)
2. Schuth, A., Marx, M.: Evaluation Methods for Rankings of Facetvalues for Faceted Search. In: Forner, P., Gonzalo, J., Kekäläinen, J., Lalmas, M., de Rijke, M. (eds.) CLEF 2011. LNCS, vol. 6941, pp. 131–136. Springer, Heidelberg (2011)
3. Koren, J., Zhang, Y., Liu, X.: Personalized Interactive Faceted Search. In: WWW 2008 (2008)
4. Kashyap, A., Hristidis, V., Petropoulos, M.: FACeTOR: Cost-Driven Exploration of Faceted Query Results. In: CIKM 2010 (2010)

Appendix 1: All the Fields or Facets in IMDB Collection

Field Type	Field (or Facet) expressed in XPath
free-text	/movie/title
numerical	/movie/overview/rating
categorical	/movie/overview/directors/director
categorical	/movie/overview/writers/writer
numerical	/movie/overview/releasedates/releasedate
categorical	/movie/overview/genres/genre
free-text	/movie/overview/tagline
free-text	/movie/overview/plot
categorical	/movie/overview/keywords/keyword
categorical	/movie/cast/actors/actor/name
categorical	/movie/cast/actors/actor/character
categorical	/movie/cast/composers/composer
categorical	/movie/cast/editors/editor
categorical	/movie/cast/cinematographers/cinematographer
categorical	/movie/cast/producers/producer
categorical	/movie/cast/production_designers/production_designer
categorical	/movie/cast/costume_designers/costume_designer
categorical	/movie/cast/miscellaneous/person
free-text	/movie/additional_details/aliases/alias
categorical	/movie/additional_details/mpaa
numerical	/movie/additional_details/runtime
categorical	/movie/additional_details/countries/country
categorical	/movie/additional_details/languages/language
categorical	/movie/additional_details/colors/color
categorical	/movie/additional_details/certifications/certification

categorical	/movie/additional_details/locations/location
categorical	/movie/additional_details/companies/company
categorical	/movie/additional_details/distributors/distributor
free-text	/movie/fun_stuff/trivias/trivia
free-text	/movie/fun_stuff/goofs/goof
free-text	/movie/fun_stuff/quotes/quote
categorical	/person/name
categorical	/person/overview/birth_name
numerical	/person/overview/birth_date
numerical	/person/overview/death_date
numerical	/person/overview/height
categorical	/person/overview/spouse
free-text	/person/overview/trademark
free-text	/person/overview/biographies/biography
categorical	/person/overview/nicknames/name
free-text	/person/overview/trivias/trivia
free-text	/person/overview/personal_quotes/quote
free-text	/person/overview/where_are_they_now/where
categorical	/person/overview/alternate_names/name
numerical	/person/overview/salaries/salary
free-text	/person/filmography/act/movie/title
numerical	/person/filmography/act/movie/year
categorical	/person/filmography/act/movie/character
free-text	/person/filmography/direct/movie/title
numerical	/person/filmography/direct/movie/year
categorical	/person/filmography/direct/movie/character
free-text	/person/filmography/write/movie/title
numerical	/person/filmography/write/movie/year
categorical	/person/filmography/write/movie/character
free-text	/person/filmography/compose/movie/title
numerical	/person/filmography/compose/movie/year
categorical	/person/filmography/compose/movie/character
free-text	/person/filmography/edit/movie/title
numerical	/person/filmography/edit/movie/year
categorical	/person/filmography/edit/movie/character
free-text	/person/filmography/produce/movie/title
numerical	/person/filmography/produce/movie/year
categorical	/person/filmography/produce/movie/character
free-text	/person/filmography/production_design/movie/title
numerical	/person/filmography/production_design/movie/year
categorical	/person/filmography/production_design/movie/character
free-text	/person/filmography/cinematograph/movie/title
numerical	/person/filmography/cinematograph/movie/year
categorical	/person/filmography/cinematograph/movie/character
free-text	/person/filmography/costume_design/movie/title

numerical	/person/filmography/costume_design/movie/year
categorical	/person/filmography/costume_design/movie/character
free-text	/person/filmography/miscellaneous/movie/title
numerical	/person/filmography/miscellaneous/movie/year
categorical	/person/filmography/miscellaneous/movie/character
free-text	/person/additional_details/otherworks/otherwork
free-text	/person/additional_details/public_listings/interviews/interview
free-text	/person/additional_details/public_listings/articles/article
free-text	/person/additional_details/public_listings/biography_prints/print
free-text	/person/additional_details/public_listings/biographical_movies/biographical_movie
free-text	/person/additional_details/public_listings/portrayed_ins/portrayed_in
free-text	/person/additional_details/public_listings/magazine_cover_photos/magazine
free-text	/person/additional_details/public_listings/pictorials/pictorial

Edit Distance for XML Information Retrieval: Some Experiments on the Datacentric Track of INEX 2011

Cyril Laitang, Karen Pinel-Sauvagnat, and Mohand Boughanem

IRIT-SIG, 118 route de Narbonne,
31062 Toulouse Cedex 9, France

Abstract. In this paper we present our structured information retrieval model based on subgraphs similarity. Our approach combines a content propagation technique which handles sibling relationships with a document query matching process on structure. The latter is based on tree edit distance (TED) which is the minimum set of insert, delete, and replace operations to turn one tree to another. As the effectiveness of TED relies both on the input tree and the edit costs, we experimented various subtree extraction techniques as well as different costs based on the DTD associated to the Datacentric collection.

1 Introduction

XML documents can be naturally represented through trees in which nodes are elements and edges hierarchical dependencies. Similarly structural constraints of CAS queries can be expressed though trees. Based on these common representations we present a SIR model using both graph theory and content scoring. This paper is organized as follows: Section 2 presents work related to the different steps of our approach; Section 3 presents our approaches and finally Section 4 discusses the results obtained in the Datacentric track of INEX 2011.

2 Related Works

As our algorithm is based on the structure of documents we will first overview document structure extraction techniques. We will then give a brief survey on tree-edit distance algorithms.

2.1 Document Structure Representation and Extraction

In the literature we identify two families of approaches regarding how to handle document structure regardless of content. The first one is relaxation [1] [3] [5]. In these approaches, the main structure is atomized into a set of node-node relationships. The weight of these relationships is then the distance between nodes in the original structure. The second family is linked to subtree extraction. The *lowest common ancestor* (LCA) is the tree rooted by the first common ancestor of two or more selected nodes [4]. In IR it aims at scoring the structure by finding the subtrees in which all the leaves contain at least on term of the query [2].

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 138–145, 2012.

2.2 Edit Distance

Two graphs are isomorphic if they share the same nodes and edges. Finding a degree of isomorphism between two structures is called approximate tree matching. There are three main families of approximate tree matching: *inclusion*, *alignment* and *edit distance*. We choose to use the latter as it offers the most general application. The original tree edit distance (TED) algorithms [14] generalize Levenshtein *edit distance*[10] to trees. The similarity is the minimal set of operations (adding, removing and relabeling) to turn one tree to another. Later, Klein et al. [9] reduced the overall complexity in time and space by splitting the tree structure based on the heavy path (defined in Section 3.2). Finally Touzet et al. [6] used a decomposition strategy to dynamically select the best nodes to recurse on between rightmost and leftmost. Regarding the costs, a common practice is to use *apriori* fixed costs for the primitive operations [11]. However as these costs impact the isomorphism evaluation one can find some approaches that try to estimate these costs. Most of the non-deterministic approaches are based on learning and training techniques [13] [12].

3 Tree-Edit Distance for Structural Document-Query Matching

In our approach the document-query similarity is evaluated by scoring content and structure separately. We then combine these scores to rank relevant elements. In this section, we first describe the content evaluation and then detail our structure matching algorithm based on tree edit distance.

3.1 Content Relevance Score Evaluation

First, we used a $tf \times idf$(Term Frequency $\times$ Inverse Document Frequency [8]) to score the document leaf nodes according to query terms contained in content conditions. We propose two approaches. In the first one (which we call *Vague*) content parts of the query are merged and the content score is evaluated in a one time pass. In our second approach which we define as *Strict*, the content conditions are considered separately and summed at the end of the process.

Our propagation algorithm is based on the idea that the score of an inner node must depend on three elements. First, it must contain its leaves relevance (this is what we call the intermediate score). Second we should score higher a node located near a relevant element. Finally, the node score must depend on the score of its ancestors. Based on these constraints we define the content score $c(n)$ of an element n as the *intermediate content score of the element itself plus its father's intermediate score plus all its father's descendants score*. Recursively, and starting from the document root:

$$c(n) = \begin{cases} \underbrace{\frac{p(n)}{\mid leaves(n) \mid}}_{(i)} + \underbrace{\frac{p(a_1) - p(n)}{\mid leaves(a_1) \mid}}_{(ii)} + \underbrace{\frac{c(a_1) - \frac{p(a_1)}{\mid leaves(a_1) \mid}}{\mid children(a_1) \mid}}_{(iii)} & \text{if } n \neq root \\ \underbrace{\frac{p(n)}{\mid leaves(n) \mid}} & \text{otherwise} \end{cases} \quad (1)$$

(i) is the *intermediate content score* part, with $\mid leaves(n) \mid$ the number of leaf nodes descendants of n and $p(n)$ the intermediate score of the node based on the sum of the scores of all its leaf nodes: $p(n) = \sum_{x \in leaves(n)} p(x)$, with $p(x)$ evaluated using a $tf \times idf$ formula; (ii) is the *neighborhood score* part which allows us to convey a part of the relevance of a sibling node through its father a_1. $p(a_1)$ is the intermediate score of a_1 and $\mid leaves(a_1) \mid$ the number of leaves of a_1; (iii) is the *ancestor scores*, evaluated with $c(a_1)$ the final score of the father a_1 minus its intermediate score.

3.2 Structure Relevance Score Evaluation

Our structural evaluation process follows three steps : subtree selection and extraction, structure score through tree-edit distance, and the score combination. As the final part is strongly related to the type of subtree extracted we will postpone our explanations subtree extraction to the end of this section.

Optimal Path of Tree-Edit Distance. As seen in section 2.2, the TED measures similarity based on the minimal cost of operations to transform one tree to another. The number of subtrees stored in memory during this recursive algorithm depends on the direction we choose when applying the operations. Our algorithm is an extension of the optimal cover strategy from Touzet et al. [6]. The difference is that the optimal path is computed with the help of the *heavy path* introduced by Klein et al. [9]. The heavy path is the path from root to leaf which passes through the rooted subtrees with the maximal cardinality. Selecting always the most distant node from this path allows us to create the minimal set of subtrees in memory during the recursion : this is the *optimal cover strategy*. Formally a heavy path is defined as a set of nodes $[n_1, ..., n_z]$ with $T(x)$ the rooted tree in x, satisfying:

$$\forall (n_i, n_{i+1}) \in heavy \begin{cases} n_{i+1} \in children(n_i) \\ \forall x \in children(n_i), x \notin n_{i+1}, \mid T(n_{i+1}) \mid \geq \mid T(x) \mid \end{cases} \quad (2)$$

This strategy is used on the document and the query as input of our following TED algorithm (Algorithm 1) in which F, G are two forests (i.e. the document and the query as first input), p_F and p_G are positions in O_F and O_G the *optimal paths* (i.e. paths of the *optimal cover strategy*). Function $O.get(p)$ returns the node in path O corresponding to position p.

Algorithm 1. Tree-Edit Distance using Optimal Paths

```
d(F, G, p_F, p_G) begin
    if F = ⊘ then
        if G = ⊘ then
            return 0;
        else
            return d(⊘, G - O_G.get(p_G)), p_F, p_G++) + c_del
            (O_G.get(p_G));
        end
    end
    if G = ⊘ then
        return d(F - O_F.get(p_F)), ⊘, p_F++, p_G) + c_del (O_F.get(p_F));
    end
    a = d(F - O_F.get(p_F), G, p_F++, p_G) + c_del (O_F.get(p_F));
    b = d(F, G - O_F.get(p_F), p_F, p_G++) + c_del (O_G.get(p_G));
    c = d(T(O_F.get(p_F)) - O_F.get(p_F), T(O_G.get(p_G)) - O_G.get(p_G),
    p_F++, p_G++) + d(F - T(O_F.get(p_F)), G - T(O_G.get(p_G)),
    next(p_F), next(p_G)) + c_match (O_F.get(p_F), O_G.get(p_G));
    return min(a, b, c);
end
```

Tree-Edit Distance Costs Evaluation. TED operation costs are generally set to 1 for removing, to 0 for relabeling similar tags and to 1 otherwise [14] which is sufficient for evaluating relatively similar trees. However in our approach document trees are larger than query trees which means that the edit costs must be less discriminative. There is two constraints in estimating these costs. First, as relabeling is equivalent to removing and then adding a node, its cost should be at most equivalent to two removings. Second, we need to reduce the estimation of these costs to the minimum computation cost. For these reasons we propose to use the DTD of the considered collection to create an undirected graph representing all the possible transitions between elements. The idea behind is that the lower degree a node have the less its cost must be. As the Datacentric collection comes up with two distinct DTDs (respectively *movie* and *person*) we choose to create three graphs : one for each DTD and a last one merged on the labels equivalent in the two. In order to process the substitution cost $c_{match}(n_1, n_2)$ of a node n_1 by a node n_2, respectively associated with the tags t_1 and t_2, we seek the shortest path in these DTD graphs through a Floyd-Warshall [7] algorithm which allows to overcome the cycle issues. We divide this distance by the longest of all the shortest paths that can be computed from this node label to any of the other tags in the DTD graph. Formally, with $sp()$ our shortest path algorithm :

$$c_{match}(n_1, n_2) = \frac{sp(t_1, t_2)}{max(sp(t_1, t_x))} \forall x \in DTD \tag{3}$$

The removing cost is the highest cost obtained from all the substitution costs between the current document node and all of the query nodes :

$$c_{del}(n_1) = max(\frac{sp(t_1, t_y)}{max(sp(t_1, t_x))}) \forall x \in DTD; \forall y \in Q \tag{4}$$

Subtree Extraction and Evaluation. In our *Strict* model we use the minimal subtree representing all the relevant nodes labeled with a label contained in the query as input for the matching process. In our *Vague* algorithm we extract all the subtrees rooted from the first node with a label matching a label in the query to the documents root. Formally, for the *Vague* approach, with $Anc(n)$ the set of n ancestors; $a \in Anc(n)$; $T(a)$ the subtree rooted in a; $d(T(a), Q)$ the TED between $T(a)$ and Q, the structure score $s(n)$ is :

$$s(n) = \frac{\sum_{a \in \{n, Anc(n)\}} (1 - \frac{d(T(a),Q)}{|T(a)|})}{\mid Anc(n) \mid} \tag{5}$$

The idea behind this extraction is that a node located near another one matching the structural constraint should get an improvement to its score.

For our *Strict* algorithm the subtree S is created from the combination of all the paths from the deepest relevant nodes which contain a label of the query to the higher in the hierarchy. The subtree is then the merged paths rooted by a node having the same label than the query root. Formally it is composed of all the nodes a extracted as $\{a \in G \mid a \in \{n, Anc(n)\}, \forall n \in leaves \wedge p(n) \neq 0\}$

As we apply one TED for all the nodes in the created subtree, the final score is then :

$$s(n) = \frac{d(S, Q)}{\mid S \mid} \tag{6}$$

3.3 Final Structure and Content Combination

For both models, the final score $score(n)$ for each candidate node n is evaluated through the linear combination of the previously normalized scores $\in [0, 1]$. Formally, with $\lambda \in [0, 1]$:

$$score(n) = \lambda \times c(n) + (1 - \lambda) \times s(n). \tag{7}$$

4 Experiments and Evaluation

In order to evaluate both the efficiency of our algorithms as well as the usefulness of the DTD based costs we run various combinations of our *Strict* and *Vague* algorithms. These runs are *with split DTD* in which we used the three DTD graphs; *with no DTD* for our solution in which the TED operation costs are fixed to 1 for removing a node not in the query, 0.5 for a node with a tag in the query, 0 for a relabeling of one node with another if their label are equivalent and 1 otherwise. λ parameter from the equation (7) is set to 0.4 for *Strict*

approachand 0.6 for the *Vague* one. Finally *baseline* are the runs in which we only use the content part ($\lambda = 1$). As the task is to rank documents, and as our method retrieve elements, we decided to score documents as the score of their best element. Finally it is important to notice the presented runs are not the official ones as the submitted runs for the INEX 2011 Datacentric track were launched over a corrupted index missing around 35% of the documents. Results are presented in table 1. Our *Strict* algorithm scores overall better that the *Vague* version. It tends to demonstrate that using the content location over the structure improves the results. However our TED structure scoring process doesn't seem to improve the search process even if the score tend to drop less with the DTD costs than with the empirically set costs for our *Strict* algorithm. Regarding the *Vague* version we cannot conclude are the results are very similar between our three versions.

Table 1. Our corrected INEX 2011 results with λ set to 0.4 for our *Strict* method and 0.6 for our *Vague* approach compared to our baselines with no use of the DTD

Runs	**MAP**	**P@5**	**P@10**	**P@20**	**P@30**
Strict with split DTD	0.1636	**0.2667**	0.2361	**0.2208**	0.2213
Strict with no DTD	0.125	0.2167	0.1917	0.2069	0.2185
Strict baseline	**0.1756**	0.25	**0.2389**	0.2125	**0.23**
Vague with split DTD	0.105	0.1842	0.1789	0.1605	0.1439
Vague with no DTD	0.1083	0.1895	0.1658	0.1421	0.1289
Vague baseline	0.1104	0.1737	0.1579	0.1382	0.1351

Queries of the Datacentric track were of four types [15]. These types are *Known-item* in which the aim is to retrieve a particular document; *Informational* in which the user tries to find information about a subject or event; *List* for which the aim is to retrieve a list of documents matching a particular subject; and finally *Others* for the queries not listed in one of the previous categories. Results against these types are shown in table 2. It appears that the ranking between the different versions of our algorithm stays the same. However we notice that our *Strict* algorithm scores significantly for the *Know-item* queries. As these types of queries are particularly focused on finding only a few relevant documents this

Table 2. Our corrected INEX 2011 results for the four different types of queries

Runs	**Known-item**	**Informational**	**List**	**Others**
Strict with split DTD	0.3439	0.1082	0.1473	**0.1122**
Strict with no DTD	0.1796	0.1093	0.1043	0.1097
Strict Baseline	**0.3637**	**0.1416**	**0.1646**	0.1002
Vague with split DTD	0.1498	0.0262	0.138	0.086
Vague no DTD	0.1978	0.0254	0.1404	0.0695
Vague baseline	0.1848	0.027	0.1421	0.0772

tends to demonstrate that our strict algorithm is more relevant in the context of a focus search process.

4.1 Conclusions and Future Work

In this paper we presented two XML retrieval models whose main originality is to use graph theory through tree edit distance (TED). We proposed a way of estimating the TED operation costs based on the DTD. It appears that the use of the DTD as well as the edit distance doesn't improve our results for this particular track. In future work we plan to modify our final scoring formula to score documents. Then we will update our leave scoring process by using a more up-to date algorithm such as a language model. Finally we will conduct future studies on Datacentric 2010 to determine if our DTD based edit distance system work betters on element retrieval than it does for document retrieval.

References

1. Alilaouar, A., Sedes, F.: Fuzzy querying of XML documents. In: International Conference on Web Intelligence and Intelligent Agent Technology, Compigne, France, pp. 11–14. IEEE/WIC/ACM (September 2005)
2. Barros, E.G., Moro, M.M., Laender, A.H.F.: An Evaluation Study of Search Algorithms for XML Streams. JIDM 1(3), 487–502 (2010)
3. Ben Aouicha, M., Tmar, M., Boughanem, M.: Flexible document-query matching based on a probabilistic content and structure score combination. In: Symposium on Applied Computing (SAC), Sierre, Switzerland. ACM (March 2010)
4. Bender, M.A., Farach-Colton, M.: The LCA Problem Revisited. In: Gonnet, G.H., Viola, A. (eds.) LATIN 2000. LNCS, vol. 1776, pp. 88–94. Springer, Heidelberg (2000)
5. Damiani, E., Oliboni, B., Tanca, L.: Fuzzy Techniques for XML Data Smushing. In: Reusch, B. (ed.) Fuzzy Days 2001. LNCS, vol. 2206, pp. 637–652. Springer, Heidelberg (2001)
6. Dulucq, S., Touzet, H.: Analysis of Tree Edit Distance Algorithms. In: Baeza-Yates, R., Chávez, E., Crochemore, M. (eds.) CPM 2003. LNCS, vol. 2676, pp. 83–95. Springer, Heidelberg (2003)
7. Floyd, R.W.: Algorithm 97: Shortest path. Commun. ACM 5, 345 (1962)
8. Sparck Jones, K.: Index term weighting. Information Storage and Retrieval 9(11), 619–633 (1973)
9. Klein, P.N.: Computing the Edit-Distance between Unrooted Ordered Trees. In: Bilardi, G., Pietracaprina, A., Italiano, G.F., Pucci, G. (eds.) ESA 1998. LNCS, vol. 1461, pp. 91–102. Springer, Heidelberg (1998)
10. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady 10, 707 (1966)
11. Mehdad, Y.: Automatic cost estimation for tree edit distance using particle swarm optimization. In: Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, ACLShort 2009, pp. 289–292 (2009)

12. Neuhaus, M., Bunke, H.: Automatic learning of cost functions for graph edit distance. Information Science 177(1), 239–247 (2007)
13. Oncina, J., Sebban, M.: Learning stochastic edit distance: Application in handwritten character recognition. Pattern Recogn. 39, 1575–1587 (2006)
14. Tai, K.-C.: The tree-to-tree correction problem. J. ACM 26, 422–433 (1979)
15. Wang, Q., Ramírez, G., Marx, M., Theobald, M., Kamps, J.: Overview of the INEX 2011 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 118–137. Springer, Heidelberg (2012)

UPF at INEX 2011: Books and Social Search Track and Data-Centric Track

Georgina Ramírez

Universitat Pompeu Fabra, Barcelona, Spain
georgina.ramirez@upf.edu

Abstract. This article describes our participation at INEX 2011 in the Books and Social Search track and the Data-Centric track. In the Books and Social Search track we participated in the Social Search for Best Books task and studied the performance effects of using different query fields for query expansion. In the Data-Centric track we participated in the Adhoc task and the Faceted Search task. In the Adhoc task we studied the performance effects of using different indices depending on what type of object is asked for. For the Faceted task we used a fixed set of facets and experimented with the hierarchical and non-hierarchical presentation of them.

Keywords: XML, focused retrieval, INEX, faceted search, query expansion.

1 Introduction

We describe the experiments performed on the INEX 2011 data for the Books and Social Search track and the Data-Centric track. For the Social Search for Best Books task we experiment with the *genre* and *group* fields of the topic description. We investigate whether the terms contained in these fields are useful terms for query expansion. For the Adhoc task of the Data-Centric track we experiment with the use of two different indices: movies and persons. Topics are manually classified and run on one of these indices depending on which type of object the user wants to see. For the Faceted task of the Data-Centric track we use a fixed set of facets and experiment with the hierarchical and non-hierarchical presentation of them.

The rest of the paper is organized as follows: In Section 2 we describe our participation in the Books and Social Search track. Section 3 describes the runs and discusses the results for the two tasks of the Data-Centric track: Adhoc and Faceted. We conclude in Section 4 discussing the main contributions and future work.

2 Social Search for Best Books

From the *Books and Social Search* track we participated in the *Social Search for Best Book* (SB) task. The goal of this task is to investigate the value of user-generated metadata (e.g., reviews and tags) in addition to publisher-supplied

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 146–154, 2012.

and library catalogue metadata, to aid retrieval systems in finding the best, most relevant books for each of the topics of interest.

Thus the task is to find the most relevant books given a specific topic. After performing a few experiments on the training set, we observed the following: 1) There are generally only a few relevant books per topic. Most of the time it is not difficult to find them. The main problem is to rank them high, to obtain a good early precision. 2) When giving emphasis to the words contained in the *title* field we improve precision but we miss some relevant results, we lose in recall. 3) When giving emphasis to the words contained in the *tags* field, we slightly improve both, precision and recall.

Besides expecting to verify whether these observations hold in the larger set, in this track we investigate the following research question: Do the *genre* and *group* fields from the topic description contain useful terms for query expansion?

2.1 Data Collection and Experimental Setup

The social data collection used for this task contains metadata for 2.8 million books crawled from the online book store of Amazon and the social cataloging web site of LibraryThing in February and March 2009 by the University of Duisburg-Essen. The data set is available as a MySQL database or as XML. For each book, the following metadata is included: From Amazon: ISBN, title, binding, label, list price, number of pages, publisher, dimensions, reading level, release date, publication date, edition, Dewey classification, title page images, creators, similar products, height, width, length, weight, reviews (rating, author id, total votes, helpful votes, date, summary, content) editorial reviews (source, content). From LibraryThing: Tags (including occurrence frequency), blurbs, dedications, epigraphs, first words, last words, quotations, series, awards, browse nodes, characters, places, subjects.

For all our experiments in this task we have used the Indri Search Engine [1]. Indri uses a retrieval model based on a combination of language modeling and inference network retrieval frameworks. We have used linear smoothing and lambda 0.2. Topics and documents have been pre-processed using the Krovetz stemmer [2] and the Smart stop-word list [3]. We indexed all the fields in the collection.

2.2 Runs and Results

Runs. The different NEXI [4] queries used for our official runs are presented in Table 1. Runs number 1 and 2 are our baselines and simply give emphasis to the words contained in different fields of the documents. The first one gives a bit more importance to the words contained in the *tags* field. The second one gives emphasis to the words contained in the *tags* and *title* fields. For these runs we use the *title* field of the topic description as query (q).

Table 1. Description of the official runs

Run id	NEXI query
1 UPF_base_BT02	//book[about(., q) AND about (.//tags, q)]
2 UPF_base_BTT02	//book[about(., q) AND about (.//tags, q) AND about (.//title, q)]
3 UPF_QE_genre_BTT02	//book[about(., q+ge) AND about (.//tags, q+ge) AND about (.//title, q+ge)]
4 UPF_QE_group_BTT02	//book[about(., q+gr) AND about (.//tags, q+gr) AND about (.//title, q+gr)]
5 UPF_QE_genregroup_BTT02	//book[about(., q+ge+gr) AND about (.//tags, q+ge+gr) AND about (.//title, q+ge+gr)]
6 UPF_QEGr_BTT02_RM	//book[about(., q+ge+gr) AND about (.//tags, q+ge+gr) AND about (.//title, q+ge+gr)]

Runs number 3, 4, and 5 perform query expansion with the terms contained in the *genre* and *group* fields of the topic description. These runs use the same NEXI query type as run 2 and should give us some insight whether the terms contained in these fields are useful terms for query expansion.

Run number 6 is the same as run number 5 applying a post-processing step were all isbn numbers from the *similar* and *dissimilar* fields of the topic description are removed. Our assumption for this run is that the books contained in the *similar* and *dissimilar* fields are examples of books that the user gives to help with the search but are not books the user wants to see in the result set. Note that only 56 out of the 211 topics contain any such information in their description.

Results. The results of our official runs are shown in Table 2.

Our best run is the one that performs query expansion using the terms contained in the *group* field of the topic (fourth row). This run improves substantially over its baseline (second row). However, it performs very poorly compared to the best performing run in this task (last row). Trying to understand why the performance of our baselines are so poor, we realized that the Indri Search Engine uses a different retrieval model when structured queries are given. This retrieval model seems to perform worse than the one used when non-structured queries are given.

Note that when removing the similar and dissimilar books from our best run, we obtain a much worse overall performance (sixth row). This indicates that our assumption that the user does not want to see the examples he or she suggests is not true.

Table 2. Official results for the SB runs

Run id	MAP	MRR	P@10	P@20	R-prec
UPF_base_BT02	0.1048	0.2039	0.0796	0.0756	0.0949
UPF_base_BTT02	0.1018	0.2135	0.0863	0.0706	0.0909
UPF_QE_genre_BTT02	0.0910	0.2089	0.0844	0.0725	0.0841
UPF_QE_group_BTT02	**0.1223**	**0.2478**	**0.0995**	**0.0834**	**0.1265**
UPF_QE_genregroup_BTT02	0.1001	0.2283	0.0934	0.0787	0.1042
UPF_QEGr_BTT02_RM	0.0973	0.2183	0.0872	0.0718	0.1049
Best performing run at INEX 2011	0.2283	0.4811	0.2071	0.1569	0.2225

Performing query expansion with the *genre* information performs much worse than performing it with the *group* one. That is probably because the terms contained in the *genre* field are generally more generic. For example, in topic number 399 the *title* is "cakes", the *group* is "sweet treat" and the *genre* is "technology home economics cooking". It could be that when adding the *genre* terms to the query, terms such as *technology* or *economics* introduce some noise (i.e., irrelevant results). Further analysis needs to be done in order to confirm this hypothesis.

Having a look at the relevance assessments we can see that there are very few relevant documents per topic, an average of 11.3 (median 7). This confirms our observation from the training set. Furthermore, for more than a third of the topics, the task can be seen as a know-item search since a very small set of relevant results can be found: 17 topics have only 1 relevant result, 37 topics have 1 or 2 relevant results, and 79 topics have less or 5 relevant results.

3 Data-Centric track

The goal of the Data-Centric Track is to investigate retrieval over a strongly structured collection of documents, in particular, the IMDB collection. The track features two tasks. In the *Ad Hoc* Search Task retrieval systems are asked to answer informational requests with the entities contained in the collection (movies, actors, directors, etc.). In the *Faceted* Search Task retrieval systems are asked to provide a restricted list of facets and facet-values that will optimally guide the searcher toward relevant information.

3.1 Data Collection and Experimental Setup

The track uses the IMDB data collection generated from the plain text files published on the IMDb web site on April 10, 2010. There are two kinds of objects in the collection, movies and persons involved in movies, e.g. actors/actresses, directors, producers and so on. Each object is richly structured. For example, each movie has title, rating, directors, actors, plot, keywords, genres, release dates, trivia, etc.; and each person has name, birth date, biography, filmography,

etc. Each XML file contains information about one object, i.e. a single movie or person. In total, the IMDB data collection contains 4,418,081 XML files, including 1,594,513 movies, 1,872,471 actors, 129,137 directors who did not act in any movie, 178,117 producers who did not direct nor act in any movie, and 643,843 other people involved in movies who did not produce nor direct nor act in any movie.

For all our experiments in this task we have used the Indri Search Engine [1]. Indri uses a retrieval model based on a combination of language modeling and inference network retrieval frameworks. We have used linear smoothing and lambda 0.15 (based on our experiments from last year in the same collection). Topics and documents have been pre-processed using the Krovetz stemmer [2] and the Smart stop-word list [3].

We created two different indices for the collection: one for movies and one for persons. We indexed all fields for both types of documents.

3.2 Adhoc Search Runs and Results

Runs. Our approach for the *Adhoc Search* task is based on the results obtained from our participation in the track last year, where we found out that indexing only the *movie* documents of the collection performed much better than indexing the whole collection, one of the best performing runs of last year [5]. For this year, we wanted to check whether we can improve those results by running topics in different indices according to the object they ask for.

Therefore, we manually classified all topics according to which type of object the users are searching for: movies or persons. We did that by looking at the *description* and the *castitle* fields of the topic. We found out that only 9 out of the 45 topics were asking for persons. From those 9 topics only 7 were assessed. All this topics had a CAS title of the form //person[about()] or //actor[about()] which made topic classification a rather easy task. The description of these topics and the number and type of documents found relevant for each of them are shown in Table 3.

We then ran each topic on its specific index, either movies or persons, and merged the results for submission. We only submitted two runs, both following the same approach. The first one, UPFbaseCO2i015, uses the CO title of the topic while the second one, UPFbaseCAS2i015, uses the CAS one.

Results. The results of our official runs are shown in Table 4.

We can see that using the CO title of the topic performs much better than using the CAS version of it. That could be due to the structural strictness with which the query is treated. However, results are not comparable because Indri uses a different retrieval model when the queries introduced are structured. Thus, it could be simply that the retrieval model for structured queries performs worse.

Our runs are clearly better at precision than recall (when looking at their ranking position). This suggests that the use of independent indices might indeed help to improve early precision. The low performance at high recall levels

Table 3. Description of the topics classified as searching for person objects and number of movie and person documents found relevant for each of them

Topic id	Description	Movie	Pers.
2011103	Name all actors that played the role of James Bond	0	6
2011107	I want to know more about Tom Hanks	1	7
2011126	I want to find all German fellow actors of Mel Gibson	0	0
2011131	List all actors that played in the three Lord of the Ring movies "The Fellowship of the Ring", "The Return of the King" and "The Two Towers"	24	392
2011133	I want a list of actors that played in the tvseries Friends	0	5
2011142	I'm looking for information about actors that were born in Vietnam	134	100
2011144	I'm looking for information about members of the Cannes jury	44	255

Table 4. Official results for the Data-Centric, adhoc search runs. The number in parentheses indicates the run position in the official ranking.

Run id	MAP	P@5	P@10	P@20	P@30
UPFbCO2i015	**0.2696** (9)	**0.4211** (9)	**0.4342** (5)	**0.4171** (3)	**0.3825** (5)
UPFbCAS2i015	0.1117 (26)	0.3579 (18)	0.3474 (14)	0.3211 (13)	0.3070 (13)

could be due to the strictness of retrieving only one type of object per topic. Many topic authors have assessed both types of objects as relevant. For instance, when looking at the topics that ask for person objects (see Table 3) we find that some of these topics have only person objects assessed as relevant (e.g., topics 2011103 and 2011133). However, some others have also many movie objects assessed as relevant (e.g. topics 2011142 and 2011142). For these topics we perform poorly on recall since we do not retrieve any movie element. The same happens with the topics that ask for movie objects. Some of them have more person objects assessed as relevant than movie ones. For example, topic number 2011118 which description is *I am searching for movies about musicals from Andrew Llyod Webber*, found 34 movie documents relevant and 104 person documents relevant. The same happens with topic number 2011122 which description is *I want to know everything about movies that deal with the nuclear-accident in Chernobyl*. This topic has 90 movie documents assessed as relevant against 209 person documents found relevant. A possible way to solve this problem is to run all topics on both indices but giving priority to the one is being asked for. In this way we might improve recall. We leave this experimentation for future work.

3.3 Faceted Search Runs and Results

Runs. For the *Faceted search* task we used a fixed set of facets, namely *directors*, *actors*, and *genres*, and experiment with the hierarchical and non-hierarchical presentation of them. For that, we first took the reference run and extracted

the most popular values for each of our facets. By popular we mean the most repeated facet-value in the result set. We then presented these facet-values in different ways.

We submitted three runs. The parameters and description of our runs can be found in Table 5.

Table 5. Description of the official faceted search runs

Run id	Type	facet-value number
1 UPFfixGDAnh	Non-hierarchical	genre (7), director (5), actor (8)
2 UPFfixGDAh	Hierarchical	genre (7), director (5), actor (8)
3 UPFfixGDAh2	Hierarchical	genre (20), director (20), actor (20)

The first run is a non-hierarchical run, which means that it presents all facet-value pairs at the same level. Thus, we return 20 facet-value pairs per topic. The second and third run are hierarchical, meaning that for each genre-value pair, we return all director-value pairs, and within each director-value pair we return all actor-value pairs. Thus we return 280 and 8000 results per topic respectively.

Results. Two different measures have been used to evaluate Faceted Runs. One is the interaction cost based on a simple user simulation model (estimated as ANG), and the other is the NDCG of facet-values [6]. See the overview of the track for more details [7]. The official results of our runs are shown in Table 6.

Table 6. Official results for the Data-Centric, faceted search runs. The number in parentheses indicates the run position in the official ranking.

Run id	ANG	NDCG
UPFfixGDAnh	0.14 (8)	0 (6)
UPFfixGDAh	0.14 (7)	0 (6)
UPFfixGDAh2	0.21 (5)	0 (6)

According to the first measure (ANG), our third run performs the best. It ranks position 5 out of 11 submitted runs. The other two runs seem to perform similar. However, the second run is slightly better than the first (when looking at the topic per topic performance or using more decimals in the averaged score). Although these results could suggest that presentation of the faceted-values in a hierarchical manner does help the user to find faster the relevant documents, it could also be that the more results we return the better. Thus our third run would outperform the other two simply because it returns many more facet-value pairs.

All of our runs had a zero score when using the second measure. The same happened to half of the submitted runs. Furthermore, in 8 out of 14 topics all the 11 runs get a 0 score. That means that only in 6 of the topics some difference between runs can be measured. It is therefore quite difficult to draw any conclusions from this measure. It should probably be tuned to become more sensitive to systems performance on this task since hardly any difference can be appreciated.

4 Conclusions and Future Work

This paper described our participation at INEX 2011. We participated in the Books and Social Search track and the Data-Centric track. In the Social Search for Best Books task we have seen that the the terms contained in the *group* field of the topic are useful terms for query expansion; they help to improve the overall retrieval performance. This is not the case for the terms contained in the *genre* field which seem to be more generic and introduce noise. In the Adhoc task of the Data-Centric track we experimented with the use of two different indices: movies and persons. We have seen that the use of the two indices does help to improve precision. However, our approach is maybe to strict regarding recall. We plan to investigate whether running all topics on both indices but giving priority to the one that is being asked for could further improve retrieval performance. For the Faceted task of the Data-Centric track we used a fixed set of facets and experimented with the hierarchical and non-hierarchical presentation of them. Although results suggest that hierarchical presentation of the results helps users to find the relevant documents faster, we argue that evaluation measures should be further tested and developed before drawing conclusions from the results.

Acknowledgments. This work has been supported by the Spanish Ministry of Science and Education under the HIPERGRAPH project and the Juan de la Cierva Program.

References

1. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: a language model based search engine for complex queries. In: Proceedings of the International Conference on Intelligent Analysis (2005)
2. Krovetz, R.: Viewing morphology as an inference process. In: Proc. of the 16th ACM SIGIR Conference, Pittsburgh, June 27-July 1, pp. 191–202 (1993)
3. Salton, G.: The SMART Retrieval SystemExperiments in Automatic Document Processing. Prentice-Hall, Inc., Upper Saddle River (1971)
4. Trotman, A., Sigurbjörnsson, B.: Narrowed Extended XPath I (NEXI). In: Fuhr, N., Lalmas, M., Malik, S., Szlávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 16–40. Springer, Heidelberg (2005)
5. Ramírez, G.: UPF at INEX 2010: Towards Query-Type Based Focused Retrieval. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 206–218. Springer, Heidelberg (2011)

6. Schuth, A., Marx, M.: Evaluation Methods for Rankings of Facetvalues for Faceted Search. In: Forner, P., Gonzalo, J., Kekäläinen, J., Lalmas, M., de Rijke, M. (eds.) CLEF 2011. LNCS, vol. 6941, pp. 131–136. Springer, Heidelberg (2011)
7. Wang, Q., Ramírez, G., Marx, M., Theobald, M., Kamps, J.: Overview of the INEX 2011 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 118–137. Springer, Heidelberg (2012)

University of Amsterdam Data Centric Ad Hoc and Faceted Search Runs

Anne Schuth and Maarten Marx

ISLA, University of Amsterdam, The Netherlands
{anneschuth,maartenmarx}@uva.nl

Abstract. We describe the ad hoc and faceted search runs for the 2011 INEX data centric task that were submitted by the ILPS group of the University of Amsterdam. In our runs, we translate the content-and-structure queries into XQuery with full-text expressions and process them using the eXist+Lucene XQuery with full-text processor.

1 Introduction

The data centric track at INEX uses a very structured, indeed data centric, collection of XML to evaluate both ad hoc and faceted retrieval. The collection used is an XML version of the IMDB database (`http://www.imdb.com`). It consists of 4.4M XML files, of which 1.6M have root `movie` and all others have root `person`. The XML files contain a rich database-like structure, with functional elements (e.g. `title` and `rating`, `birth date`), elements with restricted content (e.g. dates, rating) and elements with textual content.

Ad hoc search. The ad hoc search task within the data centric task contains 45 information needs, called topics. The information need is described in natural language and formalised in two ways: as a keyword query (CO), and as a NEXI expression [9] (CAS). All CAS topics start with `//movie` or `//person`, and thus ask for either movies or persons. This makes the ad hoc search task in essence a document retrieval task. The key difference with standard ad hoc retrieval is the availability of structure, both in the documents and in the queries.

NEXI queries can be represented as unions of tree-patterns [1] having only the descendant axis and having an additional `about(.,'text')` function attached to nodes [6]. That the ad hoc track is really a document retrieval task can be seen from the fact that the output node of all tree patterns is the root. This means that in the classification of [3], they are either "contextual content information" or "search hint" queries, depending on the depth of the tree pattern.

Thus the data set and the topics make this ad hoc search task more look like a database than an information retrieval task. The ILPS group at UvA is using an open search XML database system, eXist [5], with XQuery full text search implemented using Lucene in a number of projects which involve complex content and structure search [4] We also contributed to the eXist code by adding a module for faceted search [8]. We wanted to measure the quality of information retrieval of eXist with Lucene by running the CAS queries directly.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 155–160, 2012.

For ad hoc search, we wanted to answer the following two research questions:

R1. Is it possible to create a good performing ad hoc search run without any manual intervention using an out-of-the-box XQuery with full text implementation (eXist with Lucene)?

R2. Can we improve on the "treat CAS-queries exactly as they are stated"-run by mixing in full document scores?

Faceted search. Faceted search systems [2] provide a solution to the problem of dealing with ambiguous queries. Users with a very specific information need may use broad queries to initially express their need. Systems responding to such a broad query generally know nothing about the user and should thus cater for all conceivable information needs. Faceted search systems achieve this by suggesting a ranking over query refinements taken from values in metadata attached to documents. These query refinement suggestions—which we refer to as *facet values*—act as Boolean filters on the data; a user is guided to the subset of documents relevant to his information need by subsequently selecting some facet values. After a user submits a query, in addition to a ranked list of documents, a faceted search system will also return a ranked list of facet values for this query. The top n facet values will be shown on the search engine result page (SERP). Each facet value is shown together with an integer indicating the number of hits which would result from refining the query with this value.

Research by Hearst [2] has shown that users can cope with just a small number of facets and values next to the search hits on the SERP. Selecting a small number of facets and values which best help a user in her search process is a difficult task. Within the IMDB data set almost every XML element can be used as a facet, and most have a large number of different values.

The task which is evaluated at INEX faceted search is the ranking of facet values given a query. Ideally, facet values are ordered in such a way that the top facet values each cover a distinct set of documents relevant to an aspect of the ambiguous query. At the same time, the subset of documents relevant to one of the conceivable information needs expressed through the query should be reachable through the selection of a minimal number of facet values.

In a Boolean retrieval system without ranking of documents there is a natural ranking of the facet-values: by the number of hits for which the facet value holds. In a retrieval system in which not all hits are equal, orderings which take document-scores into account might perform better. This is the question we wanted to address in the faceted search task:

R3. Can a ranking of facet-values based on hit-counts be improved by also considering the document scores of the hits?

We return to our research questions in Section 3. First we discuss the submitted runs.

2 Description of the Runs

As our XPath/XQuery processor we used eXist version 1.4.1, a native XML database [5]. Running NEXI queries [9] —all CAS titles are NEXI expressions— is easy in eXist because it supports full integration of XQuery with full-text search. Full-text search is implemented using Lucene 2.9.2. We defined Lucene indexes on all elements E, for which a topic exists which checks whether E is `about()` some full text expression T. This includes the two "document-indexes" on `movie` and `person` elements. Note that this step depends on the set of queries.

From NEXI to XPath with Full Text. The INEX NEXI queries can be automatically translated without change of meaning into the specific syntax of XQuery with full-text search employed by eXist. Table 1 describes the rewrite rules used. It is tempting to think that translating `about(P,T)` into `contains(P,'T')` results in an XPath query which would exactly express the Boolean Search interpretation of the NEXI query. Recall that `contains(P,'T')` evaluates to true if and only if the string `T` is a substring of the atomized value of the path expression `P`. When `P` denotes a single element, this is just the concatenation of all text() elements below it. But when `P` denotes a sequence this is not defined and `contains(P,'T')` produces a runtime error and thus no results at all. This is easily fixed in the translation because the meaning of `contains(P,'T')` inside a filter expression is equal to `P[contains(.,'T')]`. To see this, note that `contains(P,'T')` is true at a context node n if and only if there exists a node reachable by path expression `P` from n, whose atomic value contains the string `T`. This is exactly the truth definition of the expression `.[P[contains(.,'T')]]`. However the semantics of the the eXist function `ft:query` already internalizes this existential quantification and thus for `ft:query` this extra step is not needed.

Table 1. Rewrite rules for CAS titles

before	after
`OR`	`or`
`AND`	`and`
`about(P,T)`	`ft:query(P,'T')`

For our runs mixing element-scores and document scores, we also need a version of the query in which all structure is removed. Thus a CO query, but written formally. (Note that we slightly extend the usual INEX meaning of a CO query: we allow a Boolean combination of keyword queries.) We obtain this query automatically from the CAS query. This has the advantage that Boolean logic in the query is preserved. The transformation first replaces all paths in the query by '.', and then replaces expressions of the form `.[Q1]/.[Q2]` by `.[Q1 and Q2]`. Boolean and's and or's were kept. For example (based on the Dr Gonzo query),

CAS `//A[ about(.//B,'b')]//C[ about(.//D,'d') or about(.//E,'e')]`
CO `.[about(.,'b') and ( about(.,'d') or about(.,'c') )]`

The result is a Boolean combination of `about()` queries on one element. If we evaluate this query on the root of an XML document, and there is a Lucene full text index on the root, this has exactly the effect of evaluating the query on the complete textual content of the XML document.

Both our ad hoc and faceted search runs are implemented as XQueries. We now describe the specific settings used.

Ad hoc runs. For ad hoc, we created mixture models. Experiments with estimating the best value for λ on the INEX 2010 collection had no success. Thus we used the extreme values: 0.0, 0.5 and 1.0. For each document d, and query Q, we calculate the Document Score $ds(d, Q)$ and the Element Score $es(d, Q)$, as follows

$ds(d, Q)$ is simply the Lucene score of the document for the CO version Q' of the CAS query Q. For `$d` a document and `Q` a NEXI expression, this score is available in eXist by the function `ft:score($d/Q)`).

$es(d, Q)$ is the maximum Lucene-score for any element in document d answering the NEXI query.
For `$d` a document and `Q` a NEXI expression, this score is calculated with the XPath expression `max( for $e in $d/Q return ft:score($e) )`.

The score per document is calculated as

$$d(d, Q) = \lambda \cdot ds(d, Q) + 1 - \lambda \cdot es(d, Q).$$

Faceted search runs Out of all XML elements in the IMDB files, we manually defined a limited number of facets that could be used. We selected facets in such a way that each can cover a broad range of values while the values on this facet are not unique for a single document. Movies and persons have different facets. For each facet, we define its name and an XPath expression by which the facet values can be retrieved. The possible values are dictated by the data. We list our selection in Table 2.

A facet value now is a combination of a facet name and a value. For instance, the pair "country: Belgium" is a legal facet value for movies.

We defined two strategies for the selection and ordering of facet values. The first system, which we will call *count*, ranks facet values based on the number of —not necessarily relevant— hits that would be the result if the facet value were selected. This is captured by the following calculation: (here the variable `$hits` contains all documents with a Lucene score higher than 0)

```
count($hits[facet-path eq $value])
```

This strategy treats each hit as an equally good hit.

Our second strategy takes relevance into account: for each facet value we calculate the amount of relevance in the top hits of the query filtered through

Table 2. Names and paths, defined as XPath expressions, of our selections of facets. Some apply to topics that ask for movies, others to topics that ask for persons, there is no overlap.

name	facet-path	applies to
director	.//directors/director	movie
writer	.//writers/writer	movie
genre	.//genres/genre	movie
keyword	.//keywords/keyword	movie
actor	.//cast/actors/actor/name	movie
producer	.//cast/producers/producer	movie
certification	.//certifications/certification	movie
language	.//additional_details/languages/language	movie
country	.//additional_details/countries/country	movie
name	.//person/name	person
height	.//height	person
other-movie	.//filmography/act/movie/title	person
other-movie-year	.//filmography/act/movie/year	person
nickname	.//nicknames/name	person

the facet value. In our run we have used the top 10 documents with the highest Lucene score. We simply take the sum of the Lucene scores of the top 10 documents. Facet values were then ranked based on this *sumscore*.

Both runs use the provided standard run for retrieval of the documents.

Results

Both absolutely and relatively our ad hoc run scored worse than in INEX 2010. Our best run was the pure element based run with $\lambda = 0$. This more precision oriented run received a MAP score of .25, making it 11th out of the 34 submitted runs (maximum MAP was .40). Precision ranges from .38 for p@5 (max: .52) to .34 (max: .41) for p@30. Our second best run was the mixed run.

The results on the faceted search runs are difficult to interpret because of the high number of null values. On both Average Normalized gain and the mean NDCG measure our best run is located in the middle of the ranking of systems. On both measures, the run based on hit counts outperformed the run which took relevance scores into account.

3 Discussion and Conclusions

Our ad hoc search run showed that a CAS query taken as is and evaluated with an out-of-the-box XQuery with Full Text system performs relatively well. The recall boosting technique of removing the structural hints from the CAS query deteriorated performance. Thus better stick to the query stated by the user. This means that we can answer our research question **R1** positively and **R2** negatively.

The results from the faceted search task seem to imply that the simple way of counting hits for choosing facet values works best. This goes against our hypothesis that taking relevance into account will give a better score. Actually, earlier experiments on the INEX 2010 collection gave the opposite results [7], but there we summed the scores of all relevant documents, not just the top ten as we did here. Clearly, using the top ten hits with a retrieval run with on average only 3.4 relevant hits in the top 10 is not going to be very robust. Thus the results so far lead us to answer research question **R3** negatively.

Acknowledgements. The authors acknowledge the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under the FET-Open grant agreement FOX, number FP7-ICT-233599. This research was supported by the Netherlands organization for Scientific Research (NWO) under project number 380-52-005 (PoliticalMashup).

References

1. Amer-Yahia, S., Cho, S., Lakshmanan, L.V.S., Srivastava, D.: Tree pattern query minimization. The VLDB Journal 11, 315–331 (2002)
2. Hearst, M.: Search User Interfaces. Cambridge University Press (2009)
3. Kamps, J., Marx, M., de Rijke, M., Sigurbjörnsson, B.: Articulating information needs in XML query languages. ACM Trans. Inf. Syst. 24(4), 407–436 (2006)
4. Kaptein, R., Marx, M.: Focused retrieval and result aggregation with political data. Information Retrieval 13(5), 412–433 (2010), http://www.springerlink.com/content/3321087120654370/
5. Meier, W.: eXist: An Open Source Native XML Database. In: Chaudhri, A.B., Jeckle, M., Rahm, E., Unland, R. (eds.) NODe-WS 2002. LNCS, vol. 2593, pp. 169–183. Springer, Heidelberg (2003)
6. Peetz, H., Marx, M.: Tree patterns with full text search. In: Proceedings WebDB 2010 (2010)
7. Schuth, A., Marx, M.: Evaluation Methods for Rankings of Facetvalues for Faceted Search. In: Forner, P., Gonzalo, J., Kekäläinen, J., Lalmas, M., de Rijke, M. (eds.) CLEF 2011. LNCS, vol. 6941, pp. 131–136. Springer, Heidelberg (2011)
8. Schuth, A., Marx, M.: Fast Faceted Search in XML. In: Proc. XML Prague 2011 (2011)
9. Trotman, A., Sigurbjörnsson, B.: Narrowed Extended XPath I (NEXI). In: Fuhr, N., Lalmas, M., Malik, S., Szlávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 16–40. Springer, Heidelberg (2005)

BUAP: A Recursive Approach to the Data-Centric Track of INEX 2011*

Darnes Vilariño Ayala, David Pinto, Saúl León Silverio[1], Esteban Castillo[3], and Mireya Tovar Vidal

Facultad de Ciencias de la Computación
Benemérita Universidad Autónoma de Puebla, México
{darnes,dpinto,mtovar}@cs.buap.mx, saul.ls@live.com, ecjbuap@gmail.com

Abstract. A recursive approach for keyword search on XML data for the Ad-Hoc Search Task of INEX 2011 is presented in this paper. The aim of this approach was to detect the concrete part (in the representation tree) of the XML document containing the expected answer. For this purpose, we initially obtain a tree structure, which represents an XML document, tagged by levels. A typical search engine based on posting lists is used in order to determine those documents that match in some degree with the terms appearing in the given query(topic). Thereafter, in a recursively process, we navigate into the tree structure until we find the best match for the topic. The obtained results are shown and compared with the best overall submission score obtained in the competition.

1 Introduction

The Data-Centric track, introduced first at 2010 and now presented in its second edition at INEX 2011, aims to provide a common forum for researchers or users to compare different retrieval techniques on Data-Centric XML, thus promoting the research work in this field [1,2]. Compared with the traditional information retrieval process, where whole documents are usually indexed and retrieved as single complete units, information retrieval from XML documents creates additional retrieval challenges. In the task of document-centric XML keyword search, a simple structure of long text field predominates, however, in Data-Centric XML, the structure of document representation is very rich and carries important information about objects and their relationships [3].

The Data-Centric track uses the IMDB data collection gathered from the following website: http://www.imdb.com. It consists of information about more than 1,590,000 movies and people involved in movies, e.g. actors/actresses, directors, producers and so on. Each document is richly structured. For example, each movie has title, rating, directors, actors, plot, keywords, genres, release dates, trivia, etc.; and each person has name, birth date, biography, filmography, and so on. A best description of this dataset can be found at [2].

* This work has been partially supported by the CONACYT project #106625, VIEP # VIAD-ING11-II and # PIAD-ING11-II, as well as by the PROMEP/103.5/09/4213 grant.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 161–166, 2012.

The Data-Centric track aims to investigate techniques for finding information by using queries considering content and structure. Participating groups have contributed to topic development and evaluation, which will then allow them to compare the effectiveness of their XML retrieval techniques for the Data-Centric task. This process has lead to the development of a standard test collection allowing participating groups to undertake future comparative experiments.

The rest of this paper is structured as follows. Section 2 describes the approach used for preparing the run submitted to the competition. Section 3 show the results obtained, as well as the comparison scores reported by other teams. Finally, in Section 4 we discuss findings and future work.

2 Description of the System

In this section we describe how we have indexed the corpus provided by the task organizers. Moreover, we present the algorithms developed for tackling the problem of searching information based on structure and content. The original XML file has been previously processed in order to eliminate stopwords and punctuation symbols. Additionally, we have transformed the original hierarchical structure given by the XML tags to a simple string containing both the corresponding XML tag and a numeric value which indicates the level at the original structure. In Figure 1 we may see the original structure of a part of an XML document ([XML]). The same Figure depicts the corresponding strings obtained as a result of the document representation transformation ([TXT]).

For the presented approach we have used an inverted index tree in order to store the XML templates of the corpus. In this kind of data structure we have considered to include both, the term and the XML tag (with its corresponding hierarchy). The aim was to be able to find the correct position of each term in the XML hierarchy and, therefore, to retrieve those parts of the XML file containing the correct answer for a given query. The numeric value introduced in the document representation, mentioned above, allows to store the same term in the inverted index, even when this term occurs in different contexts. In Figure 2, we show an example of the inverted index. We may see that the dictionary entry "person.overview.alternate_names.name[1].smith" refers to the term "smith" which has a document frequency of 699, a term frequency of 1 at the document identified as "person_990001", etc. The complete name of this "smith" instance is "smith kamani ray", but there is, at least at the example showed, another "smith" whose complete name is "smith kimani". Therefore, the numeric value introduced allows to avoid confusions for identifying each of the different instances.

We have created five different inverted indexes, for the each one of the following categories: actors, directors, movies, producers and others. Based on the query, we identify a single possible relevant index and, thereafter, the query is executed only on this relevant index.

```
[XML]
      <alternate_names>
         <name>Smith, Kamani Ray
         </name>
         <name>Smith, Kimani
         </name>
         <name>Smithlou, Kimani Ray
         </name>
      </alternate_names>

[TXT]
person.overview.alternate_names.name[1] smith
person.overview.alternate_names.name[1] kamani
person.overview.alternate_names.name[1] ray
person.overview.alternate_names.name[2] smith
person.overview.alternate_names.name[2] kimani
person.overview.alternate_names.name[3] smithlou
person.overview.alternate_names.name[3] kimani
person.overview.alternate_names.name[3] ray
       :                                          :
```

Fig. 1. Example of the transformation for the document representation

```
person.overview.alternate_names.name[1].smith : (699) person_990001:1,
                                                      person_993004:1 ...

       :                                          :
```

Fig. 2. Example of the type of inverted index used in the experiments

Once the dataset was indexed we may be able to respond to the query. In this case, we have also processed the query by identifying the corresponding logical operators (AND, OR) in a recursive manner, i.e., we produce answers for the inner operators first, and recursively we merge the results until we reach the external operators, which lead us to obtain the complete evaluation of the query.

In order to obtain the list of candidate documents for each topic, we have calculated the similarity score between the topic and each target document as shown in Eq. (1) [4].

$$\mathrm{SIM}(q,d) = \sum_{c_k \in B} \sum_{c_l \in B} CR(c_k, c_l) \sum_{t \in V} weight(q,t,c_k) \frac{weight(d,t,c_l)}{\sqrt{\sum_{c \in B, t \in V} weight(d,t,c)^2}} \tag{1}$$

where V is the vocabulary of non-structural terms; B is the set of all XML contexts. $weight(q,t,c)$ and $weight(d,t,c)$ are the weights of term t in XML

context c in query q and document d, respectively. These weights were calculated by using the $idf_t * tf_{t,d}$ in the given context.

The CR function is calculated as shown in Eq. (2).

$$\mathrm{CR}(c_q, c_d) = \begin{cases} \frac{1+|c_q|}{1+|c_d|} & \text{if } c_q \text{ matches } c_d \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $|c_q|$ and $|c_d|$ are the number of nodes in the query path and the document path, respectively. We say that c_q matches c_d iff we can transform c_q into c_d by inserting additional nodes.

The implementation of the scoring function is given in Algorithm 1. This algorithm receives as input, the XML contexts, the query, the number of documents and a *normalizer* value ($\sqrt{\sum_{c \in B, t \in V} weight(d,t,c)^2}$), which is the same value for all documents.

Algorithm 1. Scoring of documents given a topic q

```
   Input: q, B, V, N : Number of documents, normalizer
   Output: score
 1 for n = 1 to N do
 2 |   score[n] = 0
 3 |   foreach ⟨c_q, t⟩ ∈ q do
 4 |   |   w_q = weight(q, t, c_q)
 5 |   |   foreach c ∈ B do
 6 |   |   |   if CR(c_q, c) > 0 then
 7 |   |   |   |   postings = GetPostings(c, t)
 8 |   |   |   |   foreach posting ∈ postings do
 9 |   |   |   |   |   x = CR(c_q, c) * w_q * PostingWeight(posting)
10 |   |   |   |   |   score[docID(posting)] += x
11 |   |   |   |   end
12 |   |   |   end
13 |   |   end
14 |   end
15 end
16 for n = 1 to N do
17 |   score[n] = score[n]/normalizer[n]
18 end
19 return score
```

3 Experimental Results

We have evaluated 45 topics with the corpus provided by the competition organizers (1,594,513 movies, 1,872,471 actors, 129,137 directors, 178,117 producers and, finally, 643,843 files categorized as others). For this purpose, we submitted one run which we was named: "p47-FCC-BUAP-R1".

Table 1 **a)** shows the Mean Average Precision (MAP) and Precision at 5 (P@5) obtained by the best overall submission, together with the score obtained by the run that we submitted (the median of the runs at the competition is also shown as a comparison value).

Table 1. Scores reported at the Ad-Hoc Track of INEX 2011

Run	Score
best overall score	0.3969
median	0.1973
p47-FCC-BUAP-R1	0.1479
worse overall score	0.0194

a) Mean Average Precision (MAP)

Run	Score
best overall score	0.5158
p47-FCC-BUAP-R1	0.4000
median	0.3236
worse overall score	0.0474

b) Precision at 5 (P@5)

In Figure 3 we may see the Precision-Recall graph for the best runs measured with MAP. The curve behaviour clearly shows that we have retrieved some relevant documents at the first positions of the ranking list, however, as the number of answers increases, we introduce a number of documents that were not considered at the gold standard. This analysis lead us to consider a better way of filtering the noisy documents in order to bring the rest of the relevant documents in a better position at the ranking list.

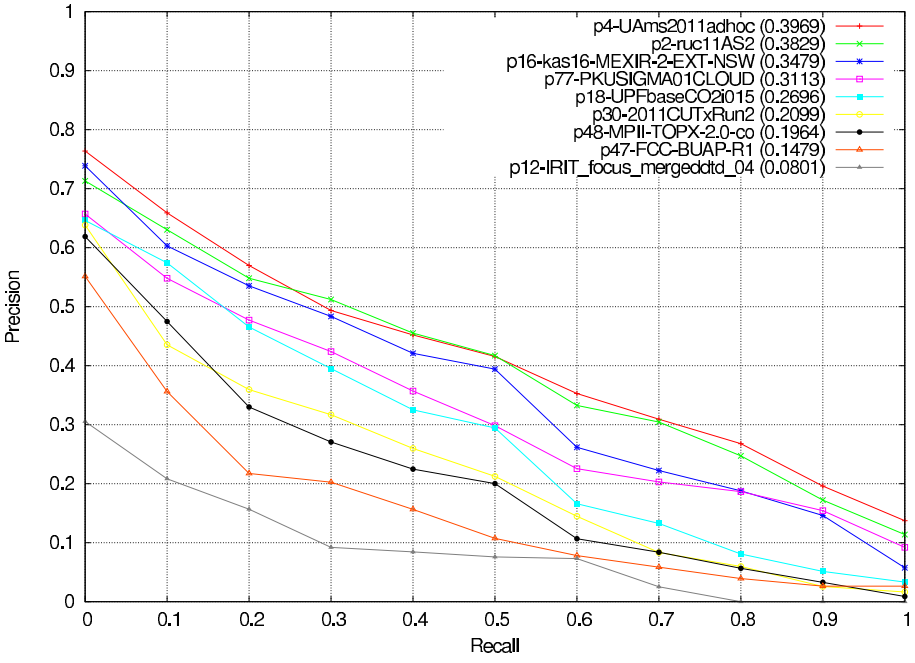


Fig. 3. Precision-Recall graph

In this edition of INEX, we have implemented a better mechanism for translating the topics, and we have introduced a better way of indexing the documents, by including the reference to the complete XML hierarchical structure, which has lead us to obtain better results.

4 Conclusions

In this paper we have presented details about the implementation of an information retrieval system which was used to evaluate the task of Ad-Hoc retrieval of XML documents, in particular, in the Data-Centric track of the Initiative for the Evaluation of XML retrieval (INEX 2011).

We presented an indexing method based on an inverted index with XML tags embedded. For each category (movies, actors, producers, directors and others), we constructed an independent inverted index. The dictionary of the index considered both, the category and the indexed term with its corresponding hierarchy to correctly identify the specific part of the XML file associated to the topic.

A recursive method for evaluating the topic was used considering only the "castitle" tag. The obtained results are mainly above median which encourages us to still participating in this competition forum, after analyzing the manner we may improve the document representation, document indexing and document retrieval.

References

1. Trotman, A., Wang, Q.: Overview of the INEX 2010 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 171–181. Springer, Heidelberg (2011)
2. Wang, Q., Ramírez, G., Marx, M., Theobald, M., Kamps, J.: Overview of the INEX 2011 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 118–137. Springer, Heidelberg (2012)
3. Wang, Q., Li, Q., Wang, S., Du, X.: Exploiting Semantic Tags in XML Retrieval. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 133–144. Springer, Heidelberg (2010)
4. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2009)

RUC @ INEX 2011 Data-Centric Track

Qiuyue Wang[1,2], Yantao Gan[1], and Yu Sun[1]

[1] School of Information, Renmin University of China
[2] Key Lab of Data Engineering and Knowledge Engineering, MOE,
Beijing 100872, P.R. China
qiuyuew@ruc.edu.cn, ganyantao19901018@163.com,
yusun.aldrich@gmail.com

Abstract. We report our experiment results on the INEX 2011 Data-Centric Track. We participated in both the ad hoc and faceted search tasks. On the ad hoc search task, we employ language modeling approaches to do structured object retrieval, trying to capture both the structure in data and structure in query and unify the structured and unstructured information retrieval in a general framework. However, our initial experimental results using INEX test bed show that the unstructured retrieval model performs better than structured retrieval models. On the faceted search task, we propose a simple user-simulation model to evaluate the effectiveness of a faceted search system's recommending facet-values. We implemented the evaluation system and conducted the evaluations for the track. We also tested basic redundancy and relevance based approaches for recommending facet-values. The results show that our basic approaches of recommending facet-values perform quite well.

1 Introduction

More and more data on the Web are structured, e.g. data in Deep Web and Semantic Web, while most end users prefer to search any data with a simple keyword query interface. There has been increasing interest structured data retrieval in recent years. INEX 2011 Data-Centric Track is one of the efforts to investigate retrieval techniques on highly structured XML data, where rich structure carries important semantic and relationship information about pieces of data. Basically, structural information can be exploited to improve ad hoc retrieval performance, and also can be used to help users navigate or explore a large set of results as in faceted search systems. We participated in both the ad hoc and faceted search tasks.

On the ad hoc search task, we study how to exploit structural information in data, and how to infer and exploit structural information implicit in queries to improve retrieval performance. We base our studies on language modeling approaches, and propose structured language models to automatically infer structural information from unstructured queries, and match structure in data and in query probabilistically. Compared with other structured language modeling approaches in information retrieval, our approach can capture both structure in data and in query and unify structured and unstructured data retrieval in a general framework. However, our experimental results on INEX show that the structured models are inferior to the unstructured ones.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 167–179, 2012.

On the faceted search task, to avoid expensive and non-repeatable user studies, we propose a cost-based metric and implement an evaluation system based on user simulations to evaluate all the submitted runs for the task. We also tested the basic redundancy and relevance based approaches for recommending facet-values. Among all the submitted runs based on the same reference result list, the basic redundancy based approach performs best.

2 Ad Hoc Search Task

Language modeling approach has a solid statistical foundation, and can be easily adapted to model various kinds of complex and special retrieval problems, such as structured document retrieval. In particular, mixture models [1] and hierarchical language models [2][3][4] were proposed to be applied in XML retrieval. On the ad hoc search task in INEX 2011 data-centric track, we employ the language modeling approach to do structured object retrieval as the IMDB data collection can be viewed as a set of structured objects, i.e. movies and persons. With the rich structural information in data, we intend to investigate how to capture the structural information in data as well as that in query in language models to retrieve more accurate results for an ad hoc information need.

In this section, we discuss different ways of adapting language modeling approach to structured object retrieval, and evaluate them on the IMDB data collection.

2.1 Unstructured Data, Unstructured Query

The basic idea of language modeling approach in IR is to estimate a language model for each document (θ_D) and the query (θ_Q), and then rank the document in one of the two ways: by estimating the probability of generating the query string with the document language model, i.e. $P(Q|\theta_D)$, as in Equation 1, or by computing the Kullback-Leibler divergence of the query language model from the document language model, i.e. $D(\theta_Q \,/\!/\, \theta_D)$, as in Equation 2.

$$P(Q \mid \theta_D) = \sum_{w \in Q} P(w \mid \theta_D) \tag{1}$$

$$-D(\theta_Q \,||\, \theta_D) = -\sum_{w \in V} P(w \mid \theta_Q) \log \frac{P(w \mid \theta_Q)}{P(w \mid \theta_D)} \tag{2}$$

On the surface, the KL-divergence model appears to be quite different from the query likelihood method. However, it turns out that the KL-divergence model covers the query likelihood method as a special case when we use the empirical distribution to estimate the query language model, i.e. maximum-likelihood estimate.

In IMDB data collection, each document is a structured object, i.e. movie or person. Our first retrieval strategy is to ignore the structural information in each object, estimate a language model for each object based on its free-text content, and rank them using query likelihood.

2.2 Structured Data, Unstructured Query

When taking into account the structural information in data, the common way is to view each object as consisting of multiple fields and represent its language model as a combination of different language models estimated from its different fields [5][6], as shown in Equation 3.

$$P(w \mid \theta_D) = \sum_{i=1}^{k} \lambda_i P(w \mid \theta_{D_i}) \tag{3}$$

Here we assume that object D consists of k fields. $P(w \mid \theta_{D_i})$ is the language model estimated from its ith field, which is normally smoothed by interpolating with the collection's ith field's language model.

The key problem in Equation 3 is to determine the combination weights, i.e. the value of λ_i. In the generative model, λ_i is set to be $P(\theta_{D_i} \mid \theta_D)$, the probability of object D generating the ith field, which could be uniform, i.e. $1/k$, or proportional to the frequency or length of the ith field in D as suggested in [3]. Alternatively, λ_i can be set to reflect the importance of the ith field to D, which could be learned or tuned to the task. If no training data exist, we suggest a new heuristic using the normalized average IDF values of all the terms in a field to determine the importance weight of this field. The intuition behind this is that the more discriminative the content of a field is, the more important this field is. For example, the "name" field of a person object is more important than its "birth_date" field. In [7], a concept of mapping probability is proposed to be used as λ_i to combine different fields' language models in Equation 3. The mapping probability of a given query term to a related field is the probability that this term maps to the field. Thus, the importance of a field depends on how likely query terms occur in this field.

Table 1 summarizes the various ways of determining λ_i, where N_i is the number of instances of the ith field in D and N is the number of instances of all fields in D. C_i denotes the set of distinct terms occurring in the ith field in the collection.

Table 1. Different concepts and calculations for determining λ_i

Concept	Calculation
generating probability of the ith field: $P(\theta_{D_i} \mid \theta_D)$	AVG: $1/k$ FREQ: N_i / N LENGTH: $length(D_i) / length(D)$
importance or prior probability of the ith field: $P(\theta_i)$	AVG: $1/k$ IDF: $\frac{(\sum_{w \in C_i} idf_w)/\mid C_i \mid}{\sum_{j=1}^{k}[(\sum_{w \in C_j} idf_w)/\mid C_j \mid]}$
posterior or mapping probability of the ith field given a query term w: $P(\theta_i \mid w)$	MAPPR: $\frac{P(w \mid \theta_i)P(\theta_i)}{\sum_{j=1}^{k}(P(w \mid \theta_j)P(\theta_j)}$

We implemented all the 5 strategies of calculating λ_i shown in Table 1, i.e. AVG, FREQ, LENGTH, IDF, MAPPR, and evaluate them on the IMDB data collection.

2.3 Structured Data, Structured Query

In the previous two sections, we treat a query as a bag of words with no structure. However, when search is against structured data, it is beneficial for the search system to discover the latent structural information in queries and exploit it to improve retrieval accuracy. For example, if a user wants to find a very recent romantic movie directed by Yimou Zhang, he/she may issue the keyword query "Yimou Zhang 2010 romance" to the system. If the retrieval system knows that the return element type should be *movie*, "Yimou Zhang" should be the *director* of the movie, "2010" and "romance" should appear in the *release-date* and *genre* fields respectively, the retrieval precision can be greatly improved and the correct movie "The Love of the Hawthorn Tree" could be returned.

Since asking users to provide such information for each query would be an excessive burden on them and most users are only willing to express the simplest forms of queries, a lot of research work has been done to automatically infer structured queries from the keyword queries submitted by users [8][9][10]. This process is typically divided into three steps: firstly, generating all possible structured queries from the input unstructured query by incorporating the knowledge of schemas, data statistics, heuristics, user/pseudo relevance feedback and etc.; secondly, ranking the structured queries according to their likelihood of matching user's intent; thirdly, selecting the top-k structured queries and evaluating them. However, if the inferred structured queries are not intended by the user, we cannot expect to get the right result. Another line of research work done in this direction is to infer some structural hints, not necessarily hard structural constraints as in structured queries, from the input keyword query, such as the mapping probability proposed in [7].

In this paper, we use structured language models to capture the structural hints in query. With a keyword query and data collection, we first infer a structured language model for the query as shown in Equation 4, and then we match query's structured language model with each object's structured language model by computing their KL divergence and rank the objects accordingly. With Equation 3 and 4, we can use Equation 2 to compute the KL divergence between two language models. But to address the structural matching, we use Equation 5 to compute the structural KL divergence, that is, to compute the overall KL divergence between the language models on all fields.

$$P(w \mid \theta_Q) = \sum_{i=1}^{k} \pi_i P(w \mid \theta_{Q_i}) \tag{4}$$

$$-D(\theta_Q \,||\, \theta_D) = -\sum_{i=1}^{k} D(\theta_{Q_i} \,||\, \theta_{D_i}) = -\sum_{i=1}^{k} \sum_{w \in V} P(w \mid \theta_{Q_i}) \log \frac{P(w \mid \theta_{Q_i})}{P(w \mid \theta_{D_i})} \tag{5}$$

Now, the key issue is how to estimate a structured language model for a keyword query. With no other information about queries, we use the pseudo relevance feedback approach similar to that used in [11] to estimate the structured language model for a query. Note that in [11] documents are unstructured while in our case documents are structured, so the details of our estimator are different from [11]'s. Since query and document models are modeling the same types of objects, we assume that the field weights in the query model, π_i, are the same as the field weights in the document model, λ_i, and both are equal to $P(\theta_i)$, the field importance or prior probabilities. Given a set of feedback documents F, we assume that the content of each document field are sampled from a model mixing the query's and collection's corresponding field language models, which model the relevant and non-relevant information respectively as shown in Equation 6, where α_D is a document-specific mixing weight parameter.

$$P(w \mid \theta_{D_i}) = \alpha_D P(w \mid \theta_{Q_i}) + (1-\alpha_D) P(w \mid \theta_{C_i}) \tag{6}$$

All the parameters to be estimated can thus be represented as $\Lambda = \{\{P(w \mid \theta_{Q_i})\}_{i=1,\ldots,k}, \{\alpha_D\}_{D \in F}\}$. We use the Maximum A Posterior (MAP) estimator as shown in Equation 7 to estimate the parameters, where $P(F \mid \Lambda)$ is the likelihood of the feedback documents given in Equation 8 and $P(\Lambda)$ is a conjugate prior on all the field language models of the query as shown in Equation 9. In Equation 9, $P(w \mid \overline{\theta_{Q_i}})$ is the prior field language model of the query and parameter μ specifies our confidence about the prior.

$$\hat{\Lambda} = \arg\max_{\Lambda} P(F \mid \Lambda) P(\Lambda) \tag{7}$$

$$P(F \mid \Lambda) = \prod_{D \in F} \prod_{i=1}^{k} \prod_{w \in V} P(w \mid \theta_{D_i})^{c(w, D_i)} \tag{8}$$

$$P(\Lambda) \propto \prod_{i=1}^{k} \prod_{w \in V} P(w \mid \theta_{Q_i})^{\mu P(w \mid \overline{\theta}_{Q_i})} \tag{9}$$

The MAP estimate can be found using the EM algorithm. We introduce a hidden variable for the identity of each word in each field of a document, $z_{D_i,w}$, and $P(z_{D_i,w} = Q_i)$ is the probability that the word w in the ith field of document D is generated by the query's ith field's language model and $P(z_{D_i,w} = C_i)$ is the probability that the word w in the ith field of document D is generated by the collection's ith field's language model. We have $P(z_{D_i,w} = Q_i) + P(z_{D_i,w} = C_i) = 1$. The updating formulas are as follows:

E-step:

$$P^{(n+1)}(z_{D_i,w} = Q_i) = \frac{\alpha_D^{(n)} P^{(n)}(w \mid \theta_{Q_i})}{\alpha_D^{(n)} P^{(n)}(w \mid \theta_{Q_i}) + (1-\alpha_D^{(n)}) P(w \mid \theta_{C_i})} \tag{10}$$

M-step:

$$\alpha_D^{(n+1)} = \frac{\sum_{w \in V} \sum_{i=1}^{k} c(w, D_i) P^{(n)}(z_{D_i,w} = Q_i)}{\sum_{w \in V} \sum_{i=1}^{k} c(w, D_i)} \tag{11}$$

$$P^{(n+1)}(w \mid \theta_{Q_i}) = \frac{\sum_{D \in F} c(w, D_i) P^{(n+1)}(z_{D_i,w} = Q_i) + \mu P(w \mid \bar{\theta}_{Q_i})}{\sum_{w \in V} \sum_{D \in F} c(w, D_i) P^{(n+1)}(z_{D_i,w} = Q_i) + \mu} \tag{12}$$

As the input query is unstructured, we set its prior field language model as $P(w \mid \bar{\theta}_{Q_i}) = \frac{c(w,Q)}{|Q|}$, and the collection language model is $P(w \mid \theta_{C_i}) = \frac{c(w,C_i)}{|C_i|}$. To avoid tuning the prior confidence value μ manually, we adopt the strategy in [12]. Thus, we start with a sufficiently large μ and discount it a little bit at each iteration of the EM algorithm until the stopping condition is reached. In our experiments, we set the initial prior confidence μ=5000 and the discounting factor δ=0.9. The EM iterations start with $\alpha_D^{(0)} = 0.1$, $P^{(0)}(w \mid \theta_{Q_i}) = \frac{c(w,Q)}{|Q|}$.

2.4 Experimental Results

We implemented the structured language modeling approaches discussed in the previous two sections inside the source code of Lemur 4.11. In this section, we will report our experiment results on the ad hoc search task of INEX 2011 data-centric track.

We indexed all the leaf fields in the IMDB XML data collection using the Indri. Each field is uniquely identified by its full XML path. But we take only the last tag name of the path as the field's name. Even though some different XML paths have identical last tag names, they are of same semantics in the IMDB collection, for example, "name" wherever it appears means a person's name and "title" means a movie's title no matter in a movie or a person object. In total, there are 49 distinct field names for the IMDB data collection, 31 fields for movie objects and 23 fields for person objects. We built the index using the Krovetz stemmer and a short stop word list that contains only eleven common words: {*a, an, and, as, by, in, of, or, that, the, to*}.

In the retrieval models discussed above, the document and document field language models, i.e. θ_D and θ_{D_i}, are estimated using maximum likelihood estimate, which are then smoothed using a Dirichlet prior with the collection or collection field

language models. We set the Dirichlet prior for smoothing document language model in the unstructured language modeling approach to be 1000, which is optimal tuned on INEX 2010 data-centric track topics. In our submitted runs, we set the Dirichlet prior for smoothing document field language models in the structured language modeling approaches to be 1000 too. The results showed that the performance of structured language modeling approaches were much worse than that of unstructured approaches. This was an unfair comparison however since the optimal Dirichlet prior for smoothing document field language models may not be the same as that for document language model. Actually the lengths of document fields are typically short, much shorter than the average document length. Many fields contain only a couple of terms, e.g. director's name, release date of a movie. So we rerun our experiments setting the Dirichlet prior for document field language models to be much smaller than 1000, e.g. 10, 30, 50 and 100. The performance of structured language models is much improved. Since the lengths of different fields vary a lot, for example the plot or biography fields usually contain large pieces of text, it is better to set different smoothing Dirichlet priors for different fields. How to appropriately smooth the document field language models in the structured language modeling approaches is an interesting question that we would like to explore in the future.

Table 2. Effectiveness of unstructured and structured language modeling approaches

ID	parameters of runs	MAP	1/rank	P@5	P@10	P@20	P@30
1-1	unstructured, dir=1000, fb=0 (p2-ruc11AS2)	0.3829	0.6441	0.4474	0.4132	0.3842	0.3684
2-1	avg, dir=10, fb=0	0.307	**0.7383**	**0.5368**	**0.4868**	**0.4596**	**0.4368**
2-2	idf, dir=100, fb=0	0.2864	0.6738	0.4789	0.4263	0.3737	0.3421
2-3	mapping_prob, dir=10, fb=0	0.2295	0.6029	0.3895	0.3789	0.3461	0.3254
2-4	length, dir=100, fb=0	0.1996	0.5445	0.3737	0.3316	0.2763	0.2439
2-5	freq, dir=10, fb=0	0.1547	0.4698	0.3474	0.3	0.2697	0.2544
3-1	avg, dir=1000, fb=10	0.1993	0.5542	0.3895	0.3447	0.2974	0.2614
3-2	idf, dir=1000, fb=10	0.2025	0.5461	0.4	0.35	0.3013	0.2684
3-3	mapping_prob, dir=1000, fb=10	0.2027	0.5326	0.3842	0.3316	0.2882	0.2553
3-4	length, dir=1000, fb=10	0.1947	0.553	0.3842	0.3474	0.2882	0.2579
3-5	freq, dir=1000, fb=10	0.1876	0.5512	0.3789	0.3342	0.2829	0.2526
	Best in INEX 2011	**0.3969**	0.6999	-	0.4421	0.4171	0.3851

The results of different retrieval strategies are shown in Table 2. The first run is our baseline, which uses the standard KL-divergence method to rank the documents with no structure information taken into account as described in Section 2.1, with no relevance feedback (*fb=0*) and the Dirichlet prior for smoothing document language models is set 1000 (*dir=1000*). It is also the best official run (run ID is *p2-ruc11AS2*) that we submitted to the INEX 2011 data-centric track. Run 2-1 to 2-5 use the

structured language models and KL divergence as discussed in Section 2.2 to rank structured documents, where *avg*, *idf*, *mapping_prob*, *length*, and *freq* are the five strategies of computing the field weights. For each strategy of computing field weights, we set the Dirichlet prior 10, 30, 50, and 100, and choose the best one in terms of *P@5* from the four results and put it in Table 2. For example, the best Dirichlet prior for *avg* strategy in terms of *P@5* is 10, so we put the result of "*avg, dir=10, fb=0*" in the table. But for *idf* strategy, the best Dirichlet prior is 100. Run 3-1 to 3-5 employ the pseudo relevance feedback to estimate a structured language model for each unstructured query and then rank the structured documents using structured KL divergence as discussed in Section 2.3. *fb=10* means that we take the top 10 documents returned by the retrieval method in Section 2.2 to estimate the structured query language model.

From Table 2, we can observe that structured language modeling approaches can greatly improve the early precision, but they also hurt recall and thus the average precision. The early precision of run 2-1 is much better than our best official run and also beats the best early precisions from all the submitted runs to INEX 2011 data-centric track. Structured relevance feedback approaches unexpectedly perform worse than the unstructured approach and structured approaches without feedback. It demands further investigation.

3 Faceted Search Task

For the faceted search task, as it is the first year, our work focused on evaluating effectiveness of faceted search systems based on user simulation approach, and testing a simple redundancy based approach for recommending facet-values.

3.1 Faceted Search Evaluation

Faceted search systems are typically evaluated through user studies, which are expensive and not repeatable. For INEX 2011 data-centric track, we proposed to employ user simulation technology to evaluate faceted search systems. The measurement is the interaction cost defined as the number of results, facets or facet-values examined by the user before he/she encounters the first relevant result, which is similar to the Reciprocal Rank (RR) metric in traditional IR. Readers are referred to Section 5 in the track overview paper [13] for the detailed discussion about the user model and cost model assumed in our evaluation system.

We implemented the user simulation system in Java, which can evaluate both dynamic faceted search modules and static facet-value hierarchies. The only difference is that there is no EXPAND option for static facet-value hierarchies. The evaluation results of all the submitted runs to the faceted search task are given in Section 6.2 of [13]. We can see that the cost based metric and user simulation based evaluation system are feasible in comparing different facet-value recommending systems. We also tested the robustness of the evaluation metric in terms of different

Table 3. Evaluation results of all static runs in terms of NGs and ANG using ***Stochastic*** model

run / NG	p4-UAms2011indri-c-cnt	p4-UAms2011indri-cNO-scr2	p4-UAms2011lucene-cNO-lth	p48-MPII-TOPX-2.0-facet-entropy (TopX)	p48-MPII-TOPX-2.0-facet-entropy (Lucene)	p4-2011IIpsFtScore	p4-2011IIpsNumdoc	p18-2011UPFfixG7DAnh	p18-2011UPFfixGDAh	p18-2011UPFfixGDAh2	p2-2011Simple1Run1
201	0.47	0	-	0	-	-	-	-	-	-	-
202	0	0	0	0	0	0	0	0	0.11	0	0
203	0	0.46	0	-	0	0.79	0.79	0	0	0	0.83
204	0.68	0.87	0.94	-	0	0	0.91	0	0	0.88	0.89
205	0	0	0.75	0	0.74	0.68	0.68	0.90	0.90	0.90	0.75
207	0	0.74	0	0	0	0	0	0	0	0	0.92
208	0	0	0	0	0	0	0	0	0	0	0
209	0	0	0	0	0	0	0	0	0	0	0
210	0.79	0.68	-	0	-	-	-	-	-	-	-
211	0	0	0.51	0	0	0	0	0.31	0.6	0.22	0.31
212	0.89	0.89	-	0	-	-	-	-	-	-	-
213	0.90	0.90	-	-	-	-	-	-	-	-	-
214	0	0	0.45	0	0	0	0	0	0.55	0.25	0
ANG	0.29	**0.35**	0.20	0	0.06	0.11	0.18	0.09	0.16	0.17	**0.28**

user models. In [13], we assume that the user model is the ***First-Match*** model, i.e. at each step user selects the first relevant facet-value among the list of recommended ones. In Table 3 and 4, we show the evaluation results for the ***Stochastic*** and ***Myopic*** user models respectively. Stochastic users randomly select a relevant facet-value from the recommended list, while myopic users select the facet-value that is contained in the least number of results. From the tables, we can see that the evaluation results for different user models are consistent. As the numbers of runs and topics are too small, we can hardly draw any conclusions, and we cannot test the robustness of the evaluation metric in terms of the number of topics and so on. In principle, such RR-like metrics are less robust than NDCG-like metrics.

3.2 Recommending Facet-Values

Given a broad or exploratory query, a search system typically returns a long list of results. A faceted search system can help users navigate through the result list to identify item of interest by recommending facet-value conditions for refining the query at each navigation step. Essentially, the task is to construct a hierarchy of facet-values that can cover all the relevant results in the result list. As all the results in the

Table 4. Evaluation results of all static runs in terms of NGs and ANG using ***Myopic*** model

run / NG	p4-UAms2011indri-c-cnt	p4-UAms2011indri-cNO-scr2	p4-UAms2011lucene-cNO-lth	p48-MPII-TOPX-2.0-facet-entropy (TopX)	p48-MPII-TOPX-2.0-facet-entropy (Lucene)	p4-2011IIpsFtScore	p4-2011IIpsNumdoc	p18-2011UPFfixG7Danh	p18-2011UPFfixGDAh	p18-2011UPFfixGDAh2	p2-2011Simple1Run1
201	0.47	0	-	0	-	-	-	-	-	-	-
202	0	0	0	0	0	0	0	0	0.05	0	0
203	0	0.59	0	-	0	0.79	0.79	0	0	0	0.83
204	0.77	0.80	0.94	-	0	0	0.94	0	0	0.90	0.89
205	0	0	0.92	0	0.74	0.88	0.75	0.90	0.94	0.90	0.81
207	0	0.74	0	0	0	0	0	0	0	0	0.90
208	0	0	0	0	0	0	0	0	0	0	0
209	0	0	0	0	0	0	0	0	0	0	0
210	0.79	0.76	-	0	-	-	-	-	-	-	-
211	0.66	0.66	0.51	0	0	0	0	0.31	0.6	0.22	0.22
212	0.90	0.89	-	0	-	-	-	-	-	-	-
213	0.90	0.92	-	-	-	-	-	-	-	-	-
214	0	0	0.5	0	0	0	0	0.14	0.43	0.25	0
ANG	0.34	**0.41**	0.22	0	0.06	0.13	0.19	0.10	0.16	0.17	**0.28**

result list could be of users' interest and it is assumed that the faceted search interface can present no more than 20 facet-value conditions at a time and each result page contain no more than 10 results, the task can be reformulated more precisely as to construct a hierarchy of facet-values that covers all the results in the result list with each leaf node covering no more than 10 results and the fan-out of each non-leaf node no more than 20.

User's interaction cost with a hierarchy of facet-values, i.e. the number of results, facets or facet-values that the user examined, is proportional to the size of the hierarchy. The larger the hierarchy is, the more facet-values a user could potentially examine. So our main goal of the task is to construct a minimum hierarchy satisfying the conditions. We then divided the goal into two sub-goals:

1. Cover the current result list with a minimum number of facet-values at each branching node of the hierarchy.
2. Minimize the average path length of the hierarchy.

That is, we intend to construct a hierarchy that is neither too wide nor too deep. However, these two sub-goals compete with each other. The first sub-goal closely resembles the well-known NP-hard set-cover problem, which is commonly solved by a greedy algorithm [14]. The greedy algorithm adds at each step the facet-value that covers the maximum number of uncovered results until the whole result set is

covered. But this could probably result in very long paths in the hierarchy. For instance, at the node of facet-value fv_1, current result list contains 100 results, and a facet-value fv_2 covers 99 results and will be selected by the greedy algorithm, then next at the node of fv_2, the greedy algorithm may select a facet-value fv_3 that covers 98 results, and so on. A long path, $fv_1(100) \rightarrow fv_2(99) \rightarrow fv_3(98) \rightarrow \ldots$, is thus formed in the hierarchy. This could cause a high interaction cost. Also in practice, users may find such a facet-value condition useless since they can hardly recognize any change in the result list once they choose it to refine the query. We adopt a simple heuristic to avoid such a problem. We set a threshold for the number of results that a facet-value reduces the current result list. In our experiments, the threshold is set 20. That is, if a facet-value reduces the result list by less than 20 results, it will not be selected to be recommended. The whole algorithm for recommending a list of facet-values given a list of results is shown in Algorithm 1.

Algorithm 1. Recommending Facet-Values

```
  Input: current result list Rc
  Output: a list of recommended facet-values L
  begin
1   L := Φ;
2   R := Rc;
3   FV := set of all possible facet-values in R;
4   fv := argmax_{fv∈FV&&(|R|-|R_fv|)≥20}{|R_fv|}; //R_fv: set of
results in R that are covered by fv
5   append fv to L;
6   R := R-R_fv;
7   if (R≠Φ && |L|<20) then
8     goto 3;
9   endif;
10  return L;
  end.
```

In Algorithm 1, line 4 is to select the facet-value that covers the most unseen results while reducing the result list by at least 20 results. Once a facet-value is selected into the recommended list, all the results it covers would be removed from the result list. These are presented in line 5 and 6. The process continues until all the results in the given result list are covered or the number of recommended facet-values reaches the upper bound 20, which is set by the track.

In addition, we extend Algorithm 1 by taking into account the relevance scores of the unseen results covered by a facet-value. That is, at each step of the greedy algorithm, i.e. at line 4 in Algorithm 1, the facet-value with the highest relevance score instead of the most frequency on the unseen result set is selected to be added into the recommended list. The relevance score of a facet-value is defined as the sum of the relevance scores of all the unseen results covered by the facet-value.

3.3 Results

Two static facet-value hierarchies are generated using Algorithm 1 and its extension respectively, both based on the reference result list provided by the track. Table 5 presents the evaluation results for them in terms of NGs and ANG under *first-match* user model. We call Algorithm 1 "most-frequent" strategy, and its extension "most-relevant" strategy. Note that the run generated by "most-frequent" strategy is our submitted run, whose ID is p2-2011Simple1Run1. In the table, we ignore the topics that the reference result list contains no relevant result, which has no effect on the evaluation results.

Table 5. Evaluation results of Algorithm 1 and its extension in terms of NGs and ANG

algorithms	202	203	204	205	207	208	209	211	214	ANG
most-frequent	0.21	0.86	0.91	0.81	0.94	0	0	0.6	0	**0.33**
most-relevant	0.21	0	0.9	0.73	0.94	0	0	0.64	0.64	0.31

The results of two runs show no significant difference, even though the ANG of most-frequent strategy is a little greater than that of most-relevant one. In general, most-frequent strategy performs better on topics whose relevant results appear late in the result list, e.g. topic 203, while most-relevant strategy is better on topics whose relevant results appear early in the result list, e.g. topic 214. Compared with all other submitted runs to INEX generated on the reference result list, our static hierarchy is the most effective one in terms of ANG. The best performed run is p4-UAms2011indri-cNO-scr2 submitted by University of Amsterdam, but based on the result list generated by Indri, which contains relevant results for all topics while the reference result list contains no relevant results for four topics. Thus the results are not comparable. So in INEX 2012, all the submitted runs to the faceted search task should be required to be on the same result list.

4 Conclusions and Future Work

In this paper, we presented our work on INEX 2011 Data-Centric Track. We participated in both the ad hoc and faceted search tasks. We employ structured language models to capture structure both in data and in query and utilize them in ad hoc search on data-centric XML. But the experimental results on INEX are not promising for the structured language models. We are going to investigate this further on to gain more and deeper insights into structured data retrieval models and techniques.

For the faceted search task, we proposed to use a cost-based metric and user-simulation model to evaluate all the runs. The results show that it is a feasible way for faceted search evaluation. But given the very small number of topics and runs, the robustness of the metric could not be established. We intend to test it on more faceted search systems with more topics, and compare its results with that of user studies, to see how real and how robust it is. For the first year of the task, we simply tested two

basic strategies for recommending facet-values, redundancy and relevance based approaches. As our future work, we are also interested in studying more sophisticated approaches for recommending facet-values.

References

1. Hiemstra, D.: Statistical Language Models for Intelligent XML Retrieval. In: Blanken, H.M., Grabs, T., Schek, H.-J., Schenkel, R., Weikum, G. (eds.) Intelligent Search on XML Data. LNCS, vol. 2818, pp. 107–118. Springer, Heidelberg (2003)
2. Ogilvie, P., Callan, J.: Language Models and Structured Document Retrieval. In: INEX 2003 (2003)
3. Ogilvie, P., Callan, J.: Hierarchical Language Models for XML Component Retrieval. In: Fuhr, N., Lalmas, M., Malik, S., Szlávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 224–237. Springer, Heidelberg (2005)
4. Ogilvie, P., Callan, J.: Parameter Estimation for a Simple Hierarchical Generative Model for XML Retrieval. In: Fuhr, N., Lalmas, M., Malik, S., Kazai, G. (eds.) INEX 2005. LNCS, vol. 3977, pp. 211–224. Springer, Heidelberg (2006)
5. Ogilvie, P., Callan, J.: Combining Document Representations for Known-Item Search. In: SIGIR 2003 (2003)
6. Nie, Z., Ma, Y., Shi, S., Wen, J., Ma, W.: Web Object Retrieval. In: WWW 2007 (2007)
7. Kim, J., Xue, X., Croft, W.B.: A Probabilistic Retrieval Model for Semistructured Data. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 228–239. Springer, Heidelberg (2009)
8. Petkova, D., Croft, W.B., Diao, Y.: Refining Keyword Queries for XML Retrieval by Combining Content and Structure. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 662–669. Springer, Heidelberg (2009)
9. Li, X., Wang, Y., Acero, A.: Extracting Structured Information from User Queries with Semi-Supervised Conditional Random Fields. In: SIGIR 2009 (2009)
10. Sarkas, N., Paparizos, S., Tsaparas, P.: Structured Annotations of Web Queries. In: SIGMOD 2010 (2010)
11. Zhai, C., Lu, X., Ling, X., He, X., et al.: UIUC/MUSC at TREC 2005 Genomics Track. In: TREC 2005 (2005)
12. Tao, T., Zhai, C.: Regularized Estimation of Mixture Models for Robust Pseudo-Relevance Feedback. In: SIGIR 2006 (2006)
13. Wang, Q., Ramírez, G., Marx, M., Theobald, M., Kamps, J.: Overview of the INEX 2011 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 118–137. Springer, Heidelberg (2012)
14. Johnson, D.S.: Approximation algorithms for combinatorial problems. Journal of Computer and System Sciences 9, 256–278 (1974)

MEXIR at INEX-2011

Tanakorn Wichaiwong and Chuleerat Jaruskulchai

Department of Computer Science,
Faculty of Science, Kasetsart University,
Bangkok, Thailand
{g5184041,fscichj}@ku.ac.th

Abstract. This is the second year of Kasetsart University's participation in INEX. We participated in two tracks: Snippet retrieval and Data Centric. This year, we introduced an XML information retrieval system that uses MySQL and Sphinx which we call the More Efficient XML Information Retrieval (MEXIR). In our system, XML documents are stored into one table that has a fixed relational schema. The schema is independent of the logical structure of XML documents. Furthermore, we present a structure weighting function which optimizes the performance of MEXIR.

Keywords: XML Retrieval, Data Centric, INEX, Information Retrieval.

1 Introduction

Due to the grows in electronic information available there has been an increase in the size of the collection used [1]. Large collections are commonplace now. Since, the Extensible Markup Language (XML)[1] documents have additional information; document representation of these might be add up meta-data to describe data in context respect to XML language design [2,3,5].

According to previous year, we are addressing on Content Only (CO) or purely keywords search. In this year, we move forward to study the Content and Structure (CAS) for the Data Centric track of INEX. In this year, we also participated in Snippet retrieval track. Furthermore, we presented the structure weight function to improve the performance of our information system.

This paper is organized as follows; Section 2 reviews data model and notions. Section 3 explains the implementation of our system overview and new structure weight algorithm. Section 4 show the experiment, Section 5 explains the result and discussion, conclusions and further work are drawn in Section 6.

2 Data Model and Notions

In this section, we provide some historical perspectives on areas of XML research that have influenced to this article as follows.

[1] http://www.w3.org/TR/xml11/

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 180–187, 2012.

2.1 XML Data Models

The basic XML data model is a labeled, ordered tree. Fig. 1 shows the data tree of an XML document based on the node-labeled model respect to IMDB collection[2]. There are basically three types of nodes in a data tree as follows.

Element nodes correspond to tags in XML documents, for example, the "imdb", "title", "locations", "actors" and "movie" nodes.

Attribute nodes correspond to attributes associated with tags in XML documents. In contrast to element nodes, attribute nodes are not nested, not repeatable, and unordered.

Leaf nodes (i.e., text nodes) correspond to the data values in XML documents, for example, the "New York", "Fox Hills", "Big Love", "Matt Rose" and "Bill Paxton" nodes.

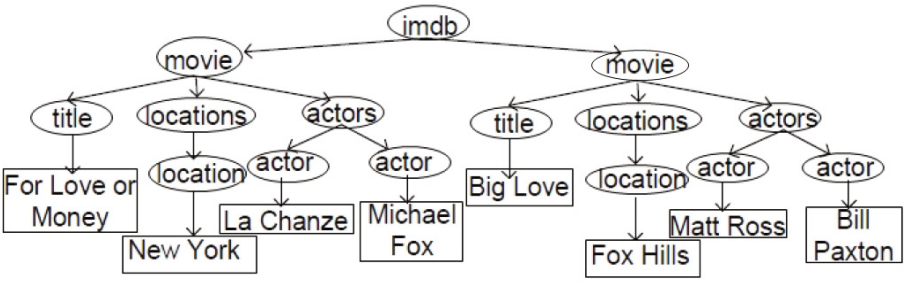


Fig. 1. The Example of IMDB in XML Element Tree

2.2 XML Query Languages

Querying in structure documents must be respect to content and structure. INEX identified two types of topics [10,8,9], they are Content Only (CO) and Content and Structure (CAS) as follows;

Content Only, these topics are requested that ignore the document structure, the traditional topics used in Information Retrieval (IR) test collections. However, they pose a challenge to XML retrieval in that the retrieval results to returning document components, i.e. XML elements instead of whole documents in response to a user query. Topics can be elements of various complexities, i.e. at different levels of the XML document's structure.

Content and Structure, these topics are requested that contain conditions referring both to content and structure. These conditions may refer to the content of specific elements, i.e. the elements to be returned must contain a section about a particular topic, or may specify the type of the requested answer elements.

[2] http://www.imdb.com

2.3 Snippet Retrieval Track

This track is aiming to investigated how to generate informative snippets for search results [7]. Such snippets should provide sufficient information to allow the user to judge the relevance of each document, without the needing to view the document itself.

2.4 Data Centric Track

This track is aiming to investigated retrieval over a strongly structured collection of documents based on IMDB [11]. The ad-hoc search has informational requests to be answered by the entities in IMDB collocations. The faceted search is aiming to asks for a restricted list of facets and facet-values that will optimally guide the searcher toward relevant information.

3 Methods

3.1 An Implementation of XML Retrieval System

The More Efficient XML Information Retrieval (MEXIR) [15] is based on the leaf-node indexing scheme that uses a relational DBMS as a storage back-end. We used the schema setup using MySQL[3] and the full-text engine Sphinx [4] with the MySQL dumps function.

For the initial step, we consider a simplified XML data model, but we disregard meta-data such as, comments, links and attributes. The main components of the MEXIR retrieval system are followed:

1. When new documents are entered to the system, the Absolute Document XPath Indexing (ADXPI) [14] indexer parses and analyzes the name of an element and its position to build the inverted lists for each index in this system.
2. The compression for ADXPI compressor [13] analyzes the frequency of each element and its position to build the mapping of dictionary base, which is stored the compressed data into MySQL database.
3. The AutoMix [12] analyzes the tree position for each element to separate content to build the Selected Weight (SW) and Leaf Node indices, which is stored in the MySQL database.
4. The Sphinx search engine is used to build all contents in this system both indices. For the Selected Wight base on Term Frequency and the Leaf Node base on BM25 [4].
5. The Score Sharing function [16] is used to assign parent scores by sharing scores from leaf nodes to their parents using a top-down approach.
6. The Double Scoring function [12] is used to adjust the Leaf Node scores based on linear recombination.

[3] http://dev.mysql.com/
[4] http://www.sphinxsearch.com/

3.2 Content Weight Function

We used the BM25 in Sphinx's formula to score the document leaf nodes according to query terms contained in content conditions as follows:

$$LeafScore(e, Q) = \sum_{e \in q} W_t * \frac{(k_1 + 1) * tf_e}{k_1 * \langle (1 - b) + b * \frac{len(e)}{avel}) + tf_e \rangle} \quad (1)$$

Note that;

$LeafScore(e, Q)$ measures the relevance of element e in leaf-node indices to query Q.

tf_e is the frequency of term t occurring in element e.

$len(e)$ is the length of element e.

$avel$ is the average length of elements in the entire collection.

k_1 and b are used to balance the weight of term frequency and element length.

And then, we compute the inverse element frequency as follows:

$$W_t = \frac{\log\langle \frac{N - e_t + 1}{e_t} \rangle}{\log(N + 1)} \quad (2)$$

Note that;

W_f is the inverse element frequency weight of term t.

N is the total number of an element in the entire collection.

e_t is the total element of a term t occur.

3.3 Structure Weight Function

Our structural scoring model essentially counts the number of navigational (i.e., element-name) query conditions that are satisfied by a result candidate and thus connect the content conditions matched for the user queries. It assigns c for every directional condition that is matched a part of an absolute path. When a matching the organizational constraints against the document tree, we calculate structural scoring using 2^c and recomputed the leaf element score as following:

$$LeafScore(Node) \leftarrow LeafScore(Node) * \langle 2^c \rangle \quad (3)$$

Note that;

c is the frequency of navigational condition that is matched a part of an absolute path

4 Experiment Setup

In this section, we present and discuss the results based on the INEX collection. This experiment was performed on Intel Pentium i5 4 * 2.79 GHz with 6 GB of memory, Microsoft Windows 7 Ultimate 64-bit Operating System and Microsoft Visual C♯.NET 2008.

4.1 INEX Collections

1. On the Snippet retrieval track, the document collections are from the INEX-Wiki09 collection was created from the October 8, 2008 dump of English Wikipedia articles, and incorporates semantic annotations from the 2008-w40-2 version of YAGO [6]. It contains 2,666,190 Wikipedia articles and has a total uncompressed size of 50.7 GB. There are 101,917,424 XML elements of at least 50 characters.
2. On the Data Centric track, Information about one movie or person is published in one XML file[5], it was generated from the plain text files published on the IMDB web site on April 10, 2010. In total, the IMDB data collection contains 4,418,081 XML documents, including 1,594,513 movies, 1,872,471 actors, 129,137 directors, and the total size is 1.40 GB.

5 Results and Discussion

5.1 Snippet Retrieval Track

In the INEX 2011 Snippet Retrieval Track, Our system retrieves XML elements based on both indices are including Selected Weight and Leaf-Node. In our engine, the Selected Weight is based on the Term Frequency, and the Leaf-Node is based on BM25. In the snippet generation system, we use significant words, the element of relevance from both Selected Weight and Leaf-Node relevance, and then we combine relevance context to the sentences. Afterwards, The sentences with the higher relevance score will be chosen as the retrieval snippet.

In this section, we present the results used to evaluate our system. In principle, any portion of an XML document can be retrieved, although some portions are more likely to be relevant to the user query. We submitted three runs; p16-kas16-MEXIR-ALL, p16-kas16-MEXIR-ANY and p16-kas16-MEXIR-EXT as shown in Table 1 base on the Geometric Mean of recall and negative recall (GM), the Mean Precision Accuracy (MPA), the Mean Normalised Prediction Accuracy (MNPA), the Recall, the Negative Recall (NR), the Positive Agreement (PA) and the Negative Agreement (NA) metrics.

Table 1. Our performing runs based on INEX Evaluation metrics

Run	GM	MPA	MNPA	Recall	NR	PA	NA
p16-kas16-MEXIR-ALL	0.0000	0.5692	0.2846	0.0000	0.5692	0.0000	0.6360
p16-kas16-MEXIR-ANY	0.0000	0.8942	0.4471	0.0000	0.8942	0.0000	0.9355
p16-kas16-MEXIR-EXT	0.0000	0.8786	0.4393	0.0000	0.8786	0.0000	0.9286

[5] https://inex.mmci.uni-saarland.de/

5.2 Data Centric Track

Table 2 shown an overview of the 10 best performing runs for this track. Our run **p16-kas16-MEXIR-2-EXT-NSW** for the Ad Hoc Task ranked Third best scoring is 0.3479 with the highest score on mean reciprocal rank (1/rank) is 0.6999 measured with MAP.

Table 2. Best performing runs based on MAP over all ad hoc topics

Run	MAP	1/Rank	P@10	P@20	P@30
p4-UAms2011adhoc	0.3969	0.6991	0.4263	0.3921	0.3579
p2-ruc11AS2	0.3829	0.6441	0.4132	0.3842	0.3684
p16-kas16-MEXIR-2-EXT-NSW	**0.3479**	**0.6999**	**0.4316**	**0.3645**	**0.3298**
p77-PKUSIGMA01CLOUD	0.3113	0.5801	0.4421	0.4066	0.3851
p18-UPFbaseCO2i015	0.2696	0.5723	0.4342	0.4171	0.3825
p30-2011CUTxRun2	0.2099	0.6104	0.3684	0.3211	0.2965
p48-MPII-TOPX-2.0-co	0.1964	0.5698	0.3684	0.3395	0.3289
p47-FCC-BUAP-R1	0.1479	0.5120	0.3474	0.2763	0.2412
p12-IRIT-focus-mergeddtd-04	0.0801	0.2317	0.2026	0.1724	0.1702

In Table 3 shows the detail for each run that we submitted in INEX Data Centric track. For the structure weight, Table 4 shown the results compare over the best performing runs with and without Structure Weight. The **p16-kas16-MEXIR-2-EXT-NSW** is used the Structure Weight and the without is the **p16-kas16-MEXIR-EXT2-NSW** and then the structure weight shown improve the effectiveness of search system measured in terms of MAP, P@10, P@20 and P@30 are 52.60%, 50.60%, 54.16% and 58.79%, respectively. However, for the Score Sharing, Table 5 shown the results compare over the best performing runs with and without the Score Sharing technique. The **p16-kas16-MEXIR-2-EXT-NSW** is used the Structure Weight and the used the Score Sharing is the **p16-kas16-BM25W-SS-SW** and then the structure weight shown improve the effectiveness of over the Score Sharing technique measured in terms of MAP, P@10, P@20 and P@30 are 81.58%, 82.92%, 75.09% and 67.83%, respectively.

Table 3. The details for each runs in INEX Data Centric track

Run	Structure Weight	Score Sharing
p16-kas16-MEXIR-2-EXT-NSW	YES	NO
p16-kas16-MEXIR-2-ANY-NSW	YES	NO
p16-kas16-MEXIR-2-ALL-NSW	YES	NO
p16-kas16-MEXIR-EXT2-NSW	NO	NO
p16-kas16-MEXIR-ANY2-NSW	NO	NO
p16-kas16-MEXIR-ALL2-NSW	NO	NO
p16-kas16-BM25W-SS-SW	NO	YES
p16-kas16-BM25W-NSS-SW	NO	YES
p16-kas16-BM25W-SS-NSW	NO	YES

Table 4. Compare performing runs based on MAP with and without the Structure Weight

Run	MAP	P@10	P@20	P@30
p16-kas16-MEXIR-2-EXT-NSW	0.3479	0.4316	0.3645	0.3298
p16-kas16-MEXIR-2-ANY-NSW	0.2125	0.2500	0.2171	0.1930
p16-kas16-MEXIR-2-ALL-NSW	0.1937	0.2342	0.1921	0.1675
p16-kas16-MEXIR-EXT2-NSW	0.1830	0.2184	0.1974	0.1939
p16-kas16-MEXIR-ANY2-NSW	0.0857	0.1447	0.1118	0.0921
p16-kas16-MEXIR-ALL2-NSW	0.0857	0.1447	0.1118	0.0921
%	**52.60**	**50.60**	**54.16**	**58.79**

Table 5. Compare performing runs based on MAP with the Score Sharing

Run	MAP	P@10	P@20	P@30
p16-kas16-MEXIR-2-EXT-NSW	0.3479	0.4316	0.3645	0.3298
p16-kas16-MEXIR-2-ANY-NSW	0.2125	0.2500	0.2171	0.1930
p16-kas16-MEXIR-2-ALL-NSW	0.1937	0.2342	0.1921	0.1675
p16-kas16-BM25W-SS-SW	0.0641	0.0737	0.0908	0.1061
p16-kas16-BM25W-NSS-SW	0.0641	0.0711	0.0882	0.1044
p16-kas16-BM25W-SS-NSW	0.0606	0.0605	0.0829	0.1070
%	**81.58**	**82.92**	**75.09**	**67.83**

In addition, our run **p16-kas16-MEXIR-2-EXT-NSW** shown the result ranked the first best scoring is 0.3564 measured with MAP over only the informational topics, ranked the fifth scoring is 0.6667 measured with 1/Rank over only the know-item topics, ranked the first best scoring is 0.4251 measured with MAP over only the list topics, and ranked the forth scoring is 0.2647 measured with MAP over only the faceted topics.

6 Conclusions

Since, the XML documents have additional information; document representation of these might be added up meta-data to describe data in context respect to XML language design. Due to the grows in electronic information available there has been an increase in the size of the collection used. In this paper, we have presented the details about implementation of the XML retrieval system which was used to evaluate the task of ad-hoc retrieval of XML documents which we call MEXIR, in particular, in the data centric track of the INEX 2011. The objective is aiming to use the structure weighted to improve the effectiveness of the search systems. In addition, this technique runs for the ad-hoc task of data centric shown the structural information could be utilized to improve the effectiveness of the search system up to 58.79%.

In future work, we plan to study the sensitivity of the evaluation to improve our system respect to the snippet retrieval track.

References

1. Geva, S., Kamps, J., Lethonen, M., Schenkel, R., Thom, J.A., Trotman, A.: Overview of the INEX 2009 Ad Hoc Track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 4–25. Springer, Heidelberg (2010)
2. Ricardo, B., Berthier, R.: Modern Information Retrieval. Addison Wesley Longman Publishing Co. Inc. (1999)
3. Ricardo, B., Berthier, R.: Modern Information Retrieval, 2nd edn. Addison Wesley Longman Publishing Co. Inc. (2011)
4. Robertson, S.E., Walker, S., Jones, S., Hancock, B.M.M., Gatford, M.: Okapi at TREC-3. In: Harman, D.K. (ed.) Proceedings of the Third Text REtrieval Conference, TREC-3 (April 1995)
5. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. R. Donnelley & Sons Company, USA (1983)
6. Schenkel, R., Suchanek, F.M., Kasneci, G.: YAWN: A semantically annotated Wikipedia XML corpus. In: 12. GI-Fachtagung fur Datenbanksysteme in Business, Technologie und Web (BTW 2007), pp. 277–291 (2007)
7. Trappett, M., Geva, S., Trotman, A., Scholer, F., Sanderson, M.: Overview of the INEX 2011 Snippet Retrieval Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 283–294. Springer, Heidelberg (2012)
8. Trotman, A., Sigurbjörnsson, B.: Narrowed Extended XPath I (NEXI). In: Fuhr, N., Lalmas, M., Malik, S., Szlávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 16–40. Springer, Heidelberg (2005)
9. Trotman, A., Sigurbjörnsson, B.: NEXI, Now and Next. In: Fuhr, N., Lalmas, M., Malik, S., Szlávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 41–53. Springer, Heidelberg (2005)
10. Trotman, A., Lalmas, M.: The Interpretation of CAS. In: Fuhr, N., Lalmas, M., Malik, S., Kazai, G. (eds.) INEX 2005. LNCS, vol. 3977, pp. 58–71. Springer, Heidelberg (2006)
11. Trotman, A., Wang, Q.: Overview of the INEX 2010 Data Centric Track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 171–181. Springer, Heidelberg (2011)
12. Wichaiwong, T., Jaruskulchai, C.: XML Retrieval More Efficient Using Double Scoring Scheme. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 351–362. Springer, Heidelberg (2011)
13. Wichaiwong, T., Jaruskulchai, C.: An Extension XML Compression Technique for XML Element Retrieval. Information-An International Interdisciplinary Journal 15 (2012)
14. Wichaiwong, T., Jaruskulchai, C.: XML Retrieval More Efficient Using ADXPI Indexing Scheme. In: The 4th International Symposium on Mining and Web, Biopolis, Singapore (2011)
15. Wichaiwong, T., Jaruskulchai, C.: MEXIR: An Implementation of High Performance and High Precision XML Information Retrieval. Computer Technology and Application 2(4) (2011)
16. Wichaiwong, T., Jaruskulchai, C.: A Score Sharing Method for XML Element Retrieval. Information-An International Interdisciplinary Journal 15 (2012)

Overview of the INEX 2011 Question Answering Track (QA@INEX)

Eric SanJuan[1], Véronique Moriceau[2], Xavier Tannier[2], Patrice Bellot[3], and Josiane Mothe[4]

[1] LIA, Université d'Avignon et des Pays de Vaucluse, France
eric.sanjuan@univ-avignon.fr
[2] LIMSI-CNRS, University Paris-Sud, France
{moriceau,xtannier}@limsi.fr
[3] LSIS - Aix-Marseille University, France
patrice.bellot@univ-amu.fr
[4] IRIT, Toulouse University, France
josiane.mothe@irit.fr

Abstract. The INEX QA track aimed to evaluate complex question-answering tasks where answers are short texts generated from the Wikipedia by extraction of relevant short passages and aggregation into a coherent summary. In such a task, Question-answering, XML/passage retrieval and automatic summarization are combined in order to get closer to real information needs. Based on the groundwork carried out in 2009-2010 edition to determine the sub-tasks and a novel evaluation methodology, the 2011 edition experimented contextualizing tweets using a recent cleaned dump of the Wikipedia. Participants had to contextualize 132 tweets from the New York Times (NYT). Informativeness of answers has been evaluated, as well as their readability. 13 teams from 6 countries actively participated to this track. This tweet contextualization task will continue in 2012 as part of the CLEF INEX lab with same methodology and baseline but on a much wider range of tweet types.

Keywords: Question Answering, Automatic Summarization, Focus Information Retrieval, XML, Natural Language Processing, Wikipedia, Text Readability, Text informativeness.

1 Introduction

Since 2008, Question Answering (QA) track at INEX [7] moved into an attempt to bring together Focused Information Retrieval (FIR) intensively experimented in other INEX tracks (previous ad-hoc tracks [4] and this year snippet track) on the one hand, and topic oriented summarization tasks as defined in NIST Text Analysis Conferences (TAC) [3] on the other hand. Like in recent FIR INEX tasks, the corpus is a clean XML extraction of the content of a dump from Wikipedia. However QA track at INEX differs from current FIR and TAC summarization tasks on the evaluation metrics they use to measure both informativeness and readability. Following [5,10], informativeness measure is based

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 188–206, 2012.

on lexical overlap between a pool of relevant passages (RPs) and participant summaries. Once the pool of relevant passages is constituted, the process is automatic and can be applied to unofficial runs. The release of these pools is one of the main contributions of INEX QA track. By contrast, readability evaluation is completely manual and cannot be reproduced on unofficial runs. It is based on questionnaires pointing out possible syntax problems, broken anaphora, massive redundancy or other major readability problems.

Therefore QA tasks at INEX moved from the usual IR *query / document* paradigm towards *information need / text answer*. More specifically, the task to be performed by the participating groups of INEX 2011 was contextualizing tweets, *i.e.* answering questions of the form "what is this tweet about?". The general process involved:

- Tweet analysis,
- Passage and/or XML element retrieval,
- Construction of the answer.

We target systems efficient on small terminals like smart phones, based on local resources that do not require a network access, gathering non factual contextual information that is scattered around local resources. Off-line applications on portable devices are useful to reduce the network load and safer.

Answers could contain up to 500 words. It has been required that the answer uses only elements previously extracted from the document collection. Answers needed to be a concatenation of textual passages from the Wikipedia dump.

To constitute the pool of RPs, the informativeness of all returned passages for a subset of 50 tweets has been assessed by organizers. The pool of RPs included all passages considered as relevant by at least one assessor (each passage being submitted to two assessors). We regarded as informative passages that both contain relevant information but also contained as little non-relevant information as possible (the result is specific to the question). This year, long passages including several sentences have often been considered as uninformative because they included too much non relevant information. Furthermore, informativeness of a passage was established exclusively based on its textual content, and not on the documents from which it was extracted. Despite the use of a pool of RPs, the informativeness value of answers did not only rely on the number of its RPs, but also on lexical overlap with other RPs. We found out that evaluating informativeness based on lexical overlap with a pool of RPs is robust if the variety of participant systems is large enough and includes strong baselines.

The paper is organized as follows. Section 2 presents the description of the task. Section 3 details the collection of tweets and documents. Section 4 describes the baseline system provided by the track organizers. Section 5 presents the techniques and tools used for manual evaluation, explains the final choice of metrics and presents results. Finally, Section 6 presents 2012 CLEF "tweet contextualization" task before drawing some conclusions in Section 7.

2 Task Description

The underlying scenario is to provide the user with synthetic contextual information when receiving a message like a tweet. The task is not to find an exact answer in a database of facts, but to bring out the background of the message exclusively based on its textual content. Therefore the answer needs to be built by aggregation of textual passages grasped from the resource (Wikipedia in our case). For some topics, there can be too many relevant passages that cannot be all inserted in the answer, requiring some summarization process that preserves overall informativeness. For others, only few information can be available and the answer should be shorter than expected pointing out the lack of available information.

In this edition, we have considered a recent dump of the Wikipedia. Since we target non factual answers but short contextualizing texts, we removed all the info boxes and the external references, leaving only the textual content with all its document structure (title, abstract, sections, subtitles and paragraphs) and its internal references (links towards other pages).

We wanted to consider only highly informative tweets. In this attempt to define a contextualizing task, we chose to follow the New York Times (NYT) Twitter account. As soon as the NYT publishes an article on its website, it tweets the title of this article, with its URL. We thus considered these tweets. Therefore the task had become *"given a NYT title, find and summarize all available background information in the Wikipedia"*. We also added the first sentence of the related NYT article as a hint, but only few runs used this hint and none of the participants reported using NYT paper content: all tried to tackle the contextualization task in an off-line approach using only the available corpus.

The aggregated answers had a maximum of 500 words each and have been evaluated according to:

- Their informativeness (how much they overlap with relevant passages, Section 5.2),
- Their readability (assessed by evaluators, Section 5.3).

The informativeness of a summary cannot be evaluated without its readability since informative content measures tend to favor syntactic dense summaries. It is often possible to increase an informativeness score by weakening its discursive structure and thus its readability [9].

We provided the participants with a state of the art system derived from [12,2]. Participants had to improve its informative performance without weakening too much the readability of its results.

It was initially announced that readability would be evaluated by participants according to the "last point of interest", *i.e.* the first point after which the text becomes unreadable because of:

- syntactic incoherence,
- unsolved anaphora,

– redundancy,
– other problems.

After discussion between organizers and participants at the INEX 2011 workshop, it was finally decided to disclaim considering only the last point of interest because it relied too much on assessors' subjectivity but to mark all readability issues for every sentence in a summary. It was also decided to evaluate the readability independently from the topic to be contextualized and to read all passages, even if redundant. This increased the workload left to participants in readability evaluation but resulted in a much more refined analysis.

3 Track Data

From 2009 to 2010, QA track at INEX worked on the ad-hoc Wikipedia document collection. In 2009 we considered questions related to ad-hoc topics, and in 2010, real-user, non factual questions from the OverBlog platform[1]. Best performing systems on this task were state of the art automatic summarizers that pick up few Wikipedia pages related to the question and provided a summary as answer.

In 2011, the QA track started experimenting tweets instead of real questions. There the overlap between topics and Wikipedia content becomes much weaker than previously. It was thus decided to move to a more recent and simplified dump of Wikipedia. The new corpus was made available in October 2011 leaving two months to participants for their experiments. This corpus generation process has been completely automatized and can be apply to any XML Wikipedia dump.

3.1 Questions

The question data set was composed of 132 tweets by the NYT released on the July 20th 2011 and having a URL towards the NYT website. Each topic includes the tweet which is often the title of an article just released and the first sentence of the related article. An example is provided below:

```
<topic id="2011005">
  <title>Heat Wave Moves Into Eastern U.S</title>
  <txt>The wave of intense heat that has enveloped much of the
    central part of the country for the past couple of weeks is
    moving east and temperatures are expected to top the 100-degree
    mark with hot, sticky weather Thursday in cities from
    Washington, D.C., to Charlotte, N.C.</txt>
</topic>
```

[1] http://www.over-blog.com/

All these topics were twitted three months after the Wikipedia dump used to build the corpus, therefore we had to manually check if there was any related information in the document collection[2]

3.2 Document Collection

The document collection has been built based on a dump of the English Wikipedia from April 2011. Since we target a plain XML corpus for an easy extraction of plain text answers, we removed all notes and bibliographic references that are difficult to handle and kept only the 3,217,015 non empty Wikipedia pages (pages having at least one section).

Resulting documents are made of a title (`title`), an abstract (`a`) and sections (`s`). Each section has a sub-title (`h`). Abstract and sections are made of paragraphs (`p`) and each paragraph can have entities (`t`) that refer to other Wikipedia pages.

Therefore the resulting corpus follows this DTD:

```
<!ELEMENT xml (page)+>
<!ELEMENT page (ID, title, a, s*)>
<!ELEMENT ID (#PCDATA)>
<!ELEMENT title (#PCDATA)><!ELEMENT a (p+)>
<!ELEMENT s (h, p+)>
<!ATTLIST s o CDATA #REQUIRED>
<!ELEMENT h (#PCDATA)>
<!ELEMENT p (#PCDATA | t)*>
<!ATTLIST p o CDATA #REQUIRED>
<!ELEMENT t (#PCDATA)>
<!ATTLIST t e CDATA #IMPLIED>
```

Figure 1 shows an example of such a cleaned article. We have released the scripts used to generate this corpus. They process any recent XML dump of the Wikipedia in two steps:

- a light `awk` command to remove in a single pass all external references, info boxes and notes using a fast substring extraction function based on index function (GNU implementation of strchr C ISO function).
- a `perl` program that generates the XML using regular expressions to detect and encapsulate document structure and internal links. It also works in a single pass.

[2] The resulting 132 topics come from an initial set of 205 tweets after removing duplicates due to single subjects producing several papers (like different testimonies and opinion papers about the same subject) and only few tweets for which there was no overlap with the Wikipedia. Hence, the 132 selected topics represent more than 64% of the tweets released by the NYT in one day.

```
<?xml version="1.0" encoding="utf-8"?>
<page>
<ID>2001246</ID>
<title>Alvin Langdon Coburn</title>
<s o="1">
<h>Childhood (1882-1899)</h>
<p o="1">Coburn was born on June 11, 1882, at 134 East Springfield
Street in <t>Boston, Massachusetts</t>, to a middle-class family.
His father, who had established the successful firm of
Coburn &amp;amp; Whitman Shirts, died when he was seven. After that he
was raised solely by his mother, Fannie,  who remained the primary
influence in his early life, even though she remarried when he was
a teenager. In his autobiography, Coburn wrote, &amp;quot;My mother was
a remarkable woman of very strong character who tried to dominate
my life. It was a battle royal all the days of our life
together.&amp;quot;</p>
<p o="2">In 1890 the family visited his maternal uncles in
Los Angeles, and they gave him a 4 x 5 Kodak camera. He immediately
fell in love with the camera, and within a few years he had developed
a remarkable talent for both visual composition and technical
proficiency in the <t>darkroom</t>. (...)</p>
(...)
</page>
```

Fig. 1. An example of a cleaned Wikipedia XML article

Once generated, it is necessary to check if the resulting large XML file (between 8 and 12 Gb for recent Wikipedia dumps) is valid. We use the Perl TWIG library by Michel Rodriguez[3] for that. This is a robust library that can process large XML files page by page and fix eventual illformed ones.[4] Current indexers like Indri do not parse such a large XML file and require to split it into pages organized in some folder structure avoiding too large folders. We also made available a Perl program that dispatches Wikipedia pages in 1000 folders. This process can take hours because of numerous file operations.

A complementary list of non-Wikipedia entities has also been made available. The named entities (person, organisation, location, date) of the document collection have been tagged using XIP [1]. For example, for the previous documents, the extracted named entities are:

[3] `http://search.cpan.org/ mirod/`

[4] We had to manually correct few errors on the April 2011 Wikipedia dump due to encoding errors in the original dump file itself, but we did not have error anymore in the last Wikipedia dump from November 2011. For the 2011 INEX edition, we used the corrected April 2011 dump.

```
Alvin Langdon Coburn
1882-1899
Coburn
June 11, 1882
134 East Springfield Street
Boston, Massachusetts
Coburn Whitman
Fannie
Coburn
1890
Los Angeles
Kodak
```

This can be used for participants willing to use named entities in texts but not having their own tagger.

3.3 Submission Requirements

Participants could submit up to three runs. Despite the fact that manual runs were allowed if there was at least one automatic, all submitted official runs have been registered as fully automatic.

Results were lists of passages extracted from the corpus. Two non consecutive passages had to be presented separately. Results in a single run could not include more than 500 words per topic. Any string of alphanumeric characters outside XML tags, without space or punctuation, was considered as a single word.

The format for results was a variant of the familiar TREC format with additional fields:[5]

```
<qid> Q0 <file> <rank> <rsv> <run_id> <column_7> <column_8>
```

where:

- The first column `qid` is the topic number.
- The second column is currently unused and should always be `Q0`. It is just a formating requirement used by the evaluation programs to distinguish between official submitted runs and q-rels.
- The third column `file` is the file name (without `.xml`) from which a result is retrieved, which is identical to the `<id>` of the Wikipedia document. It is only used to retrieve the raw text content of the passage, not to compute document retrieval capabilities. In particular, if two results only differ by their document id (because the text is repeated in both), then they will be considered as identical and thus redundant.

[5] The XML format to submit results originally proposed in 2010 was dismissed since it was never used by participants because of its useless extra complexity. However if the task evolves in the following years towards more complex results, TREC-like formats will not be sufficient and some XML formating will be required.

- The fourth column rank indicates the order in which passages should be read for readability evaluation, this differs from the expected informativeness of the passage who is indicated by the score rsv in the fifth column. Therefore, these two columns are not necessarily correlated. Passages with highest scores in the fifth column can be scattered at any rank in the result list for each topic.
- The sixth column run_id is called the "run tag" and should be a unique identifier for the participant group and for the method used.
- The remaining two columns indicate the selected passage in the document mentioned in the third field. Participants could refer to these passages as File Offset Lengths (FOL) like in usual INEX FIR tasks or directly give the raw textual content of the passage. However, computing character offsets can be tricky dependent on the text encoding and Wikipedia often mixes different encodings. Therefore all participants to this edition chose the alternative raw text format. In this format, each result passage is given as raw text without XML tags and without formatting characters. The only requirement is that the resulting word sequence has to appear at least once in the file indicated in the third field.

Here is an example of such an output:

```
2011001 Q0 3005204 1 0.9999 I10UniXRun1 The Alfred Noble Prize is ...
2011001 Q0 3005204 2 0.9998 I10UniXRun1 The prize was established in ...
2011001 Q0 3005204 3 0.9997 I10UniXRun1 It has no connection to the ...
```

4 Baselines

A baseline XML-element retrieval/summarization system has been made available for participants. The 2011 INEX QA baseline relies on:

- An index powered by Indri[6] that covers all words (no stop list, Krowetz stemming) and all XML tags.
- A PartOfSpeech tagger powered by TreeTagger[7].
- A fast summarizer algorithm powered by TermWatch[8] introduced in [2].
- A summary content evaluation based on FRESA[10].

The Indri index allows to experiment different types of queries to seek for all passages in the Wikipedia involving terms in the topic. Queries can be usual bag of words, sets of weighted multi-word phrases or more complex structured queries using Indri Language[6]. All extracted passages are segmented into sentences and PoS tagged using the TreeTagger. Sentences are then scored using TermWatch based on their *nominals* (i.e. its nouns and adjectives). Let Φ be the set of

[6] http://www.lemurproject.org/
[7] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/
[8] http://data.termwatch.es

sentences. If for each sentence $\phi \in \Phi$, we denote by φ_ϕ the set of its nominals, then the sentence score Θ_ϕ computed in [2] is:

$$\Theta_\phi = \sum_{\varphi_\phi \cap \varphi_\tau \neq \emptyset}^{\tau \in \Phi} \sum_{\varphi_\tau \cap \varphi_\sigma \neq \emptyset}^{\sigma \in \Phi} |\varphi_\phi \cap \varphi_\tau| \times |\varphi_\tau \cap \varphi_\sigma| \quad (1)$$

The idea is to weight the sentences according to the number of sentences in their neighborhood (sentences sharing at least one nominal). This gives a fast approximation of TextRank or LexRank scores [2]. Sentences are then ranked by decreasing score, only the top ranked are used for a summary of less than 500 words. The selected sentences are then re-ordered following the Indri score of the passage from which they have been extracted and the order of the sentences in these passages. This baseline summary can be computed on the fly, generating the summary taking less time than processing the query by Indri.

This system has been made available online to participants through a web interface[9]. A Perl API running on Linux to query the server was also released. By default, this API takes as input a tabulated file with three fields: topic names, selected output format and query. The output format can be the baseline summary or the first 50 retrieved documents in raw text, PoS tagged or XML source. An example of such a file allowing to retrieve 50 documents per topic based on their title was also released.

The web interface also allows to evaluate the resulting summary or user's one against the retrieved documents using Kullback-Leibler (KL) measure. This content summary evaluation also gives a lower bound using a random set of 500 words extracted from the texts and an upper bound using an empty summary. Random summaries naturally reach the closest word distributions but they are clearly unreadable.

Two baselines were then computed using the approach described in [2] and added to the pool of official submissions:

- Baseline_sum using only topic titles as bag of word queries and top ranked 50 full documents retrieved by Indri to build the summary.
- Baseline_mwt using the same process but returning only the Noun Phrases in the selected sentences to simulate a baseline run for Automatic Terminology Extractors.

5 Evaluation

In this task, readability of answers [9] is as important as the informative content. Summaries must be easy to read as well as relevant. These two properties have been evaluated separately by two distinct measures: *informativeness* and *readability*.

[9] http://qa.termwatch.es

5.1 Submitted Runs

23 valid runs by 11 teams from 6 countries (Brasil, Canada, France, India, Mexico, Spain) were submitted. All runs are in raw text format and almost all participants used their own summarization system. Only three participants did not use the online Indri IR engine. Some participants used the Perl API to query the Indri Index with expanded queries based on semantical resources. Only one participant used XML tags.

The total number of submitted passages is 37,303. The median number of distinct passages per topic is 284.5 and the median length in words is 26.9. This relative small amount of distinct passages could be due to the fact that most of the participants used the provided Indri index with its Perl API.

5.2 Informativeness Evaluation

Informativeness evaluation has been performed by organizers on a pool of 50 topics. For each of these topics, all passages submitted have been evaluated. Only passages starting and ending by the same 25 characters have been considered as duplicated, therefore short sub-passages could appear twice in longer ones. For each topic, all passages from all participants have been merged and displayed to the assessor in alphabetical order. Therefore, each passage informativeness has been evaluated independently from others, even in the same summary. The structure and readability of the summary was not assessed in this specific part, and assessors only had to provide a binary judgment on whether the passage was worth appearing in a summary on the topic, or not. This approach handicaps runs based on short passages extracted from the Wikipedia, since very short passages can be difficult to assess on their own and tend not to be included in the pool of relevant passages.

To check that the resulting pool of relevant answers is sound, a second automatic evaluation for informativeness of summaries has been carried out with respect to a reference made of the NYT article corresponding to the topic. Official evaluation could not be based on these references since most of these articles were still available on the NYT website or could have been used by participants who are NYT readers. Nevertheless, a strong correlation between the ranking based on the assessed pool of relevant passages and the one based on NYT articles would be an indication of assessment soundness.

Metrics. Systems had to make a selection of the most relevant information, the maximal length of the abstract being fixed. Focused IR systems could just return their top ranked passages meanwhile automatic summarization systems need to be combined with a document IR engine. Both need to be evaluated. Therefore answers cannot be any passage of the corpus, but at least well formed sentences. As a consequence, informative content of answers cannot be evaluated using standard IR measures since QA and automatic summarization systems do not try to find all relevant passages but to select those that could provide a comprehensive answer. Several metrics have been defined and experimented with at DUC [8] and

TAC workshops [3]. Among them, Kullback-Leibler (KL) and Jenssen-Shanon (JS) divergences have been used [5,10] to evaluate the informativeness of short summaries based on a bunch of highly relevant documents.

In this edition we intended to use the KL one with Dirichlet smoothing, like in the 2010 edition[11], to evaluate the informative content of answers by comparing their n-gram distributions with those from all assessed relevant passages. However, in 2010, references were made of complete Wikipedia pages, therefore the textual content was much longer than summaries and smoothing did not introduce too much noise.

This is not the case with the 2011 assessments. For some topics, the amount of relevant passages is very low, less than the maximal summary length. Therefore using any probabilistic metric requiring some smoothing produced very unstable rankings. We thus simply considered absolute log-diff between frequencies. Let T be the set of terms in the reference. For every $t \in T$, we denote by $f_T(t)$ its frequency in the reference and by $f_S(t)$ its frequency in the summary. Adapting the FRESA package available to participants, we computed the divergence between reference and summaries as:

$$Div(T,S) = \sum_{t \in T} |\log(\frac{f_T(t)}{f_T} + 1) - \log(\frac{f_S(t)}{500} + 1)| \quad (2)$$

As T we considered three different sets based on the FRESA sentence segmentation, stop word list and lemmatizer:

- Unigrams made of single lemmas (after removing stop-words).
- Bigrams made of pairs of consecutive lemmas (in the same sentence).
- Bigrams with 2-gaps also made of pairs of consecutive lemmas but allowing the insertion between them of a maximum of two lemmas.

As in 2010, bigrams with 2-gaps appeared to be the most robust metric. Sentences are not considered as simple bag of words and it is less sensitive to sentence segmentation than simple bi-grams. This is why bigrams with 2-gaps is our official ranking metric for informativeness.

Results. All passages within a consistent pool of 50 topics were thoroughly evaluated by organizers. This represents 14,654 passages, among which 2,801 have been judged as relevant.

This assessment was intended to be quite generous towards passages. All passages concerning a protagonist of the topic are considered relevant, even if the main subject of the topic is not addressed. The reason is that missing words in the reference can lead to artificial increase of the *divergence*, which is a known and not desirable side effect of this measure. Results are presented in Table 1 and statistical significance of gaps between runs are indicated in Table 2.

All systems above the baseline combine a full document retrieval engine with a summarization algorithm. The three top ranked runs, all by IRIT, did not use the API provided to participants meanwhile all other runs improving the

baseline used it only to query the Indri Index, some applying special query expansion techniques. None of the participants used this year the baseline summarization system which ranks 7th among all runs when returning full sentences (`Baselinesum`) and 19th when returning only noun phrases (`Baselinemwt`).

Table 1. Informativeness results from manual evaluation using equation 2 (official results are "with 2-gap")

Rank	Run	unigram	bigram	**with 2-gap**	Average
1	ID12_IRIT_default	0.0486	0.0787	**0.1055**	0.0787
2	ID12_IRIT_07_2_07_1_dice	0.0488	0.0789	**0.1057**	0.0789
3	ID12_IRIT_05_2_07_1_jac	0.0491	0.0792	**0.1062**	0.0793
4	ID129_Run1	0.0503	0.0807	**0.1078**	0.0807
5	ID129_Run2	0.0518	0.0830	**0.1106**	0.0830
6	ID128_Run2	0.0524	0.0834	**0.1110**	0.0834
7	ID138_Run1	0.0524	0.0837	**0.1115**	0.0837
8	ID18_Run1	0.0526	0.0838	**0.1117**	0.0839
9	ID126_Run1	0.0535	0.0848	**0.1125**	0.0848
10	Baselinesum	0.0537	0.0859	**0.1143**	0.0859
11	ID126_Run2	0.0546	0.0863	**0.1144**	0.0863
12	ID128_Run3	0.0549	0.0869	**0.1151**	0.0868
13	ID129_Run3	0.0549	0.0869	**0.1152**	0.0869
14	ID46_JU_CSE_run1	0.0561	0.0877	**0.1156**	0.0876
15	ID46_JU_CSE_run2	0.0561	0.0877	**0.1156**	0.0876
16	ID62_Run3	0.0565	0.0887	**0.1172**	0.0887
17	ID123_I10UniXRun2	0.0561	0.0885	**0.1172**	0.0885
18	ID128_Run1	0.0566	0.0889	**0.1174**	0.0889
19	Baselinemwt	0.0558	0.0886	**0.1179**	0.0887
20	ID62_Run1	0.0566	0.0892	**0.1180**	0.0892
21	ID123_I10UniXRun1	0.0567	0.0895	**0.1183**	0.0894
22	ID62_Run2	0.0572	0.0900	**0.1188**	0.0899
23	ID124_UNAMiiR12	0.0607	0.0934	**0.1221**	0.0933
24	ID123_I10UniXRun3	0.0611	0.0946	**0.1239**	0.0945
25	ID124_UNAMiiR3	0.0628	0.0957	**0.1248**	0.0957

Dissimilarity values are very closed, however differences are often statistically significant as shown in table 2. In particular, top four runs are significantly better than all others. It seems that these runs carried out specific NLP post-processing. It also appears that almost all runs above Baselinesum are significantly better than those under the same baseline, meanwhile differences among runs ranked between the two baselines are rarely significant.

To check that this reference was not biased, the same 50 topics have been also automatically evaluated against the corresponding NYT article, *i.e.* taking as reference the article published under the tweeted title. None of the participants reported having used this content even though part of it was publicly available on the web.

Table 2. Statistical significance for official results in table 1 (t-test, 1 : 90%, 2 = 95%, 3 = 99%, $\alpha = 5\%$)

	ID12_IRIT_default	ID12_IRIT_07_2_07_1_dice	ID12_IRIT_05_2_07_1_jac	ID129_Run1	ID129_Run2	ID128_Run2	ID138_Run1	ID18_Run1	ID126_Run1	Baselinesum	ID126_Run2	ID128_Run3	ID129_Run3	ID46_JU_CSE_run1	ID46_JU_CSE_run2	ID62_Run3	ID123_I10UniXRun2	ID128_Run1	Baselinemwt	ID62_Run1	ID123_I10UniXRun1	ID62_Run2	ID124_UNAMiiR12	ID123_I10UniXRun3	ID124_UNAMiiR3
ID12_IRIT_default	-	-	1	-	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
ID12_IRIT_07_2_07_1_dice	-	-	1	-	1	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
ID12_IRIT_05_2_07_1_jac	1	1	-	-	1	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
ID129_Run1	-	-	-	-	2	1	3	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
ID129_Run2	2	1	1	2	-	-	-	-	-	3	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3
ID128_Run2	2	2	2	1	-	-	-	-	-	1	2	3	2	2	2	3	3	3	3	3	3	3	3	3	3
ID138_Run1	2	2	2	3	-	-	-	-	-	1	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3
ID18_Run1	3	2	2	2	-	-	-	-	-	-	-	-	1	1	1	3	3	3	3	3	3	3	3	3	3
ID126_Run1	3	3	3	2	-	-	-	-	-	-	-	-	-	-	-	2	2	3	3	3	3	3	3	3	3
Baselinesum	3	3	3	3	3	1	1	-	-	-	-	-	-	-	-	-	2	1	3	2	2	3	3	3	3
ID126_Run2	3	3	3	3	2	2	2	-	-	-	-	-	-	-	-	-	1	2	2	2	2	3	3	3	3
ID128_Run3	3	3	3	3	2	3	2	-	-	-	-	-	-	-	-	-	-	1	1	1	1	2	3	3	3
ID129_Run3	3	3	3	3	2	2	2	1	-	-	-	-	-	-	-	-	-	-	1	1	1	2	3	3	3
ID46_JU_CSE_run1	3	3	3	3	2	2	2	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	3	3
ID46_JU_CSE_run2	3	3	3	3	2	2	2	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	3	3
ID62_Run3	3	3	3	3	3	3	3	3	2	-	-	-	-	-	-	-	-	-	-	-	-	-	2	3	3
ID123_I10UniXRun2	3	3	3	3	3	3	3	3	2	2	1	-	-	-	-	-	-	-	-	-	-	1	3	3	3
ID128_Run1	3	3	3	3	3	3	3	3	3	1	2	1	-	-	-	-	-	-	-	-	-	-	2	3	3
Baselinemwt	3	3	3	3	3	3	3	3	3	3	2	1	1	-	-	-	-	-	-	-	-	-	3	3	3
ID62_Run1	3	3	3	3	3	3	3	3	3	2	2	1	1	-	-	-	-	-	-	-	-	-	2	3	3
ID123_I10UniXRun1	3	3	3	3	3	3	3	3	3	2	2	1	1	-	-	-	-	-	-	-	-	-	2	3	3
ID62_Run2	3	3	3	3	3	3	3	3	3	3	3	2	2	-	-	-	1	-	-	-	-	-	1	3	3
ID124_UNAMiiR12	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	3	2	3	2	2	1	-	-	3
ID123_I10UniXRun3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	-	-	-
ID124_UNAMiiR3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	-	-	-

Results are presented in Table 3. It appears that correlation between the two rankings is quite high (Kendall's $\tau = 0.67$, Pearson's product-moment correlation = 88%, p-value $< 9.283e^{-9}$) suggesting that our approach of selecting reference text from a pool of participant runs plus the baselines is sufficient.

All previous evaluations have been carry out using FRESA package which includes a special lemmatizer. We provided the participants with a standalone evaluation toolkit based on Potter Stemmer. Based on participant feedback after

the release of the official results, we introduced in this package a normalized ad-hoc dissimilarity defined as following using the same notations as in equation 2:

$$Dis(T,S) = \sum_{t \in T} \frac{f_T(t)}{f_T} \times \left(1 - \frac{\min(\log(P), \log(Q))}{\max(\log(P), \log(Q))}\right) \quad (3)$$

$$P = \frac{f_T(t)}{f_T} + 1 \quad (4)$$

$$Q = \frac{f_S(t)}{f_S} + 1 \quad (5)$$

The idea is to have a dissimilarity which complement has similar properties to usual IR Interpolate Precision measures. Actually, $1 - Dis(T,S)$ increases with the Interpolated Precision at 500 tokens where Precision is defined as the number of word n-grams in the reference. The introduction of the log is necessary to deal with highly frequent words.

Table 4 shows results using this evaluation toolkit implementing basic stemming and normalized dissimilarity 3. Again, the correlation with official results in

Table 3. Informativeness results automatic evaluation against NYT article using equation 2

Rank	Run	unigram	bigram	**with 2-gap**	Average
1	ID12_IRIT_05_2_07_1_jac	0.0447	0.0766	**0.1049**	0.0766
2	ID12_IRIT_07_2_07_1_dice	0.0447	0.0767	**0.1049**	0.0766
3	ID12_IRIT_default	0.0447	0.0767	**0.1049**	0.0767
4	ID129_Run1	0.0456	0.0777	**0.1060**	0.0777
5	ID18_Run1	0.0462	0.0779	**0.1061**	0.0779
6	Baselinesum	0.0460	0.0781	**0.1065**	0.0781
7	ID126_Run1	0.0460	0.0781	**0.1065**	0.0781
8	ID128_Run2	0.0461	0.0782	**0.1066**	0.0782
9	ID138_Run1	0.0461	0.0782	**0.1066**	0.0782
10	ID129_Run2	0.0468	0.0788	**0.1071**	0.0787
11	ID129_Run3	0.0468	0.0789	**0.1072**	0.0788
12	ID126_Run2	0.0469	0.0789	**0.1073**	0.0789
13	ID128_Run3	0.0469	0.0789	**0.1073**	0.0789
14	ID123_I10UniXRun1	0.0471	0.0791	**0.1075**	0.0791
15	Baselinemwt	0.0475	0.0794	**0.1077**	0.0794
16	ID62_Run1	0.0473	0.0793	**0.1077**	0.0793
17	ID128_Run1	0.0475	0.0795	**0.1079**	0.0795
18	ID62_Run3	0.0476	0.0796	**0.1080**	0.0796
19	ID62_Run2	0.0477	0.0797	**0.1080**	0.0797
20	ID123_I10UniXRun2	0.0477	0.0797	**0.1080**	0.0797
21	ID123_I10UniXRun3	0.0483	0.0804	**0.1087**	0.0803
22	ID46_JU_CSE_run1	0.0487	0.0807	**0.1089**	0.0806
23	ID46_JU_CSE_run2	0.0487	0.0807	**0.1090**	0.0807
24	ID124_UNAMiiR12	0.0493	0.0812	**0.1094**	0.0812
25	ID124_UNAMiiR3	0.0505	0.0823	**0.1104**	0.0823

Table 4. Informativeness results from manual evaluation using Potter stemmer and normalized dissimilarity 3

Rank	Run	unigram	bigram	**with 2-gap**
1	ID12_IRIT_default	0.8271	0.9012	**0.9028**
2	ID126_Run1	0.7982	0.9031	**0.9037**
3	ID12_IRIT_07_2_07_1_dice	0.8299	0.9032	**0.9053**
4	ID129_Run1	0.8167	0.9058	**0.9062**
5	ID12_IRIT_05_2_07_1_jac	0.8317	0.9046	**0.9066**
6	ID128_Run2	0.8034	0.9091	**0.9094**
7	ID138_Run1	0.8089	0.9150	**0.9147**
8	ID129_Run2	0.8497	0.9252	**0.9253**
9	ID126_Run2	0.8288	0.9306	**0.9313**
10	ID128_Run3	0.8207	0.9342	**0.9350**
11	Baselinesum	0.8363	0.9350	**0.9362**
12	ID18_Run1	0.8642	0.9368	**0.9386**
13	ID129_Run3	0.8563	0.9436	**0.9441**
14	ID46_JU_CSE1	0.8807	0.9453	**0.9448**
15	ID46_JU_CSE2	0.8807	0.9452	**0.9448**
16	ID128_Run1	0.8379	0.9492	**0.9498**
17	ID62_Run3	0.8763	0.9588	**0.9620**
18	ID123_I10UniXRun2	0.8730	0.9613	**0.9640**
19	ID62_Run1	0.8767	0.9667	**0.9693**
20	ID62_Run2	0.8855	0.9700	**0.9723**
21	ID123_I10UniXRun1	0.8840	0.9699	**0.9724**
22	ID124_UNAMiiR12	0.9286	0.9729	**0.9740**
23	Baselinemwt	0.9064	0.9777	**0.9875**
24	ID124_UNAMiiR3	0.9601	0.9896	**0.9907**
25	ID123_I10UniXRun3	0.9201	0.9913	**0.9925**

Table 1 is quite high (Kendall's τ =89%, Pearson's product-moment correlation = 96%, p-value $< 4e^{-11}$).

This normalized metric does not allow to distinguish between top ranked runs above the baseline as shown by statistical significance tests reported in table 5 but it does among runs between the two baselines.

5.3 Readability Evaluation

Human Assessment. Each participant had to evaluate readability for a pool of around 50 summaries of a maximum of 500 words each on an online web interface. Each summary consisted in a set of passages and for each passage, assessors had to tick four kinds of check boxes. The guideline was the following:

- *Syntax* (S): tick the box if the passage contains a syntactic problem (bad segmentation for example),
- *Anaphora* (A): tick the box if the passage contains an unsolved anaphora,
- *Redundancy* (R): tick the box if the passage contains a redundant information, i.e. an information that has already been given in a previous passage,

Table 5. Statistical significance for manual evaluation using Potter stemmer and normalized dissimilarity in table 4 (t-test, 1 : 90%, 2 = 95%, 3 = 99%, $\alpha = 5\%$)

	ID12_IRIT_default	ID126_Run1	ID12_IRIT_07_2_07_1_dice	ID129_Run1	ID12_IRIT_05_2_07_1_jac	ID128_Run2	ID138_Run1	ID129_Run2	ID126_Run2	ID128_Run3	Baseline_sum	ID18_Run1	ID129_Run3	ID46_JU_CSE_run2	ID46_JU_CSE_run1	ID128_Run1	ID62_Run3	ID123_I10UniXRun2	ID62_Run1	ID62_Run2	ID123_I10UniXRun1	ID124_UNAMiiR12	Baseline_mwt	ID124_UNAMiiR3	ID123_I10UniXRun3
ID12_IRIT_default	-	-	-	-	1	-	-	-	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3
ID126_Run1	-	-	-	-	-	-	-	1	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3
ID12_IRIT_07_2_07_1_dice	-	-	-	-	-	-	-	-	1	1	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3
ID129_Run1	-	-	-	-	-	-	-	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
ID12_IRIT_05_2_07_1_jac	1	-	-	-	-	-	-	-	1	1	1	2	2	2	2	3	3	3	3	3	3	3	3	3	3
ID128_Run2	-	-	-	-	-	-	-	1	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3
ID138_Run1	-	-	-	-	-	-	-	-	2	1	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3
ID129_Run2	-	1	-	2	-	1	-	-	-	-	-	-	1	2	2	2	3	3	3	3	3	3	3	3	3
ID126_Run2	2	2	1	2	1	2	2	-	-	-	-	-	-	-	-	2	2	3	3	3	3	3	3	3	3
ID128_Run3	2	2	1	2	1	2	1	-	-	-	-	-	-	-	-	1	2	3	3	3	3	3	3	3	3
Baseline_sum	2	2	2	3	1	2	2	-	-	-	-	-	-	-	-	-	2	3	3	3	3	3	3	3	3
ID18_Run1	2	2	2	3	2	2	3	-	-	-	-	-	-	-	-	-	2	3	3	3	3	3	3	3	3
ID129_Run3	2	3	2	3	2	3	3	1	-	-	-	-	-	-	-	-	1	2	3	3	3	3	3	3	3
ID46_JU_CSE_run2	3	3	2	3	2	3	3	2	-	-	-	-	-	-	-	-	1	2	3	3	3	2	3	3	3
ID46_JU_CSE_run1	3	3	2	3	2	3	3	2	-	-	-	-	-	-	-	-	1	2	3	3	3	2	3	3	3
ID128_Run1	3	3	3	3	3	3	3	2	2	1	-	-	-	-	-	-	-	2	3	3	3	2	3	3	3
ID62_Run3	3	3	3	3	3	3	3	3	2	2	2	2	1	1	1	-	-	-	-	-	-	-	3	3	3
ID123_I10UniXRun2	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2	-	-	1	2	2	-	3	3	3
ID62_Run1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	-	1	-	-	-	-	3	3	3
ID62_Run2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	-	2	-	-	-	-	2	3	3
ID123_I10UniXRun1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	-	2	-	-	-	-	2	3	3
ID124_UNAMiiR12	3	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	-	-	-	-	-	-	1	3	2
Baseline_mwt	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	2	1	-	-	-
ID124_UNAMiiR3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	-	-	-
ID123_I10UniXRun3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	-	-	-

– *Trash* (T): tick the box if the passage does not make any sense in its context (*i.e.* after reading the previous passages). These passages must then be considered at trashed, and readability of following passages must be assessed as if these passages were not present.
– If the summary is so bad that you stop reading the text before the end, tick all trash boxes until the last passage.

The assessors did not know the topic corresponding to the summary, and were not supposed to judge the relevance of the text. Only readability was evaluated.

Metrics and Results. To evaluate summary readability, we consider the number of words (up to 500) in valid passages. We used two metrics based on this:

- **Relaxed metric:** a passage is considered as valid if the T box has not been ticked,
- **Strict metric:** a passage is considered as valid if no box has been ticked.

In both cases, participant runs are ranked according to the average, normalized number of words in valid passages.

A total of 1,310 summaries, 28,513 passages from 53 topics have been assessed. All participants succeeded in evaluating more than 80% of the assigned summaries. The resulting 53 topics include all of those used for informativeness assessment. Results are presented in Table 6.

None of the submitted participant runs outperformed Baselinesum (Baseline with complete summaries). This can be explained by the fact that formula 1 favors sentences with numerous Multi Word Noun Phrases. These particular sentences tend to be long, with few pronouns, thus few broken anaphora. The drawback of this Baseline is that building an extract of 500 words made of long sentences will be always less informative than a dense coherent summary made of non redundant short sentences. Therefore participants runs had to improve informativeness without hurting readability too much.

The other baseline restricted to Multi Word Noun Phrases was considered as unreadable by most assessors except by one who is a specialist in terminology and considered as acceptable any NP that corresponds to a real Multi Word Term.

6 2012 "Tweet Contextualization" Campaign

In 2012, this campaign will be integrated into CLEF (Conference and Labs of the Evaluation Forum) under the title "tweet contextualization". The aim of this task will be close to 2011 campaign, still using the most recent cleaned dump of the Wikipedia (November 2011).

About 100 tweets will be collected manually by the organizers from Twitter. They will be selected among informative accounts (for example, @CNN, @TennisTweets, @PeopleMag, @science...), in order to avoid purely personal tweets that could not be contextualized. Information such as the user name, tags or URLs will be provided. These tweets will be used for manual evaluation, but will be scatterred into 1000 other tweets, automatically collected from Twitter Search API. This will ensure that systems provide fully automatic runs.

The tweets will be made available in a JSON format, as shown in Figure 2.

In 2012, there will be no more automatic evaluation of informativeness since we do not have any reference, as it was the case in 2011 using NYT articles. However we showed that results between manual and automatic evaluations were pretty close.

The informativeness evaluation will be performed by organizers. The readability evaluation will still be performed by organizers and participants. Only the

Table 6. Readability results with the relaxed and strict metric

Relaxed metric			**Strict metric**		
Rank	Run id	Score	Rank	Run id	Score
1	Baseline_sum	447.3019	1	Baseline_sum	409.9434
2	ID46_JU_CSE_run1	432.2000	2	ID129_Run1	359.0769
3	ID128_Run2	417.8113	3	ID129_Run2	351.8113
4	ID12_IRIT_default	417.3462	4	ID126_Run1	350.6981
5	ID46_JU_CSE_run2	416.5294	5	ID46_JU_CSE_run1	347.9200
6	ID129_Run1	413.6604	6	ID12_IRIT_05_2_07_1_jac	344.1154
7	ID129_Run2	410.7547	7	ID12_IRIT_default	339.9231
8	ID12_IRIT_05_2_07_1_jac	409.4038	8	ID12_IRIT_07_2_07_1_dice	338.7547
9	ID12_IRIT_07_2_07_1_dice	406.3962	9	ID128_Run2	330.2830
10	ID126_Run1	404.4340	10	ID46_JU_CSE_run2	330.1400
11	ID138_Run1	399.3529	11	ID129_Run3	325.0943
12	ID128_Run1	394.9231	12	ID138_Run1	306.2549
13	ID129_Run3	393.3585	13	ID128_Run3	297.4167
14	ID126_Run2	377.8679	14	ID126_Run2	296.3922
15	ID128_Run3	374.6078	15	ID62_Run2	288.6154
16	ID62_Run2	349.7115	16	ID128_Run1	284.4286
17	ID62_Run1	328.2245	17	ID62_Run3	277.9792
18	ID62_Run3	327.2917	18	ID62_Run1	266.1633
19	ID18_Run1	314.8980	19	ID18_Run1	260.1837
20	ID123_I10UniXRun2	304.1042	20	ID123_I10UniXRun1	246.9787
21	ID123_I10UniXRun1	295.6250	21	ID123_I10UniXRun2	246.5745
22	ID123_I10UniXRun3	272.5000	22	ID123_I10UniXRun3	232.6744
23	ID124_UNAMiiR12	255.2449	23	ID124_UNAMiiR12	219.1875
24	ID124_UNAMiiR3	139.7021	24	Baseline_mwt	148.2222
25	Baseline_mwt	137.8000	25	ID124_UNAMiiR3	128.3261


```
{"created_at":"Fri, 03 Feb 2012 09:10:20 +0000",
"from_user":"XXX",
"from_use_id":XXX,
"from_use_id_str":"XXX",
"from_use_name":"XXX",
"geo":null,
"id":XXX,
"id_str":"XXX",
"iso_language_code":"en",
"metadata":"result_type":"recent",
"profile_image_url":"http://XXX",
"profile_image_url_https":"https://XXX",
"source":"<a href=''http://XXX",
"text":"blahblahblah",
"to_user":null,
"to_use_id":null,
"to_use_id_str":null,
"to_use_name":null}
```


Fig. 2. Example of a tweet, in JSON format

textual content of the tweet should be contextualized. For example, contextualization of a tweet from Barack Obama account, concerning war in Syria, should not come into details about Obama's life, or US elections, but only on the tweet text.

7 Conclusion

This track that brings together the NLP and the IR communities is getting more attention. The experimented measures used for evaluation based on textual content more than passage offsets seem to reach some consensus between the two communities. Taking into account readability of summary also encourages NLP and linguistic teams to participate. Next edition will start much earlier, the corpus generation from a Wikipedia dump being now completely automatic. We plan to propose a larger variety of questions from twitter. We also would like to encourage XML systems by providing more structured questions with explicit name entities and envisage to open the track to terminology extractor systems.

References

1. At-Mokhtar, S., Chanod, J.P., Roux, C.: Robustness beyond shallowness: Incremental deep parsing. Natural Language Engineering 8, 121–144 (2002)
2. Chen, C., Ibekwe-Sanjuan, F., Hou, J.: The structure and dynamics of cocitation clusters: A multiple-perspective cocitation analysis. JASIST 61(7), 1386–1409 (2010)
3. Dang, H.: Overview of the TAC 2008 Opinion Question Answering and Summarization Tasks. In: Proc. of the First Text Analysis Conference (2008)
4. Geva, S., Kamps, J., Trotman, A. (eds.): INEX 2009. LNCS, vol. 6203. Springer, Heidelberg (2010)
5. Louis, A., Nenkova, A.: Performance confidence estimation for automatic summarization. In: EACL, pp. 541–548. The Association for Computer Linguistics (2009)
6. Metzler, D., Croft, W.B.: Combining the language model and inference network approaches to retrieval. Inf. Process. Manage. 40(5), 735–750 (2004)
7. Moriceau, V., SanJuan, E., Tannier, X., Bellot, P.: Overview of the 2009 qa track: Towards a common task for qa, focused ir and automatic summarization systems. In: Geva et al. [4], pp. 355–365
8. Nenkova, A., Passonneau, R.: Evaluating content selection in summarization: The pyramid method. In: Proceedings of HLT-NAACL, vol. 2004 (2004)
9. Pitler, E., Louis, A., Nenkova, A.: Automatic evaluation of linguistic quality in multi-document summarization. In: ACL, pp. 544–554 (2010)
10. Saggion, H., Torres-Moreno, J.M., da Cunha, I., SanJuan, E., Velázquez-Morales, P.: Multilingual summarization evaluation without human models. In: Huang, C.R., Jurafsky, D. (eds.) COLING (Posters), pp. 1059–1067. Chinese Information Processing Society of China (2010)
11. SanJuan, E., Bellot, P., Moriceau, V., Tannier, X.: Overview of the INEX 2010 Question Answering Track (QA@INEX). In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 269–281. Springer, Heidelberg (2011)
12. SanJuan, E., Ibekwe-Sanjuan, F.: Combining language models with nlp and interactive query expansion. In: Geva, et al. [4], pp. 122–132

A Hybrid QA System with Focused IR and Automatic Summarization for INEX 2011

Pinaki Bhaskar, Somnath Banerjee, Snehasis Neogi, and Sivaji Bandyopadhyay

Department of Computer Science and Engineering, Jadavpur University, Kolkata, India
{pinaki.bhaskar,s.banerjee1980,snehasis.neogi}@gmail.com,
sivaji_cse_ju@yahoo.com

Abstract. The article presents the experiments carried out as part of the participation in the QA track of INEX 2011. We have submitted two runs. The INEX QA task has two main sub tasks, Focused IR and Automatic Summarization. In the Focused IR system, we first preprocess the Wikipedia documents and then index them using Nutch. Stop words are removed from each query tweet and all the remaining tweet words are stemmed using Porter stemmer. The stemmed tweet words form the query for retrieving the most relevant document using the index. The automatic summarization system takes as input the query tweet along with the tweet's text and the title from the most relevant text document. Most relevant sentences are retrieved from the associated document based on the TF-IDF of the matching query tweet, tweet's text and title words. Each retrieved sentence is assigned a ranking score in the Automatic Summarization system. The answer passage includes the top ranked retrieved sentences with a limit of 500 words. The two unique runs differ in the way in which the relevant sentences are retrieved from the associated document. Our first run got the highest score of 432.2 in Relaxed metric of Readability evaluation among all the participants.

Keywords: Information Retrieval, Automatic Summarization, Question Answering, Information Extraction, INEX 2011.

1 Introduction

With the explosion of information in Internet, Natural language Question Answering (QA) is recognized as a capability with great potential. Traditionally, QA has attracted many AI researchers, but most QA systems developed are toy systems or games confined to laboratories and to a very restricted domain. Several recent conferences and workshops have focused on aspects of the QA research. Starting in 1999, the Text Retrieval Conference (TREC)[1] has sponsored a question-answering track, which evaluates systems that answer factual questions by consulting the documents of the TREC corpus. A number of systems in this evaluation have successfully combined information retrieval and natural language processing techniques. More recently,

[1] http://trec.nist.gov/

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 207–218, 2012.

Conference and Labs of Evaluation Forums (CLEF)[2] are organizing QA lab from 2010. INEX[3] has also started Question Answering track. This year, INEX 2011 designed a QA track [1] to stimulate the research for real world application. The Question Answering (QA) task performed by the participating groups of INEX 2011 is contextualizing tweets, i.e., answering questions of the form "what is this tweet about?" using a recent cleaned dump of the Wikipedia (April 2011).

Current INEX 2011 Question answering track gives QA research a new direction by fusing IR and summarization with QA. The QA track of INEX 2011 had two major sub tasks. The first task is to identify the most relevant document from the Wikipedia dump, for this we need a focused IR system. And the second task is to extract most relevant passages from the most relevant retrieved document. So we need an automatic summarization system. The general purpose of the task involves tweet analysis, passage and/or XML elements retrieval and construction of the answer, more specifically, the summarization of the tweet topic.

Automatic text summarization [2] has become an important and timely tool for assisting and interpreting text information in today's fast-growing information age. Text Summarization methods can be classified into abstractive and extractive summarization. An Abstractive Summarization ([3] and [4]) attempts to develop an understanding of the main concepts in a document and then expresses those concepts in clear natural language. Extractive Summaries [5] are formulated by extracting key text segments (sentences or passages) from the text, based on statistical analysis of individual or mixed surface level features such as word/phrase frequency, location or cue words to locate the sentences to be extracted. Our approach is based on Extractive Summarization.

In this paper, we describe a hybrid QA system of focused IR and automatic summarization for QA track of INEX 2011. The focused IR system is based on Nutch architecture and the automatic summarization system is based on TF-IDF based sentence ranking and sentence extraction techniques. The same sentence scoring and ranking approach of [6] and [7] has been followed. We have submitted two runs in the QA track (ID46RJU_CSE_run1 and ID46RJU_CSE_run2).

2 Related Works

Recent trend shows hybrid approach of QA using Information Retrieval (IR) can improve the performance of the QA system. Reference [8] removed incorrect answers of QA system using an IR engine. Reference [9] successfully used methods of IR into QA system. Reference [10] used the IR system into QA and [11] proposed an efficient hybrid QA system using IR in QA.

Reference [12] presents an investigation into the utility of document summarization in the context of IR, more specifically in the application of so-called query-biased summaries: summaries customized to reflect the information need expressed in a query. Employed in the retrieved document list displayed after retrieval took place, the

[2] http://www.clef-initiative.eu//
[3] https://inex.mmci.uni-saarland.de/

summaries' utility was evaluated in a task-based environment by measuring users' speed and accuracy in identifying relevant documents. This was compared to the performance achieved when users were presented with the more typical output of an IR system: a static predefined summary composed of the title and first few sentences of retrieved documents. The results from the evaluation indicate that the use of query-biased summaries significantly improves both the accuracy and speed of user relevance judgments.

A lot of research work has been done in the domain of both query dependent and independent summarization. MEAD [13] is a centroid based multi document summarizer, which generates summaries using cluster centroids produced by topic detection and tracking system. NeATS [14] selects important content using sentence position, term frequency, topic signature and term clustering. XDoX [15] identifies the most salient themes within the document set by passage clustering and then composes an extraction summary, which reflects these main themes. Graph based methods have been also proposed for generating summaries. A document graph based query focused multi-document summarization system has been described by [16], [6] and [7].

In the present work, we have used the IR system as described in [10] and [11] and the automatic summarization system as discussed in [6] and [7]. In the later part of this paper, section 3 describes the corpus statistics and section 4 shows the system architecture of combined QA system of focused IR and automatic summarization for INEX 2011. Section 5 details the Focused Information Retrieval system architecture. Section 6 details the Automatic Summarization system architecture. The evaluations carried out on submitted runs are discussed in Section 7 along with the evaluation results. The conclusions are drawn in Section 8.

3 Corpus Statistics

The training data is the collection of 3,217,015 documents that has been rebuilt based on recent English Wikipedia dump (April 2011). All notes and bibliographic references have been removed from Wikipedia pages to prepare plain xml corpus for an easy extraction of plain text answers. Each training document is made of a title, an abstract and sections. Each section has a sub-title. Abstract and sections are made of paragraphs and each paragraph can have entities that refer to Wikipedia pages. Therefore, the resulting corpus has this simple DTD as shown in table 1.

Test data is made up of 132 tweets (questions) from the New York Times (NYT) paper. Each tweet includes title and first sentence of NYT paper in XML format as shown in table 2. For example,

```
<topic id="2011001">
    <title>At Comic-Con, a Testing Ground for Toymakers</title>
    <txt>This summer's hottest toys won't be coming to a toy aisle near you. The
        only place to get them will be at Comic-Con International in San
        Diego.</txt>
</topic>
```

Table 1. The DTD for Wikipedia pages

```
<!ELEMENT xml (page)+>
<!ELEMENT page (ID, title, a, s*)>
<!ELEMENT ID (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT a (p+)>
<!ELEMENT s (h, p+)>
<!ATTLIST s o CDATA #REQUIRED>
<!ELEMENT h (#PCDATA)>
<!ELEMENT p (#PCDATA | t)*>
<!ATTLIST p o CDATA #REQUIRED>
<!ELEMENT t (#PCDATA)>
<!ATTLIST t e CDATA #IMPLIED>
```

Table 2. XML tag format of NYT tweets of INEX 2011 corpus

```
<xml>
   <topic id="number">
      <title> tweeted NYT title </title>
      <text> first sentence of the news </text>
   </topic>
</xml>
```

4 System Architecture

In this section the overview of the system framework of the current INEX system has been shown. The current INEX system has two major sub-systems; one is the Focused IR system and the other one is the Automatic Summarization system. The Focused IR system has been developed on the basic architecture of Nutch[4], which use the architecture of Lucene[5]. Nutch is an open source search engine, which supports only the monolingual Information Retrieval in English, etc. The Higher-level system architecture of the combined QA system of Focused IR and Automatic Summarization is shown in the Figure 1.

5 Focused Information Retrieval (IR)

5.1 Wikipedia Document Parsing and Indexing

The web documents are full of noises mixed with the original content. In that case it is very difficult to identify and separate the noises from the actual content. INEX 2011

[4] http://nutch.apache.org/
[5] http://lucene.apache.org/

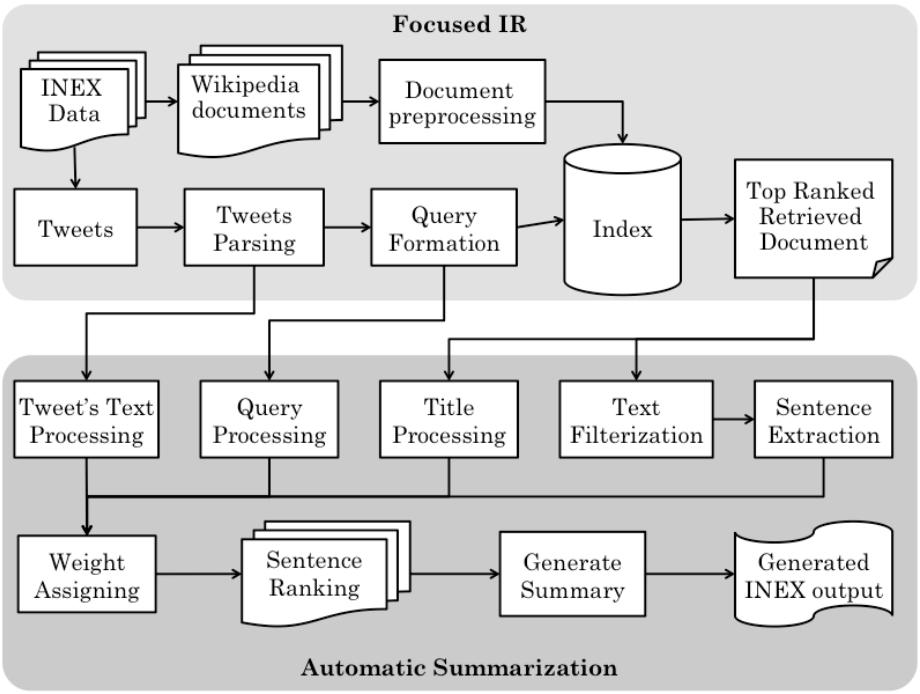


Fig. 1. Higher level system architecture of current INEX system

corpus, i.e., Wikipedia dump, had some noise in the documents and the documents are in XML tagged format. So, first of all, the documents had to be preprocessed. The document structure is checked and reformatted according to the system requirements.

XML Parser. The corpus was in XML format. All the XML test data has been parsed before indexing using our XML Parser. The XML Parser extracts the Title of the document along with the paragraphs.

Noise Removal. The corpus has some noise as well as some special symbols that are not necessary for our system. The list of noise symbols and the special symbols is initially developed manually by looking at a number of documents and then the list is used to automatically remove such symbols from the documents. Some examples are """, "&", """, multiple spaces etc.

Document Indexing. After parsing the Wikipedia documents, they are indexed using Lucene, an open source indexer.

5.2 Tweets Parsing

After indexing has been done, the tweets had to be processed to retrieve relevant documents. Each tweet / topic was processed to identify the query words for submission to Lucene. The tweets processing steps are described below:

Stop Word Removal. In this step the tweet words are identified from the tweets. The Stop words and question words (what, when, where, which etc.) are removed from each tweet and the words remaining in the tweets after the removal of such words are identified as the query tokens. The stop word list used in the present work can be found at http://members.unine.ch/jacques.savoy/clef/.

Stemming. Query tokens may appear in inflected forms in the tweets. For English, standard Porter Stemming algorithm[6] has been used to stem the query tokens. After stemming all the query tokens, queries are formed with the stemmed query tokens.

5.3 Document Retrieval

After searching each query into the Lucene index, a set of retrieved documents in ranked order for each query is received.

First of all, all queries were fired with AND operator. If at least one document is retrieved using the query with AND operator then the query is removed from the query list and need not be searched again. The rest of the queries are fired again with OR operator. OR searching retrieves at least one document for each query. Now, the top ranked relevant document for each query is considered for Passage selection. Document retrieval is the most crucial part of this system. We take only the top ranked relevant document assuming that it is the most relevant document for the query or the tweet from which the query had been generated.

6 Automatic Summarization

6.1 Sentence Extraction

The document text is parsed and the parsed text is used to generate the summary. This module will take the parsed text of the documents as input, filter the input parsed text and extract all the sentences from the parsed text. So this module has two sub modules, Text Filterization and Sentence Extraction.

Text Filterization. The parsed text may content some junk or unrecognized character or symbol. First, these characters or symbols are identified and removed. The text in the query language are identified and extracted from the document using the Unicode

[6] `http://tartarus.org/~martin/PorterStemmer/java.txt`

character list, which has been collected from Wikipedia[7]. The symbols like dot (.), coma (,), single quote (‘), double quote (“), ‘!’, ‘?’ etc. are common for all languages, so these are also listed as symbols.

Sentence Extraction. In Sentence Extraction module, filtered parsed text has been parsed to identify and extract all sentences in the documents. Sentence identification and extraction is not an easy task for English document. As the sentence marker ‘.’ (dot) is not only used as a sentence marker, it has other uses also like decimal point and in abbreviations like Mr., Prof., U.S.A. etc. So it creates lot of ambiguity. A possible list of abbreviation had to created to minimize the ambiguity. Most of the times the end quotation (”) is placed wrongly at the end of the sentence like .”. These kinds of ambiguities are identified and removed to extract all the sentences from the document.

6.2 Key Term Extraction

Key Term Extraction module has three sub modules like Query Term, i.e., tweet term extraction, tweet text extraction and Title words extraction. All these three sub modules have been described in the following sections.

Query/Tweet Term Extraction. First the query generated from the tweet, is parsed using the Query Parsing module. In this Query Parsing module, the Named Entities (NE) are identified and tagged in the given query using the Stanford NER[8] engine.

Tweet’s Text extraction. Tweet’s texts are extracted and then all the keywords from the tweet text field are extracted to be used as more keywords. As these texts are provided along with the tweets, these are the most appropriate keywords regarding the tweets or topics.

Title Word Extraction. The title of the retrieved document is extracted and forwarded as input given to the Title Word Extraction module. After removing all the stop words from the title, the remaining tile words are extracted and used as the keywords in this system.

6.3 Top Sentence Identification

All the extracted sentences are now searched for the keywords, i.e., query terms, tweet’s text keywords and title words. Extracted sentences are given some weight according to search and ranked on the basis of the calculated weight. For this task this module has two sub modules: Weight Assigning and Sentence Ranking, which are described below.

---

[7] http://en.wikipedia.org/wiki/List_of_Unicode_characters
[8] http://www-nlp.stanford.edu/ner/

Weight Assigning. This sub module calculates the weights of each sentence in the document. There are three basic components in the sentence weight like query term dependent score, tweet's text keyword dependent score and title word dependent score. These three components are calculated and added to get the final weight of a sentence.

Query Term dependent score: Query term dependent score is the most important and relevant score for summary. Priority of this query dependent score is maximum. The query dependent scores are calculated using equation 1.

$$Q_s = \sum_{q=1}^{n_q} F_q \left(20 + \left(n_q - q + 1 \right) \left(\sum_p \left(1 - \frac{f_p^q - 1}{N_s} \right) \right) \times p \right) \tag{1}$$

where, Q_S is the query term dependent score of the sentence s, q is the no. of the query term, n_q is the total no. of query terms, f_p^q is the possession of the word which was matched with the query term q in the sentence s, N_s is the total no. of words in sentence s,

$$F_q = \begin{matrix} 0; \textit{ if query term } q \textit{ is not found} \\ 1; \textit{ if query term } q \textit{ is found} \end{matrix} \tag{2}$$

and

$$p = \begin{matrix} 5; \textit{ if query term is NE} \\ 3; \textit{ if query term is not NE} \end{matrix} \tag{3}$$

At the end of the equation 1, the calculated query term dependent score is multiplied by p to give the priority among all the scores. If the query term is NE and contained in a sentence then the weight of the matched sentence are multiplied by 5 as the value of p is 5, to give the highest priority, other wise it has been multiplied by 3 (as p=3 for non NE query terms).

Tweet's Text Keyword dependent score: Tweet's text keywords are provided along with the tweet. Hence, it should be relevant to the actual topic or concept of the tweet. So, this tweet's text keyword dependent score is also very important in the weight calculation of the sentences. Equation 4 has been use to calculate the tweet's text keyword dependent score.

$$K_s = \sum_{k=0}^{n_k} F_k \left(n_k - k + 1 \right) \left(\sum_p \left(1 - \frac{f_p^k - 1}{N_s} \right) \right) \times 2 \tag{4}$$

where, K_S is the tweet's text keyword dependent score of the sentence s, k is the number of the tweet's text keyword, n_k is the total number of tweet's text keyword,

f_p^k is the possession of the word which was matched with the tweet's text keyword k in the sentence s, N_s is the total no. of words in sentence s and

$$F_k = \begin{matrix} 0; \textit{ if tweet's text keyword k is not found} \sim \\ 1; \textit{ if tweet's text keyword k is found} \end{matrix} \quad (5)$$

At the end of the equation 3, the calculated title word dependent score is multiplied by 2 to give the second highest priority among all the scores.

Title Word dependent score: Title words are extracted from the title field of the top ranked retrieved document. A title word dependent score is also calculated for each sentence. Generally title words are also the much relevant words of the document. So the sentence containing any title words can be a relevant sentence of the main topic of the document. Title word dependent scores are calculated using equation 6.

$$T_s = \sum_{t=0}^{n_t} F_t\left(n_t - t + 1\right)\left(\sum_p \left(1 - \frac{f_p^t - 1}{N_s}\right)\right) \quad (6)$$

where, T_S is the title word dependent score of the sentence s, t is the no. of the title word, n_t is the total number of title words, f_p^t is the position of the word which matched with the title word t in the sentence s, N_s is the total number of words in sentence s and

$$F_t = \begin{matrix} 0; \textit{ if title word t is not found} \\ 1; \textit{ if title word t is found} \end{matrix} \quad (7)$$

After calculating all the above three scores the final weight of each sentence is calculated by simply adding all the three scores as mentioned in the equation 8.

$$W_s = Q_s + K_s + T_s \quad (8)$$

where, W_S is the final weight of the sentence s.

Sentence Ranking. After calculating weights of all the sentences in the document, sentences are sorted in descending order of their weight. In this process if any two or more than two sentences get equal weight, then they are sorted in the ascending order of their positional value, i.e., the sentence number in the document. So, this Sentence Ranking module provides the ranked sentences.

6.4 Summary Generation

This is the final and most critical module of this system. This module generates the Summary from the ranked sentences. As in [13] using equation 9, the module selects the ranked sentences subject to maximum length of the summary.

$$\sum_i l_i S_i < L \tag{9}$$

where l_i is the length (in no. of words) of sentence i, S_i is a binary variable representing the selection of sentence i for the summary and L (=500 words) is the maximum length of the summary.

Now, the selected sentences along with their weight are presented as the INEX output format.

7 Evaluation

7.1 Informative Content Evaluation

The organizers did the Informative Content evaluation [1] by selecting relevant passages. 50 topics were evaluated which was the pool of 14 654 sentences, 471 344 tokens, vocabulary of 59 020 words. Among them, 2801 sentences, 103889 tokens, vocabulary of 19037 words, are relevant. There are 8 topics with less than 500 relevant tokens. The evaluation measures of Information content divergences over {1,2,3,4gap}-grams (FRESA package) because it was too sensitive to smoothing on the qa-rels. So simple log difference of equation 10 was used:

$$\sum \log\left(\frac{\max\left(P(t/reference), P(t/summary)\right)}{\min\left(P(t/reference), P(t/summary)\right)}\right) \tag{10}$$

We have submitted two runs (ID46RJU_CSE_run1, ID46RJU_CSE_run2). The evaluation scores with the baseline system scores of informativeness by organizers of all topics are shown in the table 3 and evaluation scores of informativeness based on NYT textual content are show in table 4.

Table 3. The evaluation scores of Informativeness by organizers of all topics

Run	*unigram*	*bigram*	*with 2-gap*	*Average*	*Ranking*
ID46RJU_CSE_run1	0.056092	0.0876557	0.115557	0.0876168	0.115557
ID46RJU_CSE_run2	0.056122	0.0876816	0.11558	0.087643	0.11558
Baselinesum	0.0536912	0.0859148	0.114346	0.0858814	0.114346
Baselinemwt	0.0557855	0.0886043	0.117854	0.0887005	0.117854

Table 4. The evaluation scores of Informativeness based on NYT textual content

Run	*unigram*	*bigram*	*with 2-gap*	*Average*
ID46RJU_CSE_run1	0.0487001	0.080679	0.108948	0.0806496
ID46RJU_CSE_run2	0.0487017	0.0806804	0.10895	0.080651
Baselinesum	0.0460489	0.0781008	0.10646	0.0780837
Baselinemwt	0.0475077	0.0793851	0.10766	0.0793874

7.2 Readability Evaluation

For Readability evaluation [1] all passages in a summary have been evaluated according to Syntax (S), Anaphora (A), Redundancy (R) and Trash (T). If a passage contains a syntactic problem (bad segmentation for example) then it has been marked as Syntax (S) error. If a passage contains an unsolved anaphora then it has been marked as Anaphora (A) error. If a passage contains any redundant information, i.e., an information that have already been given in a previous passage then it has been marked as Redundancy (R) error. If a passage does not make any sense in its context (i.e., after reading the previous passages) then these passages must be considered as trashed, and readability of following passages must be assessed as if these passages were not present, so they were marked as Trash (T).

There are two readability metrics, Relaxed and Strict and they are defined as,

Relaxed metric: "Count as VALID if not Trash (T)";
Strict metric: "Count as VALID if not Trash (T) or Redundancy (R) or Anaphora (A) or Syntax (S)".

In both cases, the score is the average, normalized number of words in valid passages. Summary word numbers are normalized to 500 words each. Relaxed score can (rarely) be lower than strict score, as assessor can consider as "not trash" a passage with anaphora or syntax error. The readability evaluation scores are shown in the table 5. Participants are ranked according to this score. Our run 1's relaxed metric score is the best score and strict metric score is the 4th best score among all the runs from all the participants.

Table 5. The evaluation scores of Readability

Run	*Relaxed Metric Score*	*Strict Metric Score*
ID46RJU_CSE_run1	432.2000	347.9200
ID46RJU_CSE_run2	416.5294	330.1400
Baselinesum	447.3019	409.9434
Baselinemwt	137.8000	148.2222

8 Conclusion and Future Works

The question answering system has been developed as part of the participation in the Question Answering track of the INEX 2011 evaluation campaign. The overall system has been evaluated using the evaluation metrics provided as part of the QA track of INEX 2011. Considering that this is the first participation in the track, the evaluation results are satisfactory as readability scores are very high and in the relaxed metric we got the highest score of 432.2, which will really encourage us to continue work on it and participate in this track in future.

Future works will be motivated towards improving the performance of the system by concentrating on co-reference and anaphora resolution, named entity (NE) identification, multi-word identification, para phrasing, feature selection etc. In future, we will also try to use semantic similarity, which will increase our relevance score.

Acknowledgements. We acknowledge the support of the IFCPAR funded Indo-French project "An Advanced Platform for Question Answering Systems".

References

1. SanJuan, E., Moriceau, V., Tannier, X., Bellot, P., Mothe, J.: Overview of the INEX 2011 Question Answering Track (QA@INEX). In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 188–206. Springer, Heidelberg (2012)
2. Jezek, K., Steinberger, J.: Automatic Text summarization. In: Snasel, V. (ed.) Znalosti 2008, pp. 1–12. FIIT STU Brarislava, UstavInformatiky a softveroveho inzinierstva (2008) ISBN 978-80-227-2827-0
3. Erkan, G., Radev, D.R.: LexRank: Graph-based Centrality as Salience in Text Summarization. Journal of Artificial Intelligence Research 22, 457–479 (2004)
4. Hahn, U., Romacker, M.: The SYNDIKATE text Knowledge base generator. In: The First International Conference on Human Language Technology Research. Association for Computational Linguistics, ACM, Morristown, NJ (2001)
5. Kyoomarsi, F., Khosravi, H., Eslami, E., Dehkordy, P.K.: Optimizing Text Summarization Based on Fuzzy Logic. In: Seventh IEEE/ACIS International Conference on Computer and Information Science, pp. 347–352. IEEE, University of ShahidBahonar Kerman, UK (2008)
6. Bhaskar, P., Bandyopadhyay, S.: A Query Focused Multi Document Automatic Summarization. In: The 24th Pacific Asia Conference on Language, Information and Computation (PACLIC 24). Tohoku University, Sendai (2010)
7. Bhaskar, P., Bandyopadhyay, S.: A Query Focused Automatic Multi Document Summarizer. In: The International Conference on Natural Language Processing (ICON), pp. 241–250. IIT, Kharagpur (2010)
8. Rodrigo, A., Iglesias, J.P., Peñas, A., Garrido, G., Araujo, L.: A Question Answering System based on Information Retrieval and Validation. ResPubliQA (2010)
9. Schiffman, B., McKeown, K.R., Grishman, R., Allan, J.: Question Answering using Integrated Information Retrieval and Information Extraction. In: NAACL HLT, pp. 532–539 (2007)
10. Pakray, P., Bhaskar, P., Pal, S., Das, D., Bandyopadhyay, S., Gelbukh, A.: JU_CSE_TE: System Description QA@CLEF 2010 – ResPubliQA. In: Multiple Language Question Answering (MLQA 2010), CLEF 2010, Padua, Italy (2010)
11. Pakray, P., Bhaskar, P., Banerjee, S., Pal, B.C., Bandyopadhyay, S., Gelbukh, A.: A Hybrid Question Answering System based on Information Retrieval and Answer Validation. In: Question Answering for Machine Reading Evaluation (QA4MRE), CLEF 2011, Amsterdam (2011)
12. Tombros, A., Sanderson, M.: Advantages of Query Biased Summaries in Information Retrieval. In: SIGIR (1998)
13. Radev, D.R., Jing, H., Styś, M., Tam, D.: Centroid- based summarization of multiple documents. J. Information Processing and Management 40, 919–938 (2004)
14. Lin, C.Y., Hovy, E.H.: From Single to Multidocument Summarization: A Prototype System and its Evaluation. In: ACL, pp. 457–464 (2002)
15. Hardy, H., Shimizu, N., Strzalkowski, T., Ting, L., Wise, G.B., Zhang, X.: Cross-document summarization by concept classification. In: SIGIR, pp. 65–69 (2002)
16. Paladhi, S., Bandyopadhyay, S.: A Document Graph Based Query Focused Multi-Document Summarizer. In: The 2nd International Workshop on Cross Lingual Information Access (CLIA), pp. 55–62 (2008)

IRIT at INEX: Question Answering Task

Liana Ermakova and Josiane Mothe

Institut de Recherche en Informatique de Toulouse
118 Route de Narbonne, 31062 Toulouse Cedex 9, France
liana.ermakova.87@gmail.com, josiane.mothe@irit.fr

Abstract. In this paper we describe an approach for tweet contextualization developed in the context of the INEX question answering track. The task is to provide a context up to 500 words to a tweet. The summary should be an extract from the Wikipedia. Our approach is based on the index which includes not only lemmas, but also named entities (NE). Sentence retrieval is based on standard TF-IDF measure enriched by named entity recognition, part-of-speech (POS) weighting and smoothing from local context. The method has been ranked first in the INEX QA track according to content evaluation.

Keywords: Information retrieval, summarization, extraction, contextual information, smoothing, part of speech tagging, named entity.

1 Introduction

This paper describes the approach we developed at IRIT in the framework of the Question answering track (QA@INEX) of INEX (Initiative for the Evaluation of XML Retrieval). In 2011 the track aims at evaluating tweet contextualization in terms of relevance of the retrieved information to tweets and readability of the presented results. There are 132 tweets which include the title and the first sentence of a New York Times articles. The summary should be made of extracted relevant sentences from a local XML dump of English Wikipedia (April 2011), totally 3 217 015 non-empty pages. [1]

In the method we developed, firstly we parsed tweets and articles with Stanford CoreNLP[1] and we looked for documents similar to queries. We computed indices for all sentences. Then we searched for relevant sentences using standard TF-IDF measure enriched by named entity recognition, part-of-speech weighting and smoothing from local context.

The idea to contextualize short texts like microblogs or tweets is quite recent [2]. In [2] a tweet is mapped into a set of Wikipedia articles and a summary is not provided. Summaries are either "extracts", if they contain the most important sentences extracted from the original text, or "abstracts", if these sentences are re-written or paraphrased, generating a new text. [3] There exist two general approaches to text summarization, namely statistical methods and linguistic ones. Apparently, the first article on automated

[1] http://nlp.stanford.edu/software/corenlp.shtml

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 219–226, 2012.

summarization was published in 1958 [4]. H. P. Luhn proposed to order sentences by the number of the most frequent meaningful words. This approach was extended by taking into account sentence position in the text, key word and key phrase occurrence etc. [5] [6]. In case of a subject related summary, the query may be expanded e.g. by synonyms [7]. CORTEX combines such metrics as word frequency, overlap with query terms, entropy of the words, shape of text etc. [8]. LexRank underlies DISQ algorithm, where special attention is paid to redirects on the Wikipedia pages. Sentence importance may be computed from text energy matrix [10] [11] . Text corpora provide much useful information on features which should be kept in a summary, how long a text should be etc. [12] . Linguistic methods fall into several categories: (1) rule-based approaches, which may be combined with statistics [13] [12], (2) methods based on genre features, text structure etc. [5] [12] [14] and (3) methods based on syntax analysis [14].

The paper is organized as follows. Firstly, we describe our approach. Then evaluation results are provided. Future development description concludes the paper.

2 Method Description

2.1 Preprocessing

To contextualize tweets we first looked for the documents similar to the queries. For this stage, document retrieval was performed by the Terrier Information Retrieval Platform[2], an open-source search engine developed by the School of Computing Science, University of Glasgow. To this end we transformed tweets into to the format accepted by Terrier. We used the default settings for Terrier. We applied the BasicIndexer with the Porter stemmer [15] and the default list of stopwords. Text was converted to lowercase before parsing. There were no limits to the maximum number of tokens indexed for a document. We chose the Ponte and Croft's language model [16]. During document retrieval words with low IDF were ignored. For query expansion we used Rocchio algorithm with the parameter 0.4 [17]. The number of top-ranked documents to be considered in the pseudo relevance set was equal to 3 and the number of the highest weighted terms to be added to the original query was set to 10. A term was considered to be informative if it was found no less than in two documents[3].

The next stage was parsing of tweets and retrieved texts by Stanford CoreNLP developed by the Stanford Natural Language Processing Group. CoreNLP integrates such tools as POS tagger, named entity recognizer, parser and the co-reference resolution system[4]. It uses the Penn Treebank tag set [18]. In our approach, tweets were transformed into queries with POS tagging and recognized named entities. It allows taking into account different weights for different tokens within a query, e.g. NE are considered to be more important than common nouns; nouns are more significant than verbs; punctuation marks are not valuable, etc.

[2] http://terrier.org/
[3] http://terrier.org/docs/v2.2.1/properties.html
[4] http://nlp.stanford.edu/software/corenlp.shtml

2.2 Sentence Retrieval

The general idea of the proposed approach is to compute similarity between the query and sentences and to retrieve the most similar passages. To this end we used standard TF-IDF measure. We extended this approach by adding weight coefficients to POS, NE, headers, sentences from abstracts, and definitional sentences. Moreover sentence meaning depends on the context. Therefore we used an algorithm for smoothing from the local context which will be described later. The sentences were sorted by their similarity scores. The sentences with the highest score were added to the summary until the total number of words exceeds 500. In the implemented system there is a possibility to choose one of the following similarity measures: cosine, Dice and Jaccard similarity [17] . We took into account only lexical vocabulary overlap between a query and a document. However it is possible also to consider morphological and spelling variants, synonyms, hyperonyms, …

Different words should not have the same weight, e.g. usually it is better not to take into account stop-words. Our system provides several ways to assign score to words. The first option is to identify stop-words by frequency threshold. The second way is to assign different weights to different parts of speech. One can specify whether vector components should be multiplied by this POS rank, e.g. determiners have zero weight, proper names have the highest weight equal to 1.0, and nouns have greater weight than verbs, adjectives and adverbs. Another option gives a possibility to consider or not IDF.

NE comparison is hypothesized to be very efficient for contextualizing tweets about news. Therefore for each NE in queries we searched corresponding NE in the sentences. If it is found, the whole similarity measure is multiplied by NE coefficient computed by the formula:

$$NE_{COEF} = weight(NE) \times \frac{NE_{common}+1}{NE_{query}+1} \quad (1)$$

where $weight(NE)$ is floating point parameter given by a user (by default it is equal to 1.0), NE_{common} is the number of NE appearing in both query and sentence, NE_{query} is the number of NE appearing in the query. We used Laplace smoothing to NE by adding one to the numerator and the denominator. The sentence may not contain a NE from the query and it can be still relevant. However, if smoothing is not performed the coefficient will be zero. NE recognition is performed by Stanford CoreNLP. We considered only the exact matches of NE. Synonyms were not identified. However, it may be done later applying WordNet, which includes major NE.

We consider that *Headers*, *labels*, …. should not be taken into account since they are not "good" sentences for summarization. Therefore we assign them lower weights. Stanford parser allows making distinction between auxiliary verbs and main verbs, personal and impersonal verb forms. We assumed that such kinds of sentences do not have personal verbs. One of the settings allows assigning weights to sentences without personal verb forms. By default this parameter is equal to 0. Sentences with personal verb forms have the weight equal to 1.0. It is possible to give smaller weights to sections than to abstracts. By default we assume that sections have the weight equal to

0.8 and for abstracts this parameter is 1.0. We assumed that definitional sentences are extremely important to contextualizing task. Therefore they should have higher weights. We took into account only definitions of NE by applying the following linguistic pattern: $< NE >< Be_{pers} >< NounPhrase >$, where Be_{pers} is a personal form of the verb *to be*. Noun phrase recognition is also performed by Stanford parser. We considered only sentences that occurred in abstracts since they contain more general and condensed information and usually include definitions in the first sentence. However, the number of extracted definitions was quite small and therefore we did not use them in our runs.

Since sentences are much smaller than documents, general IR systems provide worse results to sentence retrieval. Moreover, document retrieval systems are based on the assumption that relevant documents are about the query. However this is not enough for sentence retrieval, e.g. in QA systems the sentence containing the answer is much more relevant that the sentence which is about the subject. General approach to document IR is underlined by TF-IDF measure. In contrast, usually the number of each query term in a sentence is no more than one [19] . Traditionally, sentences are smoothed by the entire collection, but there exist another approach namely smoothing from local context [19]. This method assigns the same weight to all sentences from the context. In contrast, we assume that the importance of the context reduces as the distance increases. So, the nearest sentences should produce more effect on the target sentence sense than others. For sentences with the distance greater than k this coefficient is zero. The total of all weights should be equal to one. The system allows taking into account k neighboring sentences with the weights depending on their remoteness from the target sentence. In this case the total target sentence score R_t is a weighted sum of scores of neighboring sentences r_i and the target sentence r_0 itself:

$$R_t = \sum_{i=-k}^{k} w_i \times r_i \tag{2}$$

$$w_i = \begin{cases} \frac{1-w_t}{k+1} \times \frac{k-|i|}{k}, 0 < |i| \leq k \\ w_t, i = 0 \\ 0, |i| > k \end{cases} \tag{3}$$

$$\sum_{i=-k}^{k} w_i = 1 \tag{4}$$

where w_t is a target sentence weight set by a user, w_i are weights of the sentences from k context. The weights become smaller as the remoteness increases. If the sentence number in left or right context is less than k, their weights are added to the target sentence weight w_t. This allows keeping the sum equal to one. By default, $k = 1$, target sentence weight is equal to 0.8.

3 Evaluation

For the first run we used default settings (default), namely: NE were considered with a coefficient 1.0; abstract had weight equal to 1.0, sections had score 0.8; headers,

labels, … were not taken into account; we removed stop-words; cosine similarity was applied; POS were ranked; each term frequency was multiplied by IDF. In the second run we changed the similarity measure to Dice similarity (07_2_07_1_dice). The section weight was reduced to 0.7. The context was extended to two sentences in each direction and the target sentence weight was equal to 0.7. For NE we kept the weight equal to 1.0. In the third run we applied Jaccard similarity measure (05_2_07_1_jac) and we set the weight to sections equal to 0.5.

Evaluation was performed manually by conference organizers [1] . Passages were judged as relevant or not without context. The summaries submitted by participants were compared to each other, to the baseline summary made of sentences (baselinesum) and to the key terms (baselinemwt). The baseline system was based on Indri index without stop word list and stemming (language model). Part of speech tagging was performed by TreeTagger. Summarization algorithm was TermWatch [1] .

Since the task was to provide the context to the tweets and therefore found passages should be somehow similar to the original New York Times articles, firstly, obtained results were compared with them. Then, the overlap with relevant passages evaluated manually was computed. N-gram distribution of summaries, namely unigram distribution, bigram distribution and bigram distribution with two word gap, was compared with those from relevant passages and New York Times articles [1] . So, the comparison with two different relevant collections was performed.

In order to evaluate the informative level of summaries the simple log difference was used, since it is less sensitive to smoothing on the given collection than the Kullback-Leibler divergence [1]. **Table 1** presents the comparison of baseline systems and the submitted runs with regards to New York Times articles. All three runs are ranked higher than baseline systems. The best result is given by 05_2_07_1_jac.

Table 1. Log difference to New York Times articles

Ranking	Unigram	Bigram	With 2-gap	Average	Run
0.104925	0.0447	0.076644	0.104925	0.076629	05_2_07_1_jac
0.104933	0.044728	0.076659	0.104933	0.076646	07_2_07_1_dice
0.104937	0.044739	0.076668	0.104937	0.076653	default
0.10646	0.046049	0.078101	0.10646	0.078084	Baselinesum
0.10766	0.047508	0.079385	0.10766	0.079387	Baselinemwt

Table 2 provides comparison referring to the pool of relevant sentences. According to these evaluations, all runs we submitted are more relevant than baselines. However, the best results were provided by the run with the default settings. We think that the opposite evaluation results obtained for NYT and the pool of relevant passages from the Wikipedia may be explained by the different language models of these collections. The pool of the relevant sentences from the Wikipedia contained 103 889 tokens, which gave a vocabulary of 19 037 words, and the original news articles with a vocabulary of 26 481 words contained 154 355 tokens [1]. So, the average word frequency differs for 9%. Moreover, these two corpora have different genres and consequently

different structure. In our approach NE matching was extremely important and therefore we preferred to select sentences with proper nouns, but not pronouns and other type of references (e.g. American President instead of Barack Obama). In a news article authors try not to repeat themselves and they substitute NE by other words. Since relevant passages were selected without context, the majority of them tended to contain NE. Thus, there exist two main explanations of the opposite ranks: different language models of the collections and the pool peculiarities.

Table 2. Log difference with the set of relevant passages

Ranking	Unigram	Bigram	With 2-gap	Average	Run
0.105506	0.048639	0.07867	0.105506	0.078697	default
0.105747	0.048781	0.078857	0.105747	0.07889	07_2_07_1_dice
0.106195	0.049083	0.079249	0.106195	0.079277	05_2_07_1_jac
0.114346	0.053691	0.085915	0.114346	0.085881	Baselinesum
0.117854	0.055786	0.088604	0.117854	0.088701	Baselinemwt

The readability evaluation was also performed manually. Assessors should indicate if a passage contained one of the following drawbacks: syntactical problems (e.g. bad segmentation), unresolved anaphora, redundant information (that is to say, the information is already mentioned) or the passage is meaningless in the given context (trash). The total score was the average normalized number of words in valid passages [1]. Though the system showed the best results according the relevance judgment, it was worse than the baseline in terms of readability. The major drawback was unresolved anaphora. Trash passages refer not only to readability, but also to relevance. Therefore relevance improvement and sentence reordering may solve this problem.

4 Conclusion

In this article we describe a method to tweet contextualization on the basis of the local Wikipedia dump. Firstly, we looked for relevant Wikipedia pages using the search engine Terrier. Secondly, the input tweets and the found documents were parsed by Stanford CoreNLP. After that, a new index for sentences was constructed. It includes not only stems but also NE. Then we searched for relevant sentences. To this end similarity between the query and sentences was computed using an extended TF-IDF measure. We enhance the basic approach by adding weight coefficients to POS, NE, headers, sentences from abstracts, and definitional sentences. Moreover, the algorithm for smoothing from local context is provided. We assume that the importance of the context depends on the remoteness from the target sentence. So, the nearest sentences should produce more effect on the target sentence sense than others. Remote sentences (with the distance greater than k) should not be taken into account. The sentences with the highest score are added to the summary until the total number of words exceeds 500.

Relevance evaluation provides evidence that the approach is better than the baselines underlined by language model. All runs are closer to the original New York Times articles and contain more relevant passages. The run with default settings is the most relevant. However, the run based on Jaccard coefficient and reduced weight for sections gave results more similar to the original New York Times articles. This can be explained by different language models and by the features of the pool of the relevant passages. In terms of relevance the developed system was the first among 11 systems, but in terms of readability it was only the third [1].

Future work includes solving anaphora problems, sentence ordering, additional features selection and applying different similarity measure, e.g. expanded by synonyms and relations from WordNet. This should increase relevance as well as readability.

References

1. SanJuan, E., Moriceau, V., Tannier, X., Bellot, P., Mothe, J.: Overview of the INEX 2011 Question Answering Track (QA@INEX). In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 188–206. Springer, Heidelberg (2012)
2. Meij, E., Weerkamp, W., Rijke, M.: Adding Semantics to Microblog Posts. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (2012)
3. Vivaldi, J., Cunha, I., Ramırez, J.: The REG summarization system at QA@INEX track 2010 (2010)
4. Luhn, H.: The automatic creation of literature abstracts. IBM Journal of Research and Development, 159–165 (April 1958)
5. Seki, Y.: Automatic Summarization Focusing on Document Genre and Text Structure. ACM SIGIR Forum 39(1), 65–67 (2005)
6. Erkan, G., Radev, D.: LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. Journal of Artificial Intelligence Research 22, 457–479 (2004)
7. Soriano-Morales, E.-P., Medina-Urrea, A., Sierra, G., Mendez-Cruz, C.-F.: The GIL-UNAM-3 summarizer: an experiment in the track QA@INEX 2010 (2010)
8. Torres-Moreno, J.-M., Gagnon, M.: The Cortex Automatic Summarization System at the QA@INEX Track 2010. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 290–294. Springer, Heidelberg (2011)
9. Cabrera-Diego, L., Molina, A., Sierra, G.: A Dynamic Indexing Summarizer at the QA@INEX 2011 track. In: INEX 2011 Workshop Pre-Proceedings, pp. 154–159 (2011)
10. Linhares, A., Velazquez, P.: Using Textual Energy (Enertex) at QA@INEX track 2010 (2010)
11. Torres-Moreno, J.-M., Velazquez-Morales, P., Gagnon, M.: The Cortex and Enertex summarization systems at the QA@INEX track 2011, pp. 196–205 (2011)
12. Lin, C.-Y., Hovy, E.: Identifying Topics by Position. In: Proceedings of the Fifth Conference on Applied Natural Language Processing, pp. 283–290 (1997)
13. Lin, C.-Y.: Assembly of Topic Extraction Modules in SUMMARIST. In: AAAI Spring Symposium on Intelligent Text Summarisation (1998)
14. Barzilay, R., McKeown, K., Elhadad, M.: Information fusion in the context of multi-document summarization. In: ACL 1999 Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics, pp. 550–557 (1999)

15. Porter, M.: An algorithm for suffix stripping. In: Readings in Information Retrieval. Morgan Kaufmann Publishers Inc., San Francisco (1997)
16. Ponte, J., Croft, W.: A language modeling approach to information retrieval. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1998)
17. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
18. Marcus, M., Santorini, B., Marcinkiewicz, M.: Building a large annotated corpus of English: the Penn Treebank. Computational Linguistics 19(2) (1993)
19. Murdock, V.: Aspects of Sentence Retrieval. Dissertation (2006)

A Graph-Based Summarization System at QA@INEX Track 2011

Ana Lilia Laureano-Cruces[1,2] and Javier Ramírez-Rodríguez[1,2]

[1] Universidad Autónoma Metropolitana-Azcapotzalco
Mexico
[2] LIA, Université d'Avignon et des Pays de Vaucluse
France
{clc,jararo}@correo.azc.uam.mx
http://lia.univ-avignon.fr/

Abstract. In this paper we use REG, a graph-based system to study a fundamental problem of Natural Language Processing: the automatic summarization of documents. The algorithm models a document as a graph, to obtain weighted sentences. We applied this approach to the INEX@QA 2011 task (question-answering). We have extracted the title and some key or related words according to two people from the queries, in order to recover 50 documents from english wikipedia. Using this strategy, REG obtained good results with the automatic evaluation system FRESA.

Keywords: Automatic Summarization System, Question-Answering System, Graph-Based.

1 Introduction

Nowadays automatic summarization using graph-based ranking algorithms has drawn much attention in recent years from the computer science community and has been widely used [4, 17, 13, 10]. According to [8] a summary is "a condensed version of a source document having a recognizable genre and a very specific purpose: to give the reader an exact and concise idea of the contents of the source". Summaries has two main approaches "extraction", if they contain the most important sentences extracted from the original text, for example [15, 10] and "abstraction", if these sentences are re-written or paraphrased from the source, generating a new text [5–7]. Most of the automatic summarization systems are extractive. These systems have been used in different domains [1–3, 16]. One of the tasks where these extractive summarization systems could help is question-answering. The objective of the INEX@QA 2011 track is contextualizing tweets, i.e. answering questions of the form "what is this tweet about?" using a recent cleaned dump of the Wikipedia. In this task is where automatic summarization systems is used. Each of the selected 132 topics for 2011 includes the title and the first sentence of a New York Times paper that were twitted at least two months after the wikipedia. The expected answers are automatic summaries of less than 500 words exclusively made of aggregated passages extracted

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 227–231, 2012.

from the Wikipedia corpus. The evaluation of the answers will be automatic, using the automatic evaluation system FRESA [11, 12, 9], and manual (evaluating syntactic incoherence, unsolved anaphora, redundancy, etc.). To carry out this task, we have decided to use REG [13, 14], an automatic summarization system based on graphs.

The rest of this paper is organized as follows. In the next section we show REG, the graph-based summarization system we have used for the experiments. In Section 3 we explain how we have carried out the terms extraction of the queries. In Section 4 we present the results. In Section 5, some conclusions are given.

2 The Graph-Based System

REG presented in [13, 14] is an Enhanced Graph Summarizer (REG) for extract summarization, using a graph approach. The strategy of this system has two main stages: a) to carry out an adequate representation of the document and b) to give a weight to each sentence of the document. In the first stage, the system makes a vectorial representation of the document. In the second stage, the system uses a greedy optimization algorithm of traversal graph to obtain a desired number of relevant sentences, all if necessary. The summary generation is done with the concatenation of the most relevant sentences (previously scored in the optimization stage).

REG algorithm contains three modules. The first one carries out the vectorial transformation of the text with filtering, lemmatization/stemming and normalization processes. The second one applies the greedy algorithm and calculates the adjacency matrix. We obtain the score of the sentences directly from the algorithm. Therefore, a desired number of sentences with more score will be selected by the greedy algorithm as the most relevant. Finally, the third module generates the summary, selecting and concatenating the desired number of relevant sentences. The first and second modules use CORTEX [15], a system that carries out an unsupervised extraction of the relevant sentences of a document using several numerical measures and a decision algorithm.

3 Treatment of Queries

The 132 queries obtained from the topics of the New York Times were processed by two persons. The first one has chosen in addition to the title the words of the phrase are, from her point of view, key words. The second has been considered in addition to the title, words related to the title, for example, for the query "Largest Holder of US Debt" we consider the terms: debit, china and USA. The 132 queries were processed by the perl program of inex and sent to INDRI to subsequently obtain 50 documents per query from english wikipedia, of which REG obtained a summary of between 500 and 600 words of each. A random sample was selected from 6 queries of each form of generating it and that were evaluated with the FRESA tool.

4 Results

In this study, we used the document sets made available during the —Initiative for the Evaluation of XML retrieval (INEX) 2011[1], in particular on the INEX 2011 QA Track (QA@INEX). These sets of documents where provided by the search engine Indri.[2] REG has produced the same number of summaries of approximately 550 words each.

To evaluate the efficiency of REG over the INEX@QA corpus, we have used the FRESA package.

Table 1 shows an example of the best result obtained by REG using 50 documents as input. The query that the summary should answer in this case was the number 2011024. This table presents REG results in comparison with an intelligent baseline (Baseline summary), and two simple baselines, that is, summaries including random n-grams (Random unigram) and 5-grams (Random 5-gram). We observe that our system is always better than these baselines. Then we present the average of the random sample for each one of the people and the Baseline summary, finding that REG is acceptable.

Table 1. Example of REG results using 50 documents as input

Distribution type	unigram	bigram	with 2-gap	Average
Baseline summary	20.5418	27.5864	27.6669	25.2650
Empty baseline	29.3267	36.8774	36.9097	34.3712
Random unigram	19.6240	27.2631	27.2464	24.7112
Random 5-gram	19.0643	26.1404	26.3386	23.8478
Submitted summary	20.2178	27.2809	27.3687	24.9558

Table 2. Averge of REG results for each person to generate the query

Form	Person 1	Person 2	Baseline summary
Average	33.6984	33.9505	32.8402

5 Conclusions

We have presented the REG summarization system, an extractive summarization algorithm that models a document as a graph, to obtain weighted sentences. We applied this approach to the INEX@QA 2011 task, extracting the terms from the queries, in order to obtain a list of terms related with the main topic of the question.

[1] https://inex.mmci.uni-saarland.de/

[2] Indri is a search engine from the Lemur project, a cooperative work between the University of Massachusetts and Carnegie Mellon University in order to build language modelling information retrieval tools: http://www.lemurproject.org/indri/

Our experiments have shown that the system is always better than the two simple baselines, but in comparison with the first one the performance is variable. We think this is due to the fact that some queries are long and they have several terms we could extract, but there are some queries that are very short and the term extraction is not possible or very limited. Nevertheless, we consider that, over the INEX-2011 corpus, REG obtained good results in the automatic evaluations, but now it is necessary to wait for the human evaluation and the evaluation of other systems to compare with.

References

1. Abraços, J., Lopes, G.: Statistical methods for retrieving most significant paragraphs in newspaper articles. In: Proceedings of the ACL/EACL 1997 Workshop on Intelligent Scalable Text Summarization, Madrid, pp. 51–57 (1997)
2. da Cunha, I., Wanner, L., Cabré, M.T.: Summarization of specialized discourse: The case of medical articles in Spanish. Terminology 13(2), 249–286 (2007)
3. Farzindar, A., Lapalme, G., Desclés, J.P.: Résumé de textes juridiques par identification de leur structure thématique. Traitement Automatique des Langues 45(1), 39–64 (2004)
4. Mihalcea, R.: Graph-based ranking algorithms for sentence extraction, applied to text summarization. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Lingusitics (ACL 2004), companion volume, Barcelona (2004)
5. Ono, K., Sumita, K., Miike, S.: Abstract generation based on rhetorical structure extraction. In: Proceedings of the International Conference on Computational Linguistics, Kyoto, pp. 344–348 (1994)
6. Paice, C.D.: Constructing literature abstracts by computer: Techniques and prospects. Information Processing and Management 26, 171–186 (1990)
7. Radev, D.: Language Reuse and Regeneration: Generating Natural Language Summaries from Multiple On-Line Sources. New York, Columbia University [PhD Thesis] (1999)
8. Saggion, H., Lapalme, G.: Generating Indicative-Informative Summaries with SumUM. Computational Linguistics 28(4), 497–526 (2002)
9. Saggion, H., Torres-Moreno, J.-M., da Cunha, I., SanJuan, E., Velázquez-Morales, P.: Multilingual Summarization Evaluation without Human Models. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), Pekin (2010)
10. Torres-Moreno, J.-M.: Résumé automatique de documents: Une approche statistique. Hermés-Lavoisier, Paris (2011)
11. Torres-Moreno, J.-M., Saggion, H., da Cunha, I., Velázquez-Morales, P., SanJuan, E.: Summary Evaluation With and Without References. Polibitis: Research Journal on Computer Science and Computer Engineering with Applications 42, 13–19 (2010)
12. Torres-Moreno, J.-M., Saggion, H., da Cunha, I., Velázquez-Morales, P., SanJuan, E.: Evaluation automatique de résumés avec et sans référence. In: Proceedings of the 17e Conférence sur le Traitement Automatique des Langues Naturelles (TALN), Montreal (2010)
13. Torres-Moreno, J.-M., Ramírez, J.: REG: un algorithme glouton appliqué au résumé automatique de texte. In: JADT 2010, Roma (2010)

14. Torres-Moreno, J.-M., Ramírez, J., da Cunha, I.: Un resumeur a base de graphes, indépendant de la langue. In: Workshop African HLT 2010, Djibouti (2010)
15. Torres-Moreno, J.M., Velázquez-Morales, P., Meunier, J.G.: Condensés de textes par des méthodes numériques. In: Proceedings of the 6th International Conference on the Statistical Analysis of Textual Data (JADT), St. Malo, pp. 723–734 (2002)
16. Vivaldi, J., da Cunha, I., Torres-Moreno, J.M., Velázquez, P.: Automatic Summarization Using Terminological and Semantic Resources. In: Proceedings of 7th International Conference on Language Resources and Evaluation (LREC 2010), Valletta (2010)
17. Sidiropoulos, A., Manolopoulos, Y.: Generalized comparison of graph-based ranking algorithms for publications and authors. The Journal of Systems and Software 79, 1679–1700 (2006)

LIA at the INEX 2011 QA Track: Querying and Summarizing with XML

Killian Janod and Olivier Mistral

LIA, Université d Avignon et des Pays de Vaucluse, France
{killian.janod,olivier.mistral}@etd.univ-avignon.fr

Abstract. This paper describes our participation in the INEX 2011 Question Answering track. There several ways to answer to this track, the aim is using XML meta-data in the summarizing process. First, words occurrence probabilities are computed using a fielded language modeling approach within different document fields. Then summaries are made with sentences extracted from previously retrieved documents. These sentences are ranked according to different XML-related features as well as a term-frequency based measure.

Keywords: XML, INEX, QA track, Indri.

1 Introduction

The INEX Question Answering track is about contextualizing tweets, therefore sentences. To answer this track a summarizer was developed which selects automatically the most relevant documents in Wikipedia's corpus and summarize them to do the tweet contextualisation. XML's content plays a very important role in this summarizing system. The aim is to make a hybrid summarizer which combines probabilistic metrics and XML.

Therefore focus is given to the improvement of requests with XML and the ability of using XML in the summarizing process.

2 Processing

2.1 Improved Queries with XML

Using Indri engine[1], a Query Likelihood model was built and jointly used with XML by making improved query using XMLs tags from the initial Wikipedia's corpus:

$$\#weight(n_j\ \#1(U_j).title\ n_i\ \#1(U_i).a\ n_i\ \#1(U_i).s\ \#combine(text\ uni-grams))$$

Where U_i: uni-grams title at i position, n_i: number of iterations of U_i, i from 1 to final uni-grams position and j from 1 to 2.

Improved queries allowed us to reduce by 5 the number of documents returned from Wikipedia, while increasing the relevance of these from five documents judged relevant by human on ten returned to nine on ten.

[1] http://lemurproject.org/indri/

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 232–234, 2012.

2.2 Summarizing and XML

At First, every sentences are scored depending on tags which included them. Each tag has a value between 1 and 10 depending on how important the tag is (For example, *title* is 10). A score was created by the sum of each tag's value. Then the score made this way and a score given by the baseline summarizer are normalized and included into the CORTEX's decision making algorithm. CORTEX [5] [4] is a text summarizer based on probabilistic methods. Our decision making algorithm is defined as follows:[2]

$$Avg_{sup} = \sum_{v=1,\ score_{v,j}>0.5}^{M} (score_{j,v} - 0.5) \qquad Avg_{inf} = \sum_{v=1,\ score_{v,j}<0.5}^{M} (0.5 - score_{j,v}) \tag{1}$$

If $Avg_{sup} > Avg_{inf}$ then:

$$FinalMark_j = 0.5 + \frac{Avg_{sup}}{M} \qquad otherwise \qquad FinalMark_j = 0.5 - \frac{Avg_{inf}}{M} \tag{2}$$

This algorithm returns a single final mark for each sentence from each usual metric and the XML one. Selecting the top scored sentences to reach 500 words, makes a summary ready to be assessed with FRESA [1].

3 Runs

Three runs were submitted, each one was evaluated twice. Once, compared to the New York Times article linked to the tweet and then evaluated by organizers. With the first evaluation, runs are clearly under the baseline_sum and slightly under the baseline_wt. According to the second evaluation runs are just better than the baseline_wt. Those results tells that the method used is really close to the baseline and the use of XML made in the runs neither improved or reduced the results of the classical approach. These results can be explain because the decision algorithm used is able to give more weight to meaningful metric. That means when the XML made score does not bring useful information the final result is close to the baseline summarizer.

Focusing on the sentence's depth, in the XML tree, involves the hypothesis that the deeper a sentence is, the more specific it is and a specific sentence which is about the subject must be more relevant than a general one. Low ranks on our runs can not affirm this assumption. Furthermore using a sentence as unity is the summarizer, when a XML node is used in the ranking process may give weight to sentences irrelevant which are on the same branch than useful sentences. However, there are still slightly better performance on the run using the XML dept which proves that the use of weighted XML tags is not completely meaningless.

[2] Where M is the number of metrics used and j is the sentence we are scoring.

Table 1. LIA Questions Answering runs results [3] [2]

Position	Ranking	Run	uni-grams	bi-grams	with 2-gap	Average
Based on New York Times textual content						
06/25	0.10646	Baseline_sum	0.0460489	0.0781008	0.10646	0.0780837
15/25	0.10766	Baseline_wt	0.0475077	0.0793851	0.10766	0.0793874
16/25	0.10769	LIA Run1	0.0472831	0.0793437	0.10769	0.0793193
18/25	0.107969	LIA Run3	0.0475984	0.0796384	0.107969	0.0796139
By organizers All topics						
10/25	0.114346	Baseline_sum	0.0536912	0.0859148	0.114346	0.0858814
16/25	0.117158	LIA Run3	0.0564556	0.0886844	0.117158	0.0886665
19/25	0.117854	Baseline_wt	0.0557855	0.0886043	0.117854	0.0887005
20/25	0.118016	LIA Run1	0.0566099	0.0892074	0.118016	0.0892034

4 Conclusion

This paper presented an experimentation exploiting the element's depth in the XML tree during the baseline summarizing process and showed that it was not efficient. In future works, we plan to improve this hybrid summarizer with a probabilistic method able to determine for each document the best score to give for each XML tags. In order to have a relevant rank based on XML and a consistent summarizing process.

References

1. Saggion, H., Torres-Moreno, J.M., da Cunha, I., SanJuan, E.: Multilingual summarization evaluation without human models. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING 2010), Beijing, Chine, pp. 1059–1067. ACL (2010)
2. Torres-Moreno, J.M., Saggion, H., da Cunha, I., SanJuan, E.: Summary Evaluation With and Without References. Polibits: Research Journal on Computer Science and Computer Engineering with Applications 42, 13–19 (2010)
3. Torres-Moreno, J.M., Saggion, H., da Cunha, I., Velazquez-Morales, P., SanJuan, E.: Evaluation automatique de résumés avec et sans réferences. In: Proceedings de la Conference Traitement Automatique des Langagues Naturelles (TALN 2010), Montral, QC, Canada, ATALA (2010)
4. Torres-Moreno, J.M., St-Onge, P.L., Gagnon, M., El-Bèze, M., Bellot, P.: Automatic summarization system coupled with a question-answering system (qaas). CoRR, abs/0905.2990 (2009)
5. Torres-Moreno, J.M., Velázquez-Morales, P., Meunier, J.G.: Cortex: un algorithme pour la condensation automatique de textes. In: ARCo, vol. 2, p. 365 (2001)

Flesch and Dale-Chall Readability Measures for INEX 2011 Question-Answering Track

Jade Tavernier and Patrice Bellot

LSIS - Aix-Marseille University / CNRS
patrice.bellot@lsis.org, jade.tavernier@gmail.com

Abstract. For INEX 2011 QA track, we wanted to measure the impact of two generic measures of readability in the selection of sentences related to topics. This is a step towards adaptive information retrieval approaches that take into account the reading skills of users and their level of expertise. We show that Flesch and Dale-Chall measures do not allow to filter sentences for obtaining a satisfactory readability level for INEX QA 2011 track and that the corresponding scores are not correlated to human assessment.

Keywords: readability measures, Flesch, Dale-Chall, filtering.

1 Introduction

Adapting retrieval to users with limited reading skills is still an open problem. This may include people with language disorders (eg. dyslexia makes reading slow and complex) as well as those not proficient enough in the language of a document or that have to read a content whose necessary expertise for understanding is too high. Adaptation of information retrieval with the consideration of individual reading / understanding performance may be a major concern for modern societies where citizens learn online, without human mediation. The issue of measuring the readability of a text is important in many other areas. For example, it allows to estimate the level of difficulty of a text when a child learns reading or learns a foreign language.

Many studies have examined the concept of relevance and defined it according to extra-linguistic and contextual parameters that are not explicit in a query [1]. Indeed, users do not specify their level of expertise in the queries (how would they?) nor their average speed reading and difficulties. Moreover, the common retrieval models allow ordering documents according to how much information they convey vis-a-vis what the user expressed in its query, taking into account, possibly, a personal profile (previous queries and consultations) or the ones of some other users who entered a similar request. This is essentially an *informational* relevance that may be adequate but that does not take into account specefic needs such as the level of expertise of the user. Meet such a need for customization requires to define new information retrieval measures taking into account readability and relevance of a text and, for example, providing users with simplest documents first.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 235–246, 2012.

For INEX 2011 QA track, we wanted to measure the impact of some classical measures of readability in the selection of sentences answering to topics. We considered the readability as an indicator of cohesion of the summaries we extracted from documents. Related work concerns the estimation of linguistic quality, or the fluency, of a text such as employed in some automatic summarization systems [2,3]. However, we are more interested in detection of text easier to understand than text well written. There are many challenges: determining a good measure of readability for the task, achieving a good balance between a selection of phrases according to their informational relevance (quantity of new information) and their readability.

In Section 2, we present some related work and classical measures of readability. Then, we present some machine learning approaches for adapting readability to users although we have not been able to experiment them yet. In Section 3, we describe the experiments we realized and the runs we submitted. In Section 4, we analyze our runs and scores computed over all the runs submitted by INEX 2011 QA participants.

2 Relevance and Readability

The purpose of an information retrieval system is to provide relevant documents to the user in relation to its information need (topic or *query*). The notion of relevance has been the subject of many studies that attempt to identify the concepts to which it refers [4]. S. Mizzaro proposes a definition of relevance that encompasses several dimensions [1] : the need for information, broken down into real need, perceived need, expressed need, and a query written in a formal query language, the compounds : the information, the task and its context, the time required to retrieve the information, and the granularity of the answers (comprehensive document, segments of documents or precise answer).

The type of documents, the level of detail, the date of writing are so many criteria that can be included in retrieval models, along with the popularity (PageRank) and the authority. The integration of the criterion of readability also requires to reformulate what is a relevant document. Adapting the retrieval process to the reading skills of the user is equivalent to considering the readability as a continuous variable we seek to maximize while selecting only the best documents verifying the other criteria of relevance. Readability can be seen as an additional criterion in the retrieval model or as a new filtering step.

2.1 Classical Measures for Estimating Readability

W.H. Dubay notes that more than 200 readability formulas can be found in the literature [5]. They exploit linguistic, syntactic, semantic or pragmatic clues. Coh-Metrix[1] combines more than 50 measures to calculate the coherence of texts [6]. Some measures of readability still used date back fifty years. For the English

[1] http://cohmetrix.memphis.edu/cohmetrixpr/index.html

language, the most common measures of readability are : FOG [7] (formula 1), SMOG [8] (formula 2) and Flesch [9] (formula 3) which is proposed by software such as Microsoft Word in its statistical tools set. Derivatives of these formulas can also be found such as formula 4 described in [10].

We report the most classic formula which are the ones we experimented. Let d a document (or a sentence), $L(d)$ the readability of d. Let ASL be the average length of sentences (number of words), ASW the average number of syllables per word, MON the frequency of monosyllabic words in d and POL the number of multisyllabic words in the beginning, in the middle and at the end of d.

$$L_{FOG}(d) = 3,068 + 0,877 \times ASL + 0,984 \times MON \tag{1}$$

$$L_{SMOG}(d) = 3 + \sqrt{POL} \tag{2}$$

$$L_{Flesch}(d) = 206,835 - 1,015 \times ASL - 84,6 \times ASW \tag{3}$$

$$L_{Flesch-Kincaid}(d) = 0,39 \times ASL + 11,8 \times ASW - 15,59 \tag{4}$$

Some other measures are reported in [11] that employ closed list of words classified according to their difficulty such as the Caroll-Davies-Richman list [12] and the Dale & O'Rourke dictionary [13] : Lexile [14], Dale-Chall (formula 5) [15] and Fry [16] for very short texts.

$$L_{Dale-Chall}(d) = (0.1579 \times DS) + (0.0496 \times ASL) + 3.6365 \tag{5}$$

where DS is Dale-Chall Lexicon-based score, or % of words not in their dictionary of 3,000 common words, and ASW the average number of syllables per word. Note that many other factors, particularly sensitive for visually impaired people, could be considered: spacing and font size, contrast, paragraph alignment, the presence of images, physical structure of documents... These criteria are taken into account for Web pages by *Web Accessibility Initiative* (WAI).

2.2 Machine Learning for Readability Measures Adapted to Users

The availability of large corpora and the organization of large-scale experiments have enabled the development of new measures of readability that does not involve numerical parameters selected manually. They are used to better reflect certain contexts than do generic measures such as those defined above. L. Si and J. Callan have proposed to combine linguistic criteria and surface analysis (unigram language model) [10]. They conclude that the rarity of words (estimated through a generic unigram model) and the length of sentences are meaningful criteria for estimating the difficulty of reading a web page but not the number of syllables per word. This is consistent with psycholinguistic studies on the reading process: a long word is not necessarily more difficult to recognize if it has already been met by the reader. This is also consistent with the analysis we conducted

and that shows that the number of syllables is not correlated with INEX 2011 QA readability human assessment (see Section 4).

As recommended by W3C, K. Inui and S. Yamamoto looked at the issue of simplification of text for easy reading. Solving this problem requires the identification of the most complex sentences in the texts [17]. The authors note that the usual measures of readability do not take into account the literacy skills of persons to whom the texts were intended. They propose to learn automatically from manually labeled examples a ranking function of sentences. In their first experiments only POS-tags are taken into account. R. Barzilay and M. Lapat propose a similar solution by learning a ranking function estimating the local consistency of the texts by means of an analysis of transitions between sentences [18]. They show that their method can be effectively applied to the detection of complex sentences in that it connects cohesion and complexity to the number of common words and entities between sentences to the number of anaphoric relations and to some syntactic information. Since INEX QA 2011 human assessment [19] contain information about syntactic quality of retrieved sentences and the list of passages with unsolved anaphora, we plan to train a specific model. More recently, Tanaka-Ishii have addressed the problem of building a training corpus clustered in different classes of difficulty. They propose to see it as a classification task by tagging only pairs of texts for which is easily possible to determine, for each pair, which of both texts is more complex than the other [20]. K. Collins-Thompson and J. Callan have developed an approach to automatically determine the grade level associated with a web page using a multinomial naive Bayesian approach which assumes that the different levels of difficulty are not independent (grade 1 is closest to grade 2 than to grade 3). They lead to a better prediction quality than that obtained by means of a measure such Flesch [11]. S. Schwarm and M. Ostendorf have extended this method to n-gram models that capture the syntactic complexity of sentences [21]. They combine, using a SVM classifier, the scores obtained by the Flesch-Kincaid formula, the number of new words and some information such as the average number of noun phrases and verbal phrases in sentences. An evaluation of their model to discriminate between documents from encyclopedia and from journals or from documents destinated to children or to adults, showed results significantly higher than those achieved by means of the Flesch or Lexile measures alone. E. Pitler and A. Nenkova brought together previous approaches by including an analysis of discourse relations, lexical cohesion, unigram models and numerical values such length of text, number of pronouns, definite articles, noun phrases... [22].

3 Experiments

3.1 Informational Based Summarization

For QA track, we started by querying The New York Times in the Termwatch platform[2]. Each queries returned a set of Wikipedia documents. Then, we had to summarize every retrieved document.

[2] http://qa.termwatch.es/

Relevance Scores for Initial Selection of Sentences. After some preprocessing (stemming), we calculated 3 relevance scores F_i and employed a decision formula to attribute a final score to each sentence.

1. Entropy :

$$F_1(s) = \sum tf. \log tf \tag{6}$$

where tf is term-frequency:

$$tf_{i,s} = \frac{n_{i,s}}{\sum_k n_{k,s}} \tag{7}$$

with i a word, s a sentence, $n_{i,j}$ the number of occurrences of i in j.
2. Dot product (F_2) between the sentence vector s and the vector corresponding to the title of the document s belongs to,
3. Sum Frequency F:

$$F_3(s) = \frac{\sum tf}{N} \tag{8}$$

with N the total number of words in the sentence s.

Initial Scoring of the Sentences. In order to obtain a score for each sentence s, we followed the following steps:

1. Calculate two average values $\alpha(s)$ and $\beta(s)$

$$\alpha(s) = \sum_{v/F_v(s)>0.5}^{\gamma} (\, F_v(s) \, - \, 0.5\,) \tag{9}$$

$$\beta(s) = \sum_{v/F_v(s)<0.5}^{\gamma} (\, 0.5 \, - \, F_v(s)\,) \tag{10}$$

where $F_v(s)$ is the score of a sentence s by employing a scoring function F_i and γ the number of scoring functions (in our case, $\gamma = 3$).
2. Make decision : let $SP(s)$ be the final score of sentence s:

$$SP(s) = \begin{cases} 0.5 + \frac{\alpha(s)}{\gamma} \text{ if } \alpha(s) > \beta(s) \\ 0.5 - \frac{\beta(s)}{\gamma} \text{ otherwise} \end{cases} \tag{11}$$

3.2 Combining Relevance Score and Readability

In the following, we explain how we incorporated readability in the summarization process. We chose to combine the initial score linearly with a readability score. Then, we chose to produce a summary by ranking the top scored sentences by following the order they appear in the set of documents retrieved by TermWatch.

The final score S combining original score SP and readability SL is:

$$S(s) = \lambda.SP(s) + (1 - \lambda).SL(s) \tag{12}$$

$\forall s$, $0 \leq SP(s) \leq 1$ and $0 \leq SL(s) \leq 1$ (readability scores SL are normalized according to min and max values obtained in a complete run over all topics). According to our previous experiments, we chose to set $\lambda = 0.7$ [23].

3.3 Runs Submitted

We submitted 3 runs. Each run combines readability and original scoring function. Figure 1 show the beginning of a summary ignoring readability. It corresponds to Run 1 of University of Avignon - LIA team (ID62RRun1).

Run 1 (ID123RI10UniXRun1). We used the Flesch readability formula (3)[3]. Figure 2 shows the summary produced from the one presented in Figure 1.

Humans first landed on the Moon on July 20, 1969.
During the same period, Lincoln Towing issued 139
This Sentry is the source of the infection in "Marvel Zombies".
Trogden's father said, "I call myself Heat Moon, your elder brother is Little Heat Moon.
Elusive Heat Elusive Heat (foaled April 30, 2006) is a dark bay mare by Elusive Quality.
Private equity firms, with their investors,
Because private equity firms are continuously in the process of raising,
Rydquist became interested in drifting upon moving to California.
Flack has been inducted into the Australian Sports and Athletics Hall of Fame
The following is an outdated ranking of the largest private equity firms.
These same principals guide Goodman today.
A stub for private equity and venture capital firms.

Fig. 1. Example of summary without taking into account readability

Run2 (ID123RI10UniXRun2). We used the Dale Chall formula (5). This formula does not take into account the number of syllables per sentence, but checks if the words are in the list of Dall-Chall (list of 3000 words) [24]. Figure 3 shows the summary produced from the summary presented in Figure 1.

Run 3 (ID123RI10UniXRun3). We computed the score of each sentence by using a sliding window of three sentences and by employing Flesch formula (3). Our aim was to take into account the context of each sentence (see Figure 4).

Let $SP'(s_n)$ be the new "informational relevance" score of a sentence s in position n in a document:

$$SP'(s_n) = \max\{SP(s_{n-2,..,n}), SP(s_{n-1,...,n+1}), SP(s_{n,...,n+2})\} \tag{13}$$

[3] We use the perl library Lingua::EN::Syllable to calculate the number of syllables per sentence.

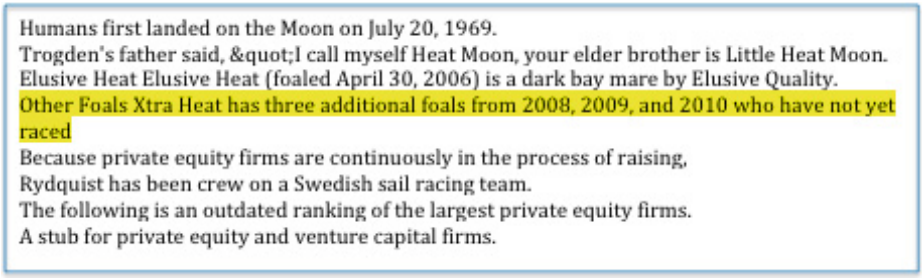

Humans first landed on the Moon on July 20, 1969.
Trogden's father said, "I call myself Heat Moon, your elder brother is Little Heat Moon.
Elusive Heat Elusive Heat (foaled April 30, 2006) is a dark bay mare by Elusive Quality.
Other Foals Xtra Heat has three additional foals from 2008, 2009, and 2010 who have not yet raced
Because private equity firms are continuously in the process of raising,
Rydquist has been crew on a Swedish sail racing team.
The following is an outdated ranking of the largest private equity firms.
A stub for private equity and venture capital firms.

Fig. 2. A summary of the document presented in Figure 1 by taking into account Flesch measure. We have highlighted the sentences that did not appear in the initial summary.

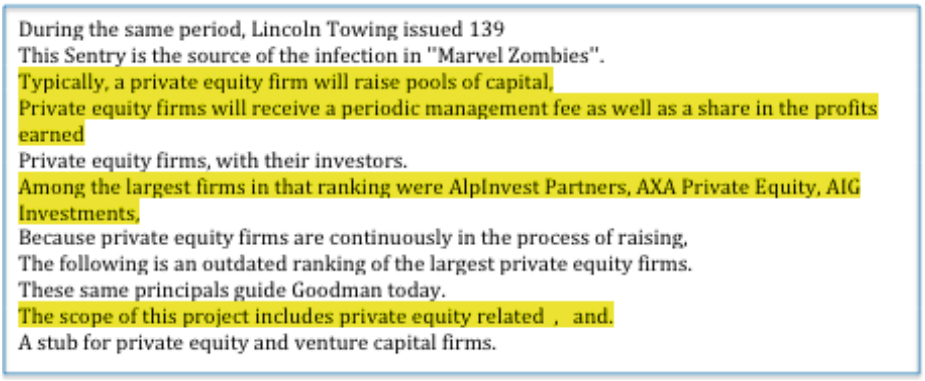

During the same period, Lincoln Towing issued 139
This Sentry is the source of the infection in "Marvel Zombies".
Typically, a private equity firm will raise pools of capital,
Private equity firms will receive a periodic management fee as well as a share in the profits earned
Private equity firms, with their investors.
Among the largest firms in that ranking were AlpInvest Partners, AXA Private Equity, AIG Investments,
Because private equity firms are continuously in the process of raising,
The following is an outdated ranking of the largest private equity firms.
These same principals guide Goodman today.
The scope of this project includes private equity related , and.
A stub for private equity and venture capital firms.

Fig. 3. A summary of the document presented in Figure 1 by taking into account Dale-Chall readability measure. We have highlighted the sentences that did not appear in the initial summary.

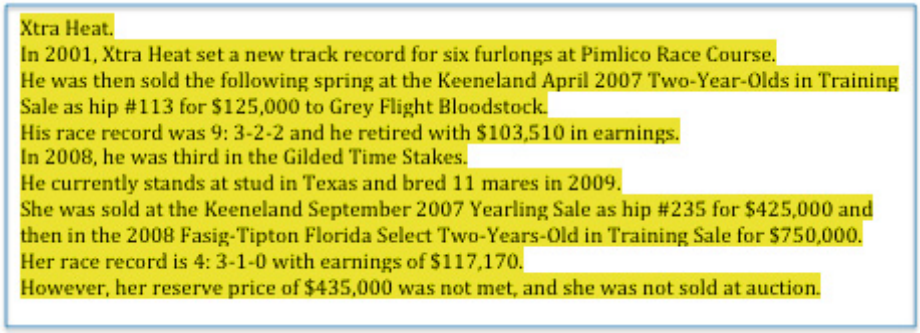

Xtra Heat.
In 2001, Xtra Heat set a new track record for six furlongs at Pimlico Race Course.
He was then sold the following spring at the Keeneland April 2007 Two-Year-Olds in Training Sale as hip #113 for $125,000 to Grey Flight Bloodstock.
His race record was 9: 3-2-2 and he retired with $103,510 in earnings.
In 2008, he was third in the Gilded Time Stakes.
He currently stands at stud in Texas and bred 11 mares in 2009.
She was sold at the Keeneland September 2007 Yearling Sale as hip #235 for $425,000 and then in the 2008 Fasig-Tipton Florida Select Two-Years-Old in Training Sale for $750,000.
Her race record is 4: 3-1-0 with earnings of $117,170.
However, her reserve price of $435,000 was not met, and she was not sold at auction.

Fig. 4. A summary by taking into account our Window-based measures. We have highlighted the sentences that did not appear in the initial summary (Figure 1).

with $SP(s_{i-1,\ldots,i+1})$ the score (formula 11) of the concatenation of the sentences in position $i-1$, i and $i+1$.

We proceed similarly to compute a new readability score SL':

$$SL'(s_n) = \max\{SL(s_{n-2,..,n}), SL(s_{n-1,\ldots,n+1}), SL(s_{n,\ldots,n+2})\} \quad (14)$$

As previously, we combined SP' and SL' to obtain a new final score S':

$$S'(s) = \lambda.SP'(s) + (1-\lambda).SL'(s) \quad (15)$$

4 Results

The official informativeness results we obtained (see Table 2) were not very good. According to "bigrams with 2-gaps" metric, our Run 1 has been ranked 21st from manual evaluation (resp. 14th from automatic evaluation), our Run 2 17th (resp. 20th), our Run 3 24th (resp. 21st). However, let us remark the summaries we produced were based on the run ID62RRun1, ranked 20th (resp. 16th). Combining informativeness and readability did not reduce informativeness even if combination changed summaries significantly (see Section 4.1).

Table 1. Proportions of queries for which the assumption of independence between the rankings (SP only or S and Flesch or Dale-Chale or Window-based) should be rejected for different values of λ (based on Kendall's tau coefficient)

	Readability measure								
	Flesch (3)			Dale-Chale (5)			Window-based (15)		
SP only	0.121			0.167			0.136		
Final score S	$\lambda = 0.6$	$\lambda = 0.7$	$\lambda = 0.8$	$\lambda = 0.6$	$\lambda = 0.7$	$\lambda = 0.8$	$\lambda = 0.6$	$\lambda = 0.7$	$\lambda = 0.8$
	0.659	0.811	0.947	0.477	0.598	0.758	0.893	0.992	1

Readability results were not good either since our runs were ranked between the twentieth and twenty-third place. The summarization approach we employed was not efficient (lot of segmenting problems were encountered) and it is clear that measures of readability are not suited to the task either. More surprising, our readability results were worse than the results of our basic summaries (ID62RRun1) whereas informativeness was better. In the following, we report some results we obtained by analyzing statistically our runs and all the runs submitted by participants.

4.1 Impact of Readability Measures on our Runs

We studied statistically the association between the different scoring functions we have tested. More precisely, we computed the Kendall rank correlation coefficient for each topic between pairs of ranking of the top 50 best scored sentences.

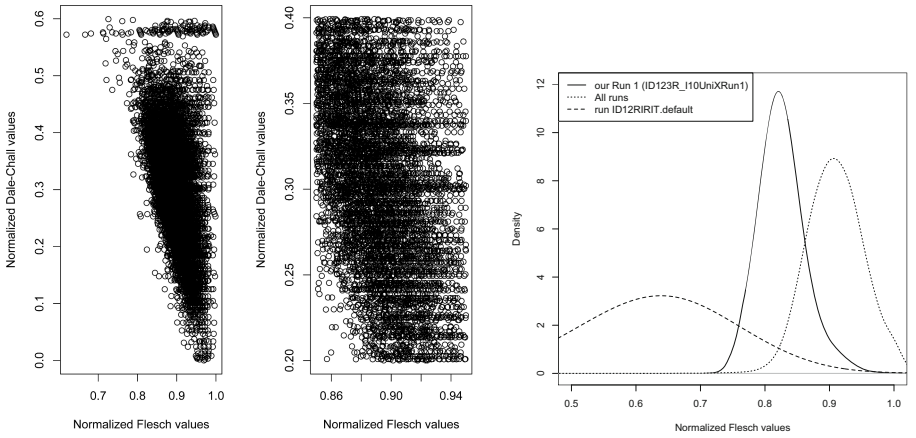


Fig. 5. Flesch and Dale-Chale values for all runs (left) and density of Flesch values for our Run 1, all INEX QA 2011 runs and run ID12RIRIT.default (right)

Table 1 shows the proportions of queries for which the assumption of independence between the rankings should be rejected (p-value < 0.1) i.e. on which there is not significant difference in the ranking of top 50 sentences. Table 1 indicates these proportions between the original relevance score and each of the three readability formulas we used and between final rankings (combinations S of relevance score SP and readability SL with different values of λ) and each of the three readability formulas.

As expected, the association between readability scores and the original score SP is very weak since only about 12-16% of the topics lead to dependent rankings. The Dale-Chale measure is the most correlated with the initial scoring SP, but it is the one that changes the final ranking the most.

4.2 Analysis of Runs Submitted by the INEX QA 2011 Participants

Flesch and Dale-Chall values (see Figure 5) are somewhat correlated since both employ the length of passages as a clue (p-value $= 0.02$ according to a Spearman rank correlation test). The normalized Flesch values estimated over the run that obtained the best manual informativeness result (ID12RIRIT.default) are more equally distributed than the values over our runs or over all the runs. This might be analyzed deeper.

In INEX QA 2011, each participant had to evaluate readability for a pool of summaries [19] without judging relevance of the text. They did not even know the topics corresponding to the summaries they had to evaluate. Assessors had to tick several kinds of check boxes. Among them, the Trash checkbox had to be ticked if the passage of the summary did not make any sense in its context

Table 2. Evaluation of our 3 runs, of the run we started from (ID62RRun1) and of the best ranked run for each criterion – official results

	Informativeness (bigrams with 2-gap) (less is better)		Readability (more is better)	
Run id	Manual	Automatic	Relaxed	Strict
ID62RRun1 – our baseline	0.1180	0.1077	328.22	266.1633
ID123RI10UniXRun1	0.1183	0.1075	295.625	246.9787
ID123RI10UniXRun2	0.1172	0.1080	304.1042	246.5745
ID123RI10UniXRun3	0.1239	0.1087	272.500	232.6744
best	0.1055	0.1049	447.3019	409.9434
median	0.1152	0.1073	393.3583	297.4167

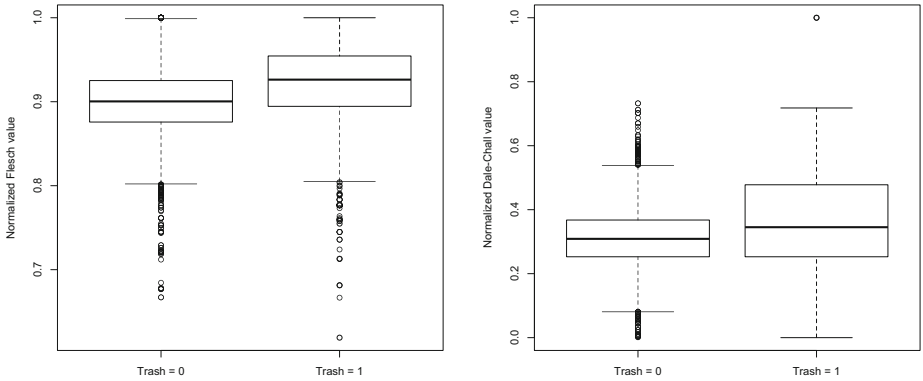


Fig. 6. Distribution of Flesch values (left) and Dale-Chall values (right) over all INEX QA 2011 runs according to the binary values of Trash index in human assessment

(*i.e.* after reading the previous passages in the same summary). Figure 6 shows the distribution of Normalized Flesch values and Dale-Chall values over all runs according to the binary values of Trash index in human assessment. No significant correlation between Flesh (resp. Dale-Chall) values and Trash values were found. Figure 7 shows the proportion of words not in Dale-Chall lexicon and mean number of syllables in all passages submitted. The mean number of syllables is not correlated to Trash values. However, we cannot reject the hypothesis that percentage of words not in Dale-Chall lexicon is a good indicator of Trash values.

5 Conclusion and Future Work

Our contributions for the INEX 2011 question-answering track have involved the combination of informational relevance scores and readability scores during extraction of sentences from documents. We submitted three runs. The first two

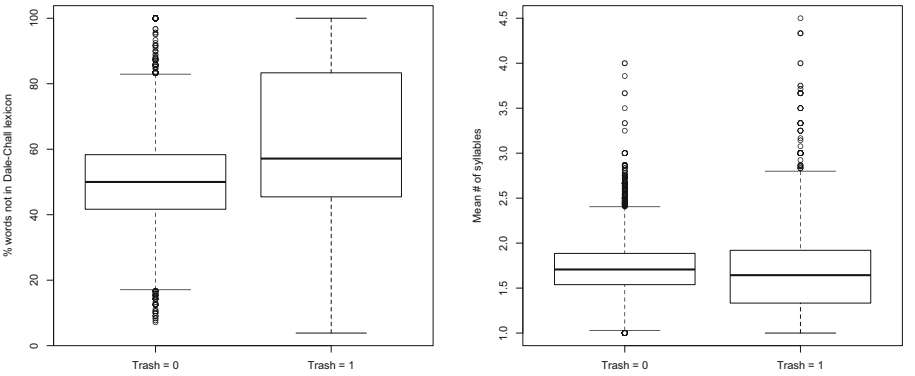


Fig. 7. Words not in Dale-Chall lexicon (left) and mean number of syllables (right) in all passages submitted, according to the values of Trash index in human assessment

by using two classical measures and the third one by proposing a smoothed window-based readability function. We conducted some statistical analysis of the runs submitted by all participants confirming that Flesch and Dale-Chall measures were not well adapted to the task. Moreover, they are too generic, they are document oriented (and not sentence / summary oriented). We plan to improve our approaches by refining readability according to human assessment.

Acknowledgments. This work is supported by the Google Grant for Digital Humanities and by ANR under CAAS project (ANR 2010 CORD 001 02).

References

1. Mizzaro, S.: Relevance: the whole history. Journal of the American Society for Information Science 48(9), 810–832 (1997)
2. Chae, J.: Predicting the fluency of text with shallow structural features: case studies of machine translation and human-written text. In: 12th Conference of the European Chapter of the ACL (EACL 2009), pp. 139–147 (2009)
3. Pitler, E., Louis, A., Nenkova, A.: Automatic evaluation of linguistic quality in multi-document summarization. In: 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010), pp. 544–554 (2010)
4. Saracevic, T.: Relevance: A review of and a framework for the thinking on the notion in information science. Journal of the American Society for Information Science 26(6), 321–343 (1975)
5. DuBay, W.H.: The principles of readability. Impact Information, 1–76 (2004)
6. Graesser, A.C., McNamara, D.S., Louwerse, M.M., Cai, Z.: Coh-metrix: Analysis of text on cohesion and language. Behavior Research Methods, Instruments, and Computers 36, 193–202 (2004)
7. Gunning, R.: The Teaching of Clear Writing. McGraw Hill, New York (1952)

8. McLaughlin, G.H.: Smog grading: A new readability formula. Journal of Reading 12(8), 639–646 (1969)
9. Flesch, R.: A new readability yardstick. Journal of Applied Psychology 32, 221–233 (1948)
10. Si, L., Callan, J.: A statistical model for scientific readability. In: Proceedings of the Tenth International Conference on Information and Knowledge Management, pp. 574–576. ACM (2001)
11. Collins-Thompson, K., Callan, J.: A language modeling approach to predicting reading difficulty. In: HLT-NAACL 4 (2004)
12. Carroll, D., Davies, P., Richman, B.: Word frequency book. American Heritage, New York (1971)
13. Dale, E., O'Rourke, J.: The living word vocabulary. World Book-Childcraft International, Chicago (1981)
14. Stenner, A.J., Horablin, I., Smith, D.R., Smith, M.: The lexile framework. MetaMetrics Inc., Durham (1988)
15. Chall, J.S., Dale, E.: Readability revisited: The new Dale-Chall readability formula. Brookline Books, Cambridge (1995)
16. Fry, E.: A readability formula for short passages. Journal of Reading 33(8), 594–597 (1990)
17. Inui, K., Yamamoto, S., Inui, H.: Corpus-based acquisition of sentence readability ranking models for deaf people. In: Sixth Natural Language Processing Pacific Rim Symposium (NLPRS 2001), Tokyo, Japan, pp. 159–166 (2001)
18. Barzilay, R., Lapata, M.: Modeling local coherence: An entity-based approach. In: 43rd Annual Meeting of the ACL, Association for Computational Linguistics, pp. 141–148. ACL (2005)
19. San Juan, E., Moriceau, V., Tannier, X., Bellot, P., Mothe, J.: Overview of the INEX 2011 Question Answering Track (QA@INEX). In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 188–206. Springer, Heidelberg (2012)
20. Tanaka-Ishii, K., Tezuka, S., Terada, H.: Sorting texts by readability. Computational Linguistics 36(2), 203–227 (2010)
21. Schwarm, S.E., Ostendorf, M.: Reading level assessment using support vector machines and statistical language models. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 523–530 (2005)
22. Pitler, E., Nenkova, A.: Revisiting readability: A unified framework for predicting text quality. In: Empirical Methods in Natural Language Processing (EMNLP 2008), pp. 186–195 (2008)
23. Sitbon, L., Bellot, P.: A readability measure for an information retrieval process adapted to dyslexics. In: 2nd International Workshop on Adaptive Information Retrieval (AIR 2008), Londres (2008)
24. Dale, Chall: A formula for predicting readability (1948)

Statistical Summarization at QA@INEX 2011 Track Using Cortex and Enertex Systems

Juan-Manuel Torres-Moreno[1],
Patricia Velázquez-Morales[2], and Michel Gagnon[1]

[1] École Polytechnique de Montréal - Département de Génie Informatique
CP 6079 Succ. Centre Ville H3C 3A7 Montréal (Québec), Canada
juan-manuel.torres@univ-avignon.fr,michel.gagnon@polymtl.ca
[2] VM Labs, 84000 Avignon, France
patricia_velazquez@yahoo.com http://www.polymtl.ca

Abstract. The Enertex system is based on a statistical physics approach. It transforms a document on a spins (words present/absent) system. A summary is obtained using the sentences with large spins interactions. In another way, the Cortex system is constructed of several different sentence selection metrics and a decision module. Our experiments have shown that the Cortex decision on the metrics always scores better than each system alone. In the INEX@QA 2011 Question Answering Track, the strategy used by Cortex system obtains very good results in automatic evaluations FRESA.

Keywords: INEX, Automatic Summarization System, Question-Answering System, Cortex, Enertex.

1 Introduction

Automatic text summarization is indispensable to cope with ever increasing volumes of valuable information. An abstract is by far the most concrete and most recognized kind of text condensation [1, 2]. We adopted a simpler method, usually called *extraction*, that allow to generate summaries by extraction of pertinence sentences [2, 3, 4, 5]. Essentially, extracting aims at producing a shorter version of the text by selecting the most relevant sentences of the original text, which we juxtapose without any modification. The vector space model [6, 7] has been used in information extraction, information retrieval, question-answering, and it may also be used in text summarization. CORTEX[1] is an automatic summarization system, recently developed [8] which combines several statistical methods with an optimal decision algorithm, to choose the most relevant sentences.

An open domain Question-Answering system (QA) has to precisely answer a question expressed in natural language. QA systems are confronted with a fine and difficult task because they are expected to supply specific information and not whole documents. At present there exists a strong demand for this kind of

[1] *CORTEX es Otro Resumidor de TEXtos* (CORTEX is anotheR TEXt summarizer).

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 247–256, 2012.

text processing systems on the Internet. A QA system comprises, *a priori*, the following stages [9]:

- Transform the questions into queries, then associate them to a set of documents;
- Filter and sort these documents to calculate various degrees of similarity;
- Identify the sentences which might contain the answers, then extract text fragments from them that constitute the answers. In this phase an analysis using Named Entities (NE) is essential to find the expected answers.

Most research efforts in summarization emphasize generic summarization [10], [11], [12]. User query terms are commonly used in information retrieval tasks. However, there are few papers in literature that propose to employ this approach in summarization systems [13], [14], [15]. In the systems described in [13], a learning approach is used (performed). A document set is used to train a classifier that estimates the probability that a given sentence is included in the extract. In [14], several features (document title, location of a sentence in the document, cluster of significant words and occurrence of terms present in the query) are applied to score the sentences. In [15] learning and feature approaches are combined in a two-step system: a training system and a generator system. Score features include short length sentence, sentence position in the document, sentence position in the paragraph, and tf.idf metrics. Our generic summarization system includes a set of eleven independent metrics combined by a Decision Algorithm. Query-based summaries can be generated by our system using a modification of the scoring method. In both cases, no training phase is necessary in our system.

This paper is organized as follows. In Section 2 we explain the INEX 2011 Question Answering Task. In Section 3 we explain the methodology of our work. Experimental settings and results obtained with Enertex and Cortex summarizers are presented in Section 4. Section 5 exposes the conclusions of the paper and the future work.

2 The INEX Initiative and the QA@INEX Track

The Initiative for the Evaluation of XML Retrieval (INEX) is an established evaluation forum for XML information retrieval (IR) [16]. This in initiative "...aims to provide an infrastructure, in the form of a large structured test collection and appropriate scoring methods, for the evaluation of focused retrieval systems".

The Question Answering track of INEX 2011 (QA) is about contextualizing tweets, i.e. answering questions of the form "What is this tweet about?" using a recent cleaned dump of the English Wikipedia.[2]

2.1 Document Collection

The document collection has been rebuilt based on a recent dump of the English Wikipedia from April 2011. Since we target a plain XML corpus for an

[2] See the official INEX 2011 Track Website:
https://inex.mmci.uni-saarland.de/tracks/qa/

easy extraction of plain text answers, all notes and bibliographic references were removed. Theses informations are difficult to handle. Then only 3,217,015 non empty Wikipedia pages (pages having at least on section) are used as corpus.

2.2 Topics

The commitee of INEX has defined 132 topics for the Track 2011. Each topic includes the title and the first sentence of a New York Times paper that were twitted at least two months after the Wikipedia dump we use. Each topic was manually checked that there is related information in the document collection.

3 The Summarization Systems Used

3.1 Cortex Summarization System

Cortex [17], [18] is a single-document extract summarization system. It uses an optimal decision algorithm that combines several metrics. These metrics result from processing statistical and informational algorithms on the document vector space representation.

The INEX 2011 Query Task evaluation is a real-world complex question (called long query) answering, in which the answer is a summary constructed from a set of relevant documents. The documents are parsed to create a corpus composed of the query and the the multi-document retrieved by a Perl program supplied by INEX organizers[3]. This program is coupled to Indri system[4] to obtain for each query, 50 documents from the whole corpus.

The idea is to represent the text in an appropriate vectorial space and apply numeric treatments to it. In order to reduce complexity, a preprocessing is performed to the question and the document: words are filtered, lemmatized and stemmed.

The Cortex system uses 11 metrics (see [19], [18] for a detailed description of these metrics) to evaluate the sentence's relevance.

1. The frequency of words.
2. The overlap between the words of the query (R).
3. The entropy the words (E).
4. The shape of text (Z).
5. The angle between question and document vectors (A).
6. The sum of Hamming weights of words per segment times the number of different words in a sentence.

[3] See: `http://qa.termwatch.es/data/getINEX2011corpus.pl.gz`

[4] Indri is a search engine from the Lemur project, a cooperative work between the University of Massachusetts and Carnegie Mellon University in order to build language modelling information retrieval tools. See: `http://www.lemurproject.org/indri/`

7. The sum of Hamming weights of the words multiplied by word frequencies.
8. The words interaction (I).
9. ...

By example, the topic-sentence overlap measure assigns a higher ranking for the sentences containing question words and makes selected sentences more relevant. The overlap is defined as the normalized cardinality of the intersection between the question word set T and the sentence word set S.

$$\text{Overlap}(T, S) = \frac{card(S \cap T)}{card(T)} \tag{1}$$

The system scores each sentence with a decision algorithm that relies on the normalized metrics. Before combining the votes of the metrics, these are partitioned into two sets: one set contains every metric $\lambda^i > 0.5$, while the other set contains every metric $\lambda^i < 0.5$ (values equal to 0.5 are ignored). We then calculate two values α and β, which give the sum of distances (positive for α and negative for β) to the threshold 0.5 (the number of metrics is Γ, which is 11 in our experiment):

$$\alpha = \sum_{i=1}^{\Gamma} (\lambda^i - 0.5); \quad \lambda^i > 0.5 \tag{2}$$

$$\beta = \sum_{i=1}^{\Gamma} (0.5 - \lambda^i); \quad \lambda^i < 0.5 \tag{3}$$

The value given to each sentence s given a query q is calculated with:

$$\begin{aligned} &\text{if}(\alpha > \beta) \\ &\quad \text{then} \;\; \text{Score}(s, q) = 0.5 + \tfrac{\alpha}{\Gamma} \\ &\quad \text{else} \;\; \text{Score}(s, q) = 0.5 - \tfrac{\beta}{\Gamma} \end{aligned} \tag{4}$$

The Cortex system is applied to each document of a topic and the summary is generated by concatenating higher score sentences.

3.2 The Enertex System

20 shows that the energy of a sentence s reflects its weight, related to the others μ sentences $\mu = 1, ...P$, in the document. We applied

$$E_{s,\mu} = (S \times S^T)^2 \tag{5}$$

where S is the matrix sentences by terms and S^T their transposed, to summarization by extraction of sentences. The summarization algorithm includes three modules. The first one makes the vectorial transformation of the text with filtering, lemmatisation/stemming and standardization processes. The second module applies the spins model and makes the calculation of the matrix of textual energy (5). We obtain the weighting of a sentence s by using its absolute energy values, by sorting according to

$$\sum_{\mu} |E_{s,\mu}| \tag{6}$$

So, the relevant sentences will be selected as having the biggest absolute energy. Finally, the third module generates summaries by displaying and concatenating of the relevant sentences. The two first modules are based on the Cortex system[5].

In order to calculate the similarity between every topic and the phrases we have used Textual Energy (5). Consequently the summary is formed with the sentences that present the maximum interaction energy with the query.

At first, the first 10 documents[6] of the cluster are concatenated into a single multi-document in chronological order. Placing the topic q (enriched or not) like the title of this long document. The Textual Energy between the topic and each of the other sentences in the document is computed using 5. Finally, we are only interested in recovering the row ρ of matrix E which corresponds to interaction energy of the topic q vs. the document. We construct the summary by sorting the most relevant values of ρ.

4 Experiments Settings and Results

In this study, we used the document sets made available during the Initiative for the Evaluation of XML retrieval (INEX)[7], in particular on the INEX 2011 QA Track (QA@INEX) https://inex.mmci.uni-saarland.de/tracks/qa/.

To evaluate the efficacy of Cortex and Enertex systems on INEX@QA track, we used the FRESA package[8].

The strategy of Cortex and Enertex systems to deal multidocument summary problem is quite simple: first, a long single document D is formed by concatenation of all $i = 1, ..., n$ relevant documents provided by Indri engine: $d_1, d_2, ...d_n$. The first line of this multidocument D is the tweet T. Both summarizers systems extract of D the most relevant sentences following T. Then, this subset of sentences is sorted by the date of documents d_i. Systems must add sentences into the summary until the word limit is reached.

[5] See next section.
[6] We used only 10 documents by limits of memory.
[7] http://www.inex.otago.ac.nz/
[8] FRESA package is available at Website: http://lia.univ-avignon.fr/fileadmin/axes/TALNE/downloads/index_fresa.html

4.1 INEX Queries Modification

Two different strategies were employed to generate the 132 queries from topics:

1. No pre-processing of topic.
2. Enrichment of the topic by manual definitions and synonyms of terms.

1) No pre-processing or modification was applied on queries set. Summarizers use the query as a title of a big multi-document retrieved by Indri engine.

2) Enrichment of topic. The query has been manually enriched using the title, the definitions and the synonyms of each term present.

Table 1 shows an example of the results obtained by Cortex and Enertex systems using 50 or 10 documents respectively as input. The topic that the summary should answer in this case was the number 2011001:

```
<topic id="2011001">
<title>At Comic-Con, a Testing Ground for Toymakers</title>
  <txt>
       THIS summer's hottest toys won't be coming to a toy aisle
       near you. The only place to get them will be at Comic-Con
       International in San Diego.
  </txt>
</topic>
```

The terms were enriched manually using their definitions and synonyms obtained from the Internet[9]. By example, for query 2011001, strategy 2 produces the next definitions for "Testing Ground" and "Toymakers" respectively:

- Testing Ground = "a place or area used for testing a product or idea"
- Toymakers = "a company that manufactures toys"

and two synonyms for "Testing":

- Testing = checking
- Testing = controlling

Then the query 2011001 is enriched as show:

q = "`At Comic-Con, a Testing Ground for Toymakers a place or area used for testing a product or idea a company that manufactures toys checking controlling`"

Table 1 presents Cortex and Enertex results (queries enriched or not) in comparison with the INEX baseline (Baseline summary), and three baselines, that is, summaries including random n-grams (Random uni-grams) and 5-grams (Random 5-grams) and empty baseline. We observe that Cortex system is always better than others summarizers.

[9] `http://dictionary.reference.com/`, `http://www.wordreference.com/` and Wikipedia.

Table 1. Example of Summarization results on topic 2011001

Summary type	Uni-grams	Bi-grams	SU4 bi-grams	FRESA Averages
Baseline summary	27.73262	35.17448	35.17507	32.69406
Empty baseline	36.20351	43.96798	43.93392	41.36847
Random uni-grams	28.99337	36.80240	36.74110	34.17896
Random 5-grams	25.36871	32.78485	32.90045	30.35133
Cortex (Query=Title)	25.62371	32.93572	32.97384	**30.51109**
Cortex (Query=Title+Definition +Synonyms)	31.50112	39.18660	39.14994	36.61255
Enertex (Query=Title)	27.93538	35.36765	35.38713	32.89672

Table 2. FRESA Averages of our three systems over 15 sampled queries

System	FRESA Averages
Cortex summary (Query = Title)	**47.6282**
Cortex summary (Query = Title + Definitions)	50.7814
Enertex summary (Query = Title)	50.5001

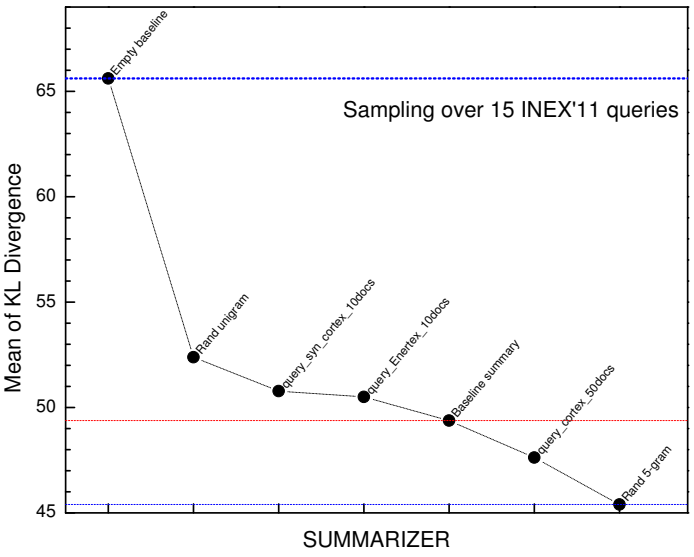


Fig. 1. FRESA Averages results for the 7 systems on INEX 2011 corpora

Table 2 presents the average results over 15 queries of our three systems (queries numbers are: 2011001, 2011005, 2011010, 2011015, 2011020, 2011025, 2011030, 2011035, 2011040, 2011050, 2011055, 2011060, 2011071, 2011080 and 2011090).

5 Conclusions

In this paper we have presented the Cortex and Enertex summarization systems. The first one is based on the fusion process of several different sentence selection metrics. The decision algorithm obtains good scores on the INEX 2011 task (the decision process is a good strategy without training corpus). The second one is based in Statistical mechanical concepts of spins models.

On INEX 2011 Question Answering Track, Cortex summarizer has obtained very good results in the automatic FRESA evaluations. The Enertex summarizer results are less good than Cortex system.

We explain this because the Enertex summarizer has used only sets of 10 documents (the Cortex system uses data sets of 50 documents). Manually query enrichment has dissappointed in this task. May be the set of documents recuperated by Indri are less pertinents if terms and their (several) definitions are used as queries. The statistical summarizers show good performances in this complex task.

References

ANSI. American National Standard for Writing Abstracts. Technical report, American National Standards Institute, Inc., New York, NY (ANSI Z39.14.1979) (1979)

Torres-Moreno, J.M.: Resume automatique de documents: une approche statistique. Hermes-Lavoisier (2011)

Luhn, H.P.: The Automatic Creation of Literature Abstracts. IBM Journal of Research and Development 2(2), 159 (1958)

Edmundson, H.P.: New Methods in Automatic Extracting. Journal of the ACM (JACM) 16(2), 264–285 (1969)

Mani, I., Mayburi, M.: Advances in automatic text summarization. The MIT Press, U.S.A. (1999)

Salton, G.: The SMART Retrieval System - Experiments un Automatic Document Processing, Englewood Cliffs (1971)

Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill (1983)

Torres-Moreno, J.M., Velazquez-Morales, P., Meunier, J.: Condensés automatiques de textes. In: Lexicometrica. L'analyse de Données Textuelles: De l'enquéte aux Corpus Littéraires (2004), www.cavi.univ-paris3.fr/lexicometrica

Jacquemin, C., Zweigenbaum, P.: Traitement automatique des langues pour l'accès au contenu des documents. Le Document en Sciences du Traitement de l'Information 4, 71–109 (2000)

Abracos, J., Lopes, G.P.: Statistical Methods for Retrieving Most Significant Paragraphs in Newspaper Articles. In: Mani, I., Maybury, M.T. (eds.) ACL/EACL 1997-WS, Madrid, Spain, July 11 (1997)

Teufel, S., Moens, M.: Sentence Extraction as a Classification Task. In: Mani, I., Maybury, M.T. (eds.) ACL/EACL 1997-WS, Madrid, Spain (1997)

Hovy, E., Lin, C.Y.: Automated Text Summarization in SUMMARIST. In: Mani, I., Maybury, M.T. (eds.) Advances in Automatic Text Summarization, pp. 81–94. The MIT Press (1999)

Kupiec, J., Pedersen, J.O., Chen, F.: A Trainable Document Summarizer. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 68–73 (1995)

Tombros, A., Sanderson, M., Gray, P.: Advantages of Query Biased Summaries in Information Retrieval. In: Hovy, E., Radev, D.R. (eds.) AAAI 1998-S, March 23-25, pp. 34–43. The AAAI Press, Stanford, California, USA (1998)

Schlesinger, J.D., Backer, D.J., Donway, R.L.: Using Document Features and Statistical Modeling to Improve Query-Based Summarization. In: DUC 2001, New Orleans, LA (2001)

Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.): INEX 2010. LNCS, vol. 6932. Springer, Heidelberg (2011)

Torres-Moreno, J.M., Velazquez-Moralez, P., Meunier, J.: CORTEX, un algorithme pour la condensation automatique de textes. In: ARCo, vol. 2, p. 365 (2005)

Torres-Moreno, J.M., St-Onge, P.L., Gagnon, M., El-Bèze, M., Bellot, P.: Automatic summarization system coupled with a question-answering system (qaas). In: CoRR, abs/0905.2990 (2009)

Torres-Moreno, J.M., Velazquez-Morales, P., Meunier, J.G.: Condensés de textes par des méthodes numériques. JADT 2, 723–734 (2002)

Fernández, S., SanJuan, E., Torres-Moreno, J.-M.: Textual Energy of Associative Memories: Performant Applications of Enertex Algorithm in Text Summarization and Topic Segmentation. In: Gelbukh, A., Kuri Morales, Á.F. (eds.) MICAI 2007. LNCS (LNAI), vol. 4827, pp. 861–871. Springer, Heidelberg (2007)

Appendix

We presents the Cortex summary of topic 2011001.

With his partner, artist Jack Kirby, he co-created Captain America, one of comics" most enduring superhero es, and the team worked extensively on such features at DC Comics as the 1940s Sandman and Sandy the Golden Boy, and co-created the Newsboy Legion, the Boy Commandos, and Manhunter. The New York Comic Con is a for-profit event produced and managed by Reed Exhibitions, a division of Reed Business, and is not affiliated with the long running non-profit San Diego Comic-Con, nor the Big Apple Convention, now known as the Big Apple Comic-Con, which is owned by Wizard World. With the relaunch, "The Swots and the Blots" became a standard bearer for sophisticated artwork, as Leo Baxendale began a three year run by adopting a new style, one which influenced many others in the comics field, just as his earlier "Beano" . Some of the strips from "Smash" survived in the new comic, including "His Sporting Lordship", "Janus Stark" and "The Swots and the Blots", but most were lost, although the "Smash" Annual continued to appear for many years afterwards . Other work includes issues of Marvel's "Captain America", "Captain Marvel", "The Power of Warlock", Ka-Zar in "Astonishing Tales", Ant-Man in "Marvel Feature", and

"The Outlaw Kid", writing a short-lived revival of Doug Wildey 's Western series from Marvel's . Powell also did early work for Fox's "Wonderworld Comics "and" Mystery Men Comics"; Fiction House's "Planet Comics", where his strips included Gale Allen and the Women's Space Battalion ; Harvey 's "Speed Comics", for which he wrote and drew the feature Ted Parrish, ; Timely 's one-shot "Tough Kid Squad . His work in the 1950s included features and covers for Street and Smith 's "Shadow Comics" ; Magazine Enterprises " " Bobby Benson's B-Bar-B Riders ", based on the children's television series, and all four issues of that publisher's "Strong Man"; and, for Harvey Comics, many war, romance, and horror stories, as well as work for the comics "Man . The winner of each round receives the same contract deal as Comic Creation Nation winners, ie a team of professional artists creating a 22-page comic using the writer's script and background elements, shared ownership of development rights, a publicity campaign built around their property, and the Zeros 2 Heroes team pushing the property towards other entertainment venues. The duo left the comic book business to pursue careers in feature films and were involved producing the feature film adaptation of "Mage" by legendary comic book creator Matt Wagner with Spyglass Entertainment, and had various projects with Mike Medavoy, Mark Canton, Akiva Goldsman, and Casey Silver. The company has been very active by participating in various events, including the L_A Times Festival of Books, Heroes Con, San Diego Comic Con, Toronto Fan Expo, D23 Disney Convention, and Baltimore Comic Con. The ECBACC STARS Workshop is an ECBACC initiative designed to use comic book art and imagery as a vehicle to foster creativity and promote literacy, with a secondary focus on introducing participants to the various career options that exist within the comic book industry. Co-presenting with comics author and scholar Danny Fingeroth during a Comics Arts Conference panel at 2008's Comic-Con International in San Diego, California, the creators explained how the first Rocket Llama story evolved into a webcomic. In addition to "The Ongoing Adventures of Rocket Llama", e-zine features from the "Rocket Llama Ground Crew" "include" "Action Flick Chick" movie reviews by G4TV 's Next Woman of the Web, cosplayer "Katrina Hill"; "The Action Chick" webcomic; "Marko's Corner" comics, cartoon arts, and podcasts by "Marko Head"; "Reddie Steady" comics for college newspapers; "The Workday Comic", the 8-hour . After the redrawn number #112's online publication came the serialized time travel story #136-137, Time Flies When You"re on the Run, appearing one page at a time throughout each week and expanding the cast with characters like the scientist Professor Percival Penguin and cavedogs who joined them by stowing away in the heroes" time rocket during the supposed previous . Other tie-in products were produced, including lunchboxes, 3-D puffy stickers, party supplies, paintable figurines, Underoos, coloring and activity books, Stain-A-Sticker, Justice League of America Skyscraper Caper game, sunglasses, playhouses, belt buckles, sneakers, cufflinks, signature stamp sets, coloring play mats, drinking glasses/tumblers, model kits, soap, stain painting sets, calendars, Play-Doh sets, jointed wall figures, wrist watches, jigsaw puzzles (Jaymar and .

QA@INEX Track 2011: Question Expansion and Reformulation Using the REG Summarization System

Jorge Vivaldi and Iria da Cunha

Universitat Pompeu Fabra
Institut Universitari de Lingüística Aplicada
Barcelona
{iria.dacunha,jorge.vivaldi}@upf.edu
http://www.iula.upf.edu

Abstract. In this paper, our strategy and results for the INEX@QA 2011 question-answering task are presented. In this task, a set of 50 documents is provided by the search engine Indri, using some queries. The initial queries are titles associated with tweets. Reformulation of these queries is carried out using terminological and named entities information. To design the queries, the full process is divided into 2 steps: a) both titles and tweets are POS tagged, and b) queries are expanded or reformulated, using: terms and named entities included in the title, terms and named entities found in the tweet related to those ones, and Wikipedia redirected terms and named entities from those ones included in the title. In our work, the automatic summarization system REG is used to summarize the 50 documents obtained with these queries. The algorithm models a document as a graph to obtain weighted sentences. A single document is generated and it is considered the answer of the query. This strategy, combining summarization and question reformulation, obtains good results regarding informativeness and readability.

Keywords: INEX, Question-Answering, Terms, Named Entities, Wikipedia, Automatic Summarization, REG.

1 Introduction

The Question-Answering (QA) task can be related to two types of questions: very precise questions (expecting short answers) or complex questions (expecting long answers, including several sentences). The objective of the QA track of INEX 2011 (Initiative for the Evaluation of XML retrieval) is oriented to the second one. Specifically, the QA task to be performed by the participating groups of INEX 2011 is contextualizing tweets, i.e. answering questions of the form "what is this tweet about?" using a dump of the Wikipedia (WP) prepared by the INEX's organization. The general process involves: tweet analysis, passage and/or XML elements retrieval and construction of the answer. Relevant passages segments should contain relevant information and as little non-relevant information as

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 257–268, 2012.

possible. The used corpus in this track contains all the texts included into the English WP. The expected answers are short documents of less than 500 words exclusively made of aggregated passages obtained from the WP corpus.

Thus, we consider that automatic extractive summarization systems could be useful in this QA task, taking into account that a summary can be defined as "a condensed version of a source document having a recognizable genre and a very specific purpose: to give the reader an exact and concise idea of the contents of the source" [1]. Summaries can be divided into "extracts", if they contain the most important sentences extracted from the original text (ex. [2], [3], [4], [5], [6], [7]) and "abstracts", if these sentences are re-written or paraphrased, generating a new text (ex. [8], [9], [10]). Most of the current automatic summarization systems are extractive. To carry out this task, we use REG ([11], [12]), an automatic extractive summarization system based on graphs. We perform some expansions and reformulations of the initial INEX@QA 2011 queries, using terms and named entities, in order to obtain a list of terms related with the main topic of all questions. This approach is similar to the one used at QA@INEX track 2010 [13], since the same summarization system is employed. Nevertheless, in this work, the strategies to build the queries are improved using semantic information, having WP as a resource.

The evaluation of the participant systems in the INEX 2011 QA Track involves two aspects: informativeness and readability. Informativeness evaluation is automatic, using the automatic evaluation system FRESA ([14], [15], [16]), and readability evaluation is manual (evaluating syntactic incoherence, unsolved anaphora, redundancy, etc.).

This paper is organized as follows. In Section 2, the summarization system REG is shown. In Section 3, queries expansions and reformulations are explained. In Section 4, experimental settings and results are presented. Finally, in Section 5, conclusions are exposed.

2 The REG System

REG. ([11], [12]) is an Enhanced Graph summarizer (REG) for extract summarization, using a graph approach. The strategy of this system has two main stages: a) to carry out an adequate representation of the document and b) to give a weight to each sentence of the document. In the first stage, the system makes a vectorial representation of the document. In the second stage, the system uses a greedy optimization algorithm. The summary generation is done with the concatenation of the most relevant sentences (previously scored in the optimization stage).

REG algorithm contains three modules. The first one carries out the vectorial transformation of the text with filtering, lemmatization/stemming and normalization processes. The second one applies the greedy algorithm and calculates the adjacency matrix. We obtain the score of the sentences directly from the algorithm. Therefore, sentences with a higher score are selected as the most relevant. Finally, the third module generates the summary, selecting and concatenating the relevant sentences. The first and second modules use CORTEX [6], a

system that carries out an unsupervised extraction of the relevant sentences of a document using several numerical measures and a decision algorithm.

3 Terms and Named Entities Extraction

In terminology, a term is usually defined as a lexical unit that designates a concept of a thematically restricted domain. Their detection implies to distinguish between domain-specific terms and general vocabulary. Its results are useful for any NLP task containing a domain specific component: ontology and (terminological) dictionary building, text indexing, automatic translation and summarization systems, among others. In spite of such numerous applications, term recognition still constitutes a bottleneck for many applications.

As shown in [17], [18] and [19] among others, there are several methods to obtaing the terms from a corpus. On the one hand, there are methods based on linguistic knowledge, like Ecode [20]. On the other hand, there are methods based on single statistical measures, such as ANA [21] or a combination of them, such as EXTERMINATOR [22]. Some tools combine both linguistic knowledge and statistically based methods, such as TermoStat [23], the algorithm shown in [24] or the bilingual extractors by [25] and [26]. However, none of these tools uses any kind of semantic knowledge. Notable exceptions are Metamap [27], Trucks [28] and YATE [29], among others. Also Wikipedia must be considered, since it is a very promising resource that is increasingly used for both monolingual ([30], [31]) and multilingual term extraction [32].

Named Entity Recognition (NER) may be defined as the task to identify names referring to persons, organizations and location in free text; later this task has been expanded to obtain other entities like dates and numeric expressions. This task was originally introduced as possible types of fillers in Information Extraction systems at the 6th Message Understanding Conference [33]. Although initially this task was limited to identify such expressions, later it has been expanded to their labeling with one entity type label. Note that an entity (such as "Stanford", the American university at the U.S.) can be referred to by multiple surface forms (e.g., "Stanford University" and "Stanford") and a single surface form (e.g., "Stanford") can refer to multiple entities (the university but also an americn financer, several places in the UK or a financer group). See [34] for an interesting review.

NER has demostated to be a task useful for a number of NLP tasks as question answering, textual entailment and coreference resolution among others. The recent interest in emerging areas like bioinformatics allow to expand this recognition task to proteins, drugs and chemical names. While early studies were mostly based on handcrafted rules, most recent ones use supervised machine learning as a way to automatically induce rule-based systems or sequence labeling algorithms starting from a collection of training examples.

The starting point of this work is to consider that the terms and named entities included into the titles and the associated tweets are representative of the main subject of these texts. If this assumption is true, the results obtained

by a search engine would be optimized with a list of terms and named entities of the queries and other ones related to them, instead the initial queries.

Thus, we decide to generate 3 types of queries to Indri:

a) Using the initial query string (the title of the tweet).

b) Using the initial query string plus a list of lexical units including: i) terms and named entities obtained from the associated tweet that are semantically related to the terms and named entities of the title and ii) Wikipedia redirected terms and named entities from those ones included in the title.

c) Using only the mentioned list of terms and named entities.

The procedure for obtaining the above mentioned additional terms and named entities from the tweet is the following:

1) To find the terms and named entities present in both query and tweet strings and verify that such lexical units are present in WP. This procedure is again splitted in two stages: first finding the terms and named entities, and then looking for such unit in WP. The last step is close to those presented in [35].

2) To compare each unit in the tweet with all the units found in the query. Such comparison is made using the algorithm described in [36].

3) To choose only those units that are higher than a given threshold.

Figure 1 shows how the enriched query is built. From the query string we obtain a number of terms: (t_{tm}) we repeat the procedure with the tweet string (t_{tm}). We look for such terms in the WP; only the terms (or a substring of them) that have an entry in WP are considered. Then, we calculate the semantic relatedness among each term of the tweet (t_{tn}) with each term of the query. Only those terms of the tweets whose similarity with some of the terms of the query is higher that a threshold value are taken into account. Assuming a query and tweet string as shown in Figure 1, each t_{tm} is compared with all t_{qn}. As a result of such comparisons, only t_{t2} and t_{t4} will be inserted in the enriched query because t_{t1} and t_{t3} will be rejected because no relations have been found among them in WP or such relation was under the threshold.

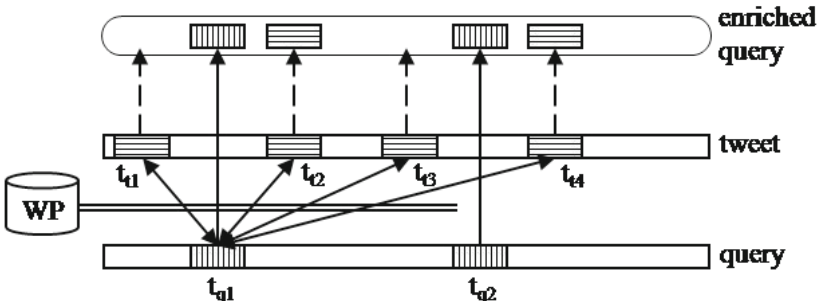


Fig. 1. Advanced query terms selection

As mentioned above, the comparison among different WP articles is done by using the algorithm described in [37]. The idea is pretty simple and it is based in the links extending each article: higher is the number of such links shared by both articles, higher is their relatedness. Figure 2 shows an outline about how to calculate the relatedness among the WP pages "automobile" and "global warming". It is clear that some outgoing links ("air pollution", "alternative fuel", etc.) are shared by both articles while other are not ("vehicle", "Henry Ford", "Ozone"). From this idea it is possible to build such measure.

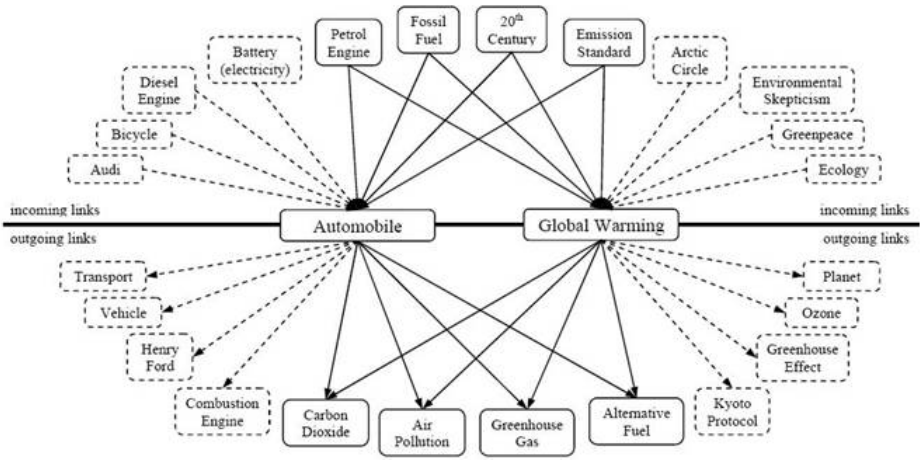


Fig. 2. Looking for the relation among WP articles (reprinted from [35])

Let's see an example of some queries generated in our experiment. For the initial query (the title of the tweet) "Obama to Support Repeal of Defense of Marriage Act", we extract the term "defense" and "marriage act", and the named entity "Obama". Moreover, we add the named entity "Barack Obama", since there is a redirection link in WP from "Obama" to "Barack Obama". Finally, some terms ("law", "legal definition", "marriage union", "man", "woman", "support" and "gay rights") and named entities ("President Obama" and "White House") semantically related with the units of the title are selected.

In this same example, the 3 different queries are the following ones:

a) Initial query: Obama to Support Repeal of Defense of Marriage Act.

b) Expanded query: Obama to Support Repeal of Defense of Marriage Act, Barack Obama, law, legal definition, marriage union, man, woman, support, gay rights, President Obama, White House.

c) Reformulated query: Obama, Defense, Marriage Act, Barack Obama, law, legal definition, marriage union, man, woman, support, gay rights, President Obama, White House.

The term and named entity extraction is carried out manually. As mentioned, nowadays several term extraction systems and named entity recognition systems exist for English. Nevertheless, their performances are not still perfect, so if these systems are employed in our work, their mistakes and the mistakes of the system presented here would be mixed. Moreover, term extractors are usually designed for a specialized domain, as medicine, economics, law, etc, but the queries provided by INEX@QA 2011 include several topics.

4 Experiments Settings and Results

In this study, we use the document sets available for the INEX 2011 QA Track (QA@INEX). These sets of documents are provided by the search engine Indri.[1] REG produces multidocument summaries using the set of 50 documents provided by Indri, employing all the initial queries of the track and the expansions and reformulations following our strategy.

The evaluation of all the participant systems in the INEX 2011 QA Track involves two aspects: informativeness and readability. On the one hand, to evaluate the informativeness, the automatic FRESA (FRamework for Evaluating Summaries Automatically) package is used. This evaluation framework includes document-based summary evaluation measures based on probabilities distribution, specifically, the Kullback-Leibler (KL) divergence and the Jensen-Shannon (JS) divergence. As in the ROUGE package [38], FRESA supports different n-grams and skip n-grams probability distributions. FRESA environment has been used in the evaluation of summaries produced in several European languages (English, French, Spanish and Catalan), and it integrates filtering and lemmatization in the treatment of summaries and documents. FRESA is available in the following link: `http://lia.univ-avignon.fr/fileadmin/axes/TALNE/Ressources.html` .

Table 1 includes the final official results of the informativenes evaluation in the INEX 2011 QA Track. This table (and the following ones) presents participants results in comparison with an intelligent baseline (Baseline_summ) and a normal baseline (Baselinemwt), including nominal phrases with more than two words, obtanined from the intelligent baseline.

In Table 1 (and the following ones), our query "a" (initial query) is named ID128RRun1, our query "b" (expanded query) is named ID128RRun2 and our query "c" (reformulated query) is named ID128RRun3. As the table shows, ID128RRun2 (FRESA average score: 0.0834384) is the sixth best run of the track participants. ID128RRun3 (FRESA average score: 0.0868463) and ID128RRun1 (FRESA average score: 0.0888517) are in positions 12 and 18, respectively. The intelligent baseline used in the track is the tenth best run, obtaining a FRESA average score of 0.0858814. Taking into account that the three best runs come from the same system (that is, they are different runs of a system developed by

[1] Indri is a search engine from the Lemur project, a cooperative work between the University of Massachusetts and Carnegie Mellon University in order to build language modelling information retrieval tools: `http://www.lemurproject.org/indri/`

Table 1. Final results of informativeness in the INEX 2011 QA Track

Run	unigram	bigram with 2-gap	Average	Ranking
ID12RIRIT_default	0.0486387	0.0786704	0.105506	0.0786968	0.105506
ID12RIRIT_07_2_07_1_dice	0.0487809	0.078857	0.105747	0.07889	0.105747
ID12RIRIT_07_2_07_1_jac	0.0490829	0.0792494	0.106195	0.0792772	0.106195
ID129RRun1	0.0502533	0.0806758	0.107806	0.0806888	0.107806
ID129RRun2	0.0517803	0.0829869	0.110616	0.0829543	0.110616
ID128RRun2	0.0523721	0.0834496	0.111033	0.0834384	0.111033
ID138RRun1	0.0523834	0.0837395	0.111516	0.0837162	0.111516
ID18RRun1	0.052567	0.0838357	0.111666	0.0838574	0.111666
ID126RRun1	0.0534638	0.0847539	0.112529	0.0847518	0.112529
Baselinesum	0.0536912	0.0859148	0.114346	0.0858814	0.114346
ID126RRun2	0.0546076	0.0863284	0.114404	0.0863106	0.114404
ID128RRun3	0.0549035	0.0868748	0.11512	0.0868463	0.11512
ID129RRun3	0.054883	0.086928	0.115219	0.0868958	0.115219
ID46RJU_CSE_run1	0.056092	0.0876557	0.115557	0.0876168	0.115557
ID46RJU_CSE_run2	0.056122	0.0876816	0.11558	0.087643	0.11558
ID62RRun3	0.0564556	0.0886844	0.117158	0.0886665	0.117158
ID123RI10UniXRun2	0.0561432	0.0885381	0.117196	0.0885373	0.117196
ID128RRun1	0.0565504	0.0888856	0.117406	0.0888517	0.117406
Baselinemwt	0.0557855	0.0886043	0.117854	0.0887005	0.117854
ID62RRun1	0.0566099	0.0892074	0.118016	0.0892034	0.118016
ID123RI10UniXRun1	0.0567167	0.0894795	0.118346	0.0894498	0.118346
ID62RRun2	0.0571956	0.0899712	0.118805	0.0899249	0.118805
ID124RUNAMiiR12	0.0607374	0.0933504	0.122111	0.0933251	0.122111
ID123RI10UniXRun3	0.0610518	0.0945555	0.123938	0.0945017	0.123938
ID124RUNAMiiR3	0.0627938	0.0957472	0.124792	0.0957262	0.124792

the same research group) and the fourth and fifth best runs come from another system, our system is considered the third best system regarding informativeness by the INEX 2011 QA Track.

With regard to our three runs, the initial query (that is, the complete title of the tweet) obtains the worst performance, while the expanded query constitutes the best strategy. These results mean that strategies based on terminology and named entity extraction are suitable for this kind of QA task, since they allow us to obtain more adequate texts from the web. Results obtained by the expanded query are better than results of the reformulated query. This result could mean that, although term and named entity expansions are very useful, the tweets titles contain any relevant information too.

On the other hand, readability is evaluated manually. Evaluators are asked to evaluate several aspects related to syntactic incoherence, unsolved anaphora, redundancy, etc. The specific orders given to evaluators are:

- Syntax S: “Tick the box is the passage contains a syntactic problem (bad segmentation for example)”.
- Anaphora A: “Tick the box if the passage contains an unsolved anaphora”.

- Redundancy R: “Tick the box if the passage contains a redundant information, i.e. an information that have already been given in a previous passage”.
- Trash T: “Tick the box if the passage does not make any sense in its context (i.e. after reading the previous passages). These passages must then be considered as trashed, and readability of following passages must be assessed as if these passages were not present”.

Two metrics are used in this readability evaluation:

- Relaxed metric: “Count as VALID if not T”.
- Strict metric: “Count as VALID if not T or R or A or S”.

In both cases, the score is the average normalized number of words in valid passages, and participants are ranked according to this score. Summary word numbers are normalized to 500 words each.

Table 2 includes the final official results of readability evaluation in the INEX 2011 QA Track, using the relaxed metric. Table 3 contains the final results of readability evaluation, using the strict metric.

Table 2. Final results of readability in the INEX 2011 QA Track: relaxed metric results

System	Score
Baseline_sum	447.3019
ID46R_JU_CSE_run1	432.2000
ID128R_Run2	417.8113
ID12R_IRIT_default	417.3462
ID46R_JU_CSE_run2	416.5294
ID129R_Run1	413.6604
ID129R_Run2	410.7547
ID12R_IRIT_05_2_07_1_jac	409.4038
ID12R_IRIT_07_2_07_1_dice	406.3962
ID126R_Run1	404.4340
ID138R_Run1	399.3529
ID128R_Run1	394.9231
ID129R_Run3	393.3585
ID126R_Run2	377.8679
ID128R_Run3	374.6078
ID62R_Run2	349.7115
ID62R_Run1	328.2245
ID62R_Run3	327.2917
ID18R_Run1	314.8980
ID123R_I10UniXRun2	304.1042
ID123R_I10UniXRun1	295.6250
ID123R_I10UniXRun3	272.5000
ID124R_UNAMiiR12	255.2449
ID124R_UNAMiiR3	139.7021
Baseline_mwt	137.8000

Table 3. Final results of readability in the INEX 2011 QA Track: strict metric results

System	Score
Baseline_sum	409.9434
ID129R_Run1	359.0769
ID129R_Run2	351.8113
ID126R_Run1	350.6981
ID46R_JU_CSE_run1	347.9200
ID12R_IRIT_05_2_07_1_jac	344.1154
ID12R_IRIT_default	339.9231
ID12R_IRIT_07_2_07_1_dice	338.7547
ID128R_Run2	330.2830
ID46R_JU_CSE_run2	330.1400
ID129R_Run3	325.0943
ID138R_Run1	306.2549
ID128R_Run3	297.4167
ID126R_Run2	296.3922
ID62R_Run2	288.6154
ID128R_Run1	284.4286
ID62R_Run3	277.9792
ID62R_Run1	266.1633
ID18R_Run1	260.1837
ID123R_I10UniXRun1	246.9787
ID123R_I10UniXRun2	246.5745
ID123R_I10UniXRun3	232.6744
ID124R_UNAMiiR12	219.1875
Baseline_mwt	148.2222
ID124R_UNAMiiR3	128.3261

Using the relaxed metric, the intelligent baseline used in the track obtains the best results (447.3019). Our system is the second best participant system (417.8113), after the best one, ID46R_JU_CSE_run1 (432.2000). Regarding our three queries, again the expanded query obtains the best results (417.8113). Nevertheless, in this case the initial query obtains better results than the reformulated query (399.3529 and 374.6078, respectively). Using the strict metric, once again the intelligent baseline used in the track obtains the best results (409.9434). Our system in the fourth best system (330.2830), using the expanded query (330.2830). In this case the reformulated query obtains better results than the initial query (297.4167 and 284.4286, respectively). Final official results of INEX 2011 QA Track can be consulted in [39] and in the INEX 2012 website: `https://inex.mmci.uni-saarland.de/`

5 Conclusions

In this paper, our strategy and results for the INEX@QA 2011 question-answering task are presented. In this task, a set of 50 documents is provided by the search engine Indri, using some queries (the initial ones are titles associated

with tweets). In our work, as explained, a reformulation of these queries is carried out using terminological and named entities information. The automatic summarization system REG is used to summarize the 50 documents obtained with these queries. This strategy, combining question reformulation and summarization, obtains good results regarding the two main aspects evaluated in the track: informativeness and readability. On the one hand, with regard to informativeness, our system is considered the third best system in the track. The two runs using terminology and named entities (that is, the expanded query and the reformulated query) obtain much better results than the run using the initial query. On the other hand, regarding readability, our system is the second best participant system applying the relaxed metric and the fourth best system applying the stric metric. In both cases, the best run uses the expanded query. These results indicate that strategies based on terminology and named entity extraction are suitable for this kind of QA task, since they allow us to obtain more adequate texts from the web. In summary, we consider that the combination of terminology and named entity extraction with automatic summarization is a promising strategy that can be used successfully in QA tasks.

In the future we plan to follow several parallel lines: i) to refine the queries to improve the text retrieved from the WP dump; ii) to further investigate the reason why the expanded query performs better than the reformulated query; iii) to automatize as much as possible the task of terminology and named entity extraction and iv) to investigate the actual weight of summarization process in the full task by testing other summarization systems.

References

1. Saggion, H., Lapalme, G.: Generating Indicative-Informative Summaries with SumUM. Computational Linguistics 28(4), 497–526 (2002)
2. Edmunson, H.P.: New Methods in Automatic Extraction. Journal of the Association for Computing Machinery 16, 264–285 (1969)
3. Nanba, H., Okumura, M.: Producing More Readable Extracts by Revising Them. In: Proceedings of the 18th Int. Conference on Computational Linguistics (COLING 2000), Saarbrucken, pp. 1071–1075 (2000)
4. Gaizauskas, R., Herring, P., Oakes, M., Beaulieu, M., Willett, P., Fowkes, H., Jonsson, A.: Intelligent access to text: Integrating information extraction technology into text browsers. In: Proceedings of the Human Language Technology Conference, San Diego, pp. 189–193 (2001)
5. Lal, P., Reger, S.: Extract-based Summarization with Simplication. In: Proceedings of the 2nd Document Understanding Conference at the 40th Meeting of the Association for Computational Linguistics, pp. 90–96 (2002)
6. Torres-Moreno, J.M., Velázquez-Morales, P., Meunier, J.G.: Condensés de textes par des méthodes numériques. In: Proceedings of the 6th Int. Conference on the Statistical Analysis of Textual Data (JADT), St. Malo, pp. 723–734 (2002)
7. da Cunha, I., Fernández, S., Velázquez Morales, P., Vivaldi, J., SanJuan, E., Torres-Moreno, J.-M.: A New Hybrid Summarizer Based on Vector Space Model, Statistical Physics and Linguistics. In: Gelbukh, A., Kuri Morales, Á.F. (eds.) MICAI 2007. LNCS (LNAI), vol. 4827, pp. 872–882. Springer, Heidelberg (2007)

8. Ono, K., Sumita, K., Miike, S.: Abstract generation based on rhetorical structure extraction. In: Proceedings of the Int. Conference on Computational Linguistics, Kyoto, pp. 344–348 (1994)
9. Paice, C.D.: Constructing literature abstracts by computer: Techniques and prospects. Information Processing and Management 26, 171–186 (1990)
10. Radev, D.: Language Reuse and Regeneration: Generating Natural Language Summaries from Multiple On-Line Sources. PhD Thesis. New York, Columbia University (1999)
11. Torres-Moreno, J.-M., Ramírez, J.: REG: un algorithme glouton appliqué au résumé automatique de texte. In: Proceedings of the 10th Int. Conference on the Statistical Analysis of Textual, Roma, Italia (2010)
12. Torres-Moreno, J.-M., Ramírez, J., da Cunha, I.: Un resumeur a base de graphes, indépendant de la langue. In: Proceedings of the Int. Workshop African HLT 2010, Djibouti (2010)
13. Vivaldi, J., da Cunha, I., Ramírez, J.: The REG Summarization System with Question Reformulation at QA@INEX Track 2010. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 295–302. Springer, Heidelberg (2011)
14. Torres-Moreno, J.-M., Saggion, H., da Cunha, I., SanJuan, E., Velázquez-Morales, P.: Summary Evaluation With and Without References. Polibitis: Research Journal on Computer Science and Computer Engineering with Applications 42 (2010a)
15. Torres-Moreno, J.-M., Saggion, H., da Cunha, I., Velázquez-Morales, P., SanJuan, E.: Ealuation automatique de résumés avec et sans référence. In: Proceedings of the 17e Conférence sur le Traitement Automatique des Langues Naturelles (TALN). Univ. de Montréal et Ecole Polytechnique de Montréal, Montreal (2010b)
16. Saggion, H., Torres-Moreno, J.-M., da Cunha, I., SanJuan, E., Velázquez-Morales, P.: Multilingual Summarization Evaluation without Human Models. In: Proceedings of the 23rd Int. Conference on Computational Linguistics (COLING 2010), Pekin (2010)
17. Cabré, M.T., Estopà, R., Vivaldi, J.: Automatic term detection. A review of current systems. Recent Advances in Computational Terminology 2, 53–87 (2001)
18. Pazienza, M.T., Pennacchiotti, M., Zanzotto, F.M.: Terminology Extraction: An Analysis of Linguistic and Statistical Approaches. STUDFUZZ, vol. 185, pp. 255–279 (2005)
19. Ahrenberg, L.: Term Extraction: A Review (2009) (Unpublished draft)
20. Alarcón, R., Sierra, G., Bach, C.: ECODE: A Pattern Based Approach for Definitional Knowledge Extraction. In: Proceedings of the XIII EURALEX Int. Congress, pp. 923–928. IULA, UPF, DOCUMENTA UNIVERSITARIA, Barcelona (2008)
21. Enguehard, C., Pantera, L.: Automatic Natural Acquisition of a Terminology. Journal of Quantitative Linguistics 2(1), 27–32 (1994)
22. Patry, A., Langlais, P.: Corpus-based terminology extraction. In: Proceedings of 7th Int. Conference on Terminology and Knowledge Engineering, Copenhagen (2005)
23. Drouin, P.: Acquisition automatique des termes: l'utilisation des pivots lexicaux spécialisés. Ph.D. Thesis. Université de Montréal, Montreal, Canada (2002)
24. Frantzi, K.T., Ananiadou, S., Tsujii, J.: The $C - value/NC - value$ Method of Automatic Recognition for Multi-word Terms. In: Nikolaou, C., Stephanidis, C. (eds.) ECDL 1998. LNCS, vol. 1513, pp. 585–604. Springer, Heidelberg (1998)
25. Vintar, S.: Bilingual term recognition revisited: The bag-of-equivalents term alignment approach and its evaluation. Terminology 16(2), 141–158 (2010)

26. Gómez Guinovart, X.: A Hybrid Corpus-Based Approach to Bilingual Terminology Extraction. In: Moskowich, I., Crespo, B. (eds.) Encoding the Past, Decoding The Future: Corpora in the 21st Century, pp. 147–175. Cambridge Scholar Publishing, Newcastle upon Tyne (2012)
27. Aronson, A., Lang, F.: An overview of MetaMap: historical perspective and recent advances. Journal of the American Medical Informatics Association 17(3), 229–236 (2010)
28. Maynard, D.: Term Recognition Using Combined Knowledge Sources. Ph.D. Thesis. Manchester Metropolitan University, Manchester, UK (1999)
29. Vivaldi, J.: Extracción de candidatos a término mediante combinación de estrategias heterogéneas. Ph.D. thesis. Universitat Politècnica de Catalunya. Barcelona, Spain (2001)
30. Vivaldi, J., Rodríguez, H.: Using Wikipedia for term extraction in the biomedical domain: first experiences. Procesamiento del Lenguaje Natural 45, 251–254 (2010)
31. Cabrera-Diego, L., Sierra, G., Vivaldi, J., Pozzi, M.: Using Wikipedia to Validate Term Candidates for the Mexican Basic Scientific Vocabulary. In: Proceedings of LaRC 2011: First Int. Conference on Terminology, Languages, and Content Resources, Seoul, pp. 76–85 (2011)
32. Erdmann, M., Nakayama, K., Hara, T., Nishio, S.: Improving the Extraction of Bilingual Terminology from Wikipedia. ACM Transactions on Multimedia Computing, Communications and Applications 5(4), 31.1–31.16 (2009)
33. Grishman, R., Sundheim, B.: Message Understanding Conference - 6: A Brief History. In: Proceedings of the 16th Int. Conference on Computational Linguistics, pp. 466–471 (1996)
34. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Journal of Linguisticae Investigationes 30(1), 3–26 (2007)
35. Milne, D., Witten, I.H.: An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In: Proceedings of the First AAAI Workshop on Wikipedia and Artificial Intelligence (2008)
36. Strube, M., Ponzetto, S.P.: WikiRelate! Computing Semantic Relatedness Using Wikipedia. Association for Artificial Intelligence (2006)
37. Milne, D., Witten, I.H.: A Learning to link with wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Mining, New York (2008)
38. Lin, C.-Y.: ROUGE: A Package for Automatic Evaluation of Summaries. In: Proceedings of Text Summarization Branches Out: ACL 2004 Workshop, pp. 74–81 (2004)
39. SanJuan, E., Moriceau, V., Tannier, X., Bellot, P., Mothe, J.: Overview of the INEX 2011 Question Answering Track (QA@INEX). In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 188–206. Springer, Heidelberg (2012)

Overview of the INEX 2011 Relevance Feedback Track

Timothy Chappell[1] and Shlomo Geva[2]

[1] Queensland University of Technology
t.chappell@student.qut.edu.au
[2] Queensland University of Technology
s.geva@qut.edu.au

Abstract. The INEX 2011 Relevance Feedback track offered a refined approach to the evaluation of Focused Relevance Feedback algorithms through simulated exhaustive user feedback. Run in largely identical fashion to the Relevance Feedback track in INEX 2010[2], we simulated a user-in-the loop by re-using the assessments of ad-hoc retrieval obtained from real users who assess focused ad-hoc retrieval submissions.

We present the evaluation methodology, its implementation, and experimental results obtained for four submissions from two participating organisations. As the task and evaluation methods did not change between INEX 2010 and now, explanations of these details from the INEX 2010 version of the track have been repeated verbatim where appropriate.

1 Introduction

This paper presents an overview of the INEX 2011 Relevance Feedback track. The purpose behind the track is to evaluate the performance of focused relevance feedback plugins in comparison to each other against unknown data. The data used for this track is the document collection and the assessments collected for the INEX 2009 Ad Hoc track. Organisations participated in the track by submitting their algorithms in the form of dynamic libraries implementing a ranking function capable of receiving relevance information from the evaluation platform and acting on it to improve the quality of future results. The interface also allows the algorithms to provide back more detailed information, such as the section or sections within a document that it believes are most relevant, enabling focused results to be returned.

The result of running the algorithms against a set of topics is a set of relevance assessments, which can then be scored against the same assessments used to provide feedback to the algorithms. The result is a reflection of how well the algorithms were able to learn from the relevance information they were given.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 269–277, 2012.

2 Focused Relevance Feedback

The relevance feedback approach that is the focus of this track is a modified form of traditional approaches to relevance feedback, which typically involved nominating whole documents as either relevant or not relevant. The end user would typically be presented with a list of documents which they would mark as relevant or not relevant before returning this input to the system which would search the remainder of the collection for similar documents and present them to the user.

Due to a fundamental paradigm change in how people use computers since these early approaches to relevance feedback, a more interactive feedback loop where the user continues to provide relevance information as they go through the results is now possible. We adopted a refined approach to the evaluation of relevance feedback algorithms through simulated *exhaustive* incremental user feedback. The approach extends evaluation in several ways relative to traditional evaluation. First, it facilitates the evaluation of retrieval where both the retrieval results and the feedback are *focused*. This means that both the search results and the feedback are specified as passages, or as XML elements, in documents - rather than as whole documents. Second, the evaluation is performed over a closed set of documents and assessments, and hence the evaluation is exhaustive, reliable and less dependent on the specific search engine in use. By reusing the relatively small topic assessment pools, having only several hundred documents per topic, the search engine quality can largely be taken out of the equation. Third, the evaluation is performed over executable implementations of relevance feedback algorithms rather than being performed over result submissions. Finally, the entire evaluation platform is reusable and over time can be used to measure progress in focused relevance feedback in an independent, reproducible, verifiable, uniform, and methodologically sound manner.

3 Evaluation

The Relevance Feedback track is concerned with the simulation of a user interacting with an information retrieval system, searching for a number of different topics. The quality of the results this user receives is then used to evaluate the relevance feedback approach.

The INEX Ad-Hoc track, which evaluates ranking algorithms, makes use of user-collected *assessments* on which portions of documents are relevant to users searching for particular topics. These assessments are perfect, not just for the evaluation of the rankings produced by the algorithms, but also for providing Focused Relevance Feedback algorithms with the relevance information they need.

As such, a Focused Relevance Feedback algorithm can be mechanically evaluated without a need of a real user by simulating one, looking up the appropriate

assessments for each document received from the algorithm and sending back the relevant passages.

To be able to accurately evaluate and compare the performance of different focused relevance feedback algorithms, it is necessary that the algorithms not be trained on the exact relevance assessments they are to receive in the evaluation. After all, a search engine isn't going to know in advance what the user is looking for. For this reason, it becomes necessary to evaluate an algorithm with data that was not available at the time the algorithm is written. Unlike in the Ad-Hoc track, the relevance submissions used to evaluate the plugins are also required for input to the plugins, so there is no way to provide participating organisations with enough information for them to provide submissions without potentially gaining an unrealistic advantage.

There are at least two potential ways of rectifying this. One is to require the submission of the algorithms a certain amount of time (for example, one hour) after the assessments for the Ad Hoc track were made available. This approach, however, is flawed as it allows very little margin for error and that it will unfairly advantage organisations that happen to be based in the right time zones, depending on when the assessments are released. In addition, it allows the relevance feedback algorithm to look ahead at relevance results it has not yet received in order to artificially improve the quality of the ranking. These factors make it unsuitable for the running of the track.

The other approach, and the one used in the Relevance Feedback track, is to have the participating organisations submit the algorithms themselves, rather than just the results. The algorithms were submitted as dynamic libraries written in Java, chosen for its cross-platform efficiency. The dynamic libraries were then linked into an evaluation platform which simulated a user searching for a number of different topics, providing relevance results on each document given. The order in which the documents were submitted to the platform was then used to return a ranking, which could be evaluated like the results of any ranking algorithm.

4 Task

4.1 Overview

Participants were asked to create one or more Relevance Feedback Modules intended to rank a collection of documents with a query while incrementally responding to explicit user feedback on the relevance of the results presented to the user. These Relevance Feedback Modules were implemented as dynamically linkable modules that implement a standard defined interface. The Evaluation Platform interacts with the Relevance Feedback Modules directly, simulating a user search session. The Evaluation Platform instantiates a Relevance Feedback Module object and provides it with a set of XML documents and a query.

The Relevance Feedback Module responds by ranking the documents (without feedback) and returning the ranking to the Evaluation Platform. This is so

that the difference in quality between the rankings before and after feedback can be compared to determine the extent of the effect the relevance feedback has on the results. The Evaluation Platform is then asked for the next most relevant document in the collection (that has not yet been presented to the user). On subsequent calls the Evaluation Platform passes relevance feedback (in the form of passage offsets and lengths) about the last document presented by the Relevance Feedback Module. This feedback is taken from the qrels of the respective topic, as provided by the Ad-Hoc track assessors. The simulated user feedback may then be used by the Relevance Feedback Module to re-rank the remaining unseen documents and return the next most relevant document. The Evaluation Platform makes repeated calls to the Relevance Feedback Module until all relevant documents in the collection have been returned.

The Evaluation Platform retains the presentation order of documents as generated by the Relevance Feedback Module. This order can then be evaluated as a submission to the ad-hoc track in the usual manner and with the standard retrieval evaluation metrics. It is expected that an effective dynamic relevance feedback method will produce a higher score than a static ranking method (i.e. the initial baseline rank ordering). Evaluation is performed over all topics and systems are ranked by the averaged performance over the entire set of topics, using standard INEX and TREC metrics. Each topic consists of a set of documents (the topic pool) and a complete and exhaustive set of manual focused assessments against a query. Hence, we effectively have a "classical" Cranfield experiment over each topic pool as a small collection with complete assessments for a single query. The small collection size allows participants without an efficient implementation of a search engine to handle the task without the complexities of scale that the full collection presents.

4.2 Submission Format

Participating organisations submitted JAR files that implemented the following specification:

```
package rf;

public interface RFInterface {
    public Integer[] first(String[] documentList, String query);
    public Integer next();
    public String getFOL();
    public String getXPath();
    public void relevant(Integer offset, Integer length,
                         String Xpath, String relevantText);
}
```

In the call to *first*, the algorithm is given the set of documents and the query used to rank them and must return an initial ranking of the documents. The

purpose of this is to quantify the improvement gained from providing the relevance assessments to the Relevance Feedback Module. The Evaluation Platform then calls *next* to request the next document from the algorithm, making a call to *relevant* to provide feedback on any relevant passages in the document. The optional methods *getFOL* and *getXPath*, if implemented, allow the Relevance Feedback Module to provide more focused results to the Evaluation Platform in order to gain better results from the focused evaluation. None of the submitted algorithms implemented these methods, however.

Before the track submission date, participants were also provided with an optional binary interface JAR file to allow participants to supply an algorithm in the form of native client code. The JAR file acts as a driver for the native client, passing information back and forth using pipes.

5 Results

5.1 Submissions

Two groups submitted a total of four Relevance Feedback Modules to the INEX 2011 Relevance Feedback track- down from nine submissions to the INEX 2010 Relevance Feedback track. QUT resubmitted the reference Relevance Feedback Module described in the next paragraph while the University of Otago submitted three native client submissions using the supplied driver.

To provide a starting point for participating organisations, a reference Relevance Feedback Module, both in source and binary form, was provided by QUT. This reference module used the ranking engine Lucene[3] as a base for a modified Rocchio[4] approach. The approach used was to provide the document collection to Lucene for indexing, then construct search queries based on the original query but with terms added from those selections of text nominated as relevant. A scrolling character buffer of constant size was used, with old data rolling off as new selections of relevant text were added to the buffer, and popular terms (ranked by term frequency) added to the search query. The highest ranked document not yet returned is then presented to the Evaluation Platform and this cycle continues until the collection is exhausted. The reference algorithm does not provide focused results and as such does not implement the *getFOL* or *getXPath* methods.

The University of Otago made three submissions of a native client that uses the ATIRE search engine with various settings, including Rocchio[4] pseudo-relevance feedback and tuning.

5.2 Evaluation

To evaluate the results, the first 20 topics from the INEX 2009 Ad Hoc track were chosen as the data set used for the evaluation. This was chosen due to the fact that the Ad Hoc track was not run in 2011, the INEX 2010 Ad Hoc track

results were already used in the INEX 2010 Relevance Feedback track and the INEX 2008 Ad Hoc track were used as training data for the reference submission.

The Relevance Feedback Modules submitted by participating organisations were run through the Evaluation Platform. As none of the submitted Relevance Feedback Modules returned focused results, *trec eval*[1] was used to evaluate the results.

Trec eval reports results using a variety of different metrics, including interpolated recall-precision, average precision, exact precision and R-precision. Recall-precision reports the precision (the fraction of relevant documents returned out of the documents returned so far) at varying points of recall (after a given portion of the relevant documents have been returned.) R-precision is calculated as the precision (number of relevant documents) after R documents have been seen, where R is the number of relevant documents in the collection. Average precision is calculated from the sum of the precision at each recall point (a point where a certain fraction of the documents in the collection have been seen) divided by the number of recall points.

As the Otago submissions do not make use of relevance feedback, alternative no-feedback results for them have not been listed for these submissions. The reference run therefore appears twice; as Reference when feedback is used and Reference (NF) without applying feedback. The Otago submissions have abbreviated names for clarity, but Otago (BM25) refers to untuned BM25 applied as-is. Otago (Rocchio) refers to BM25 with Rocchio pseudo-relevance feedback. Otago (Tuned) refers to BM25 with Rocchio, stemming and tuning.

Table 1. Average precision and R-precision for submitted modules

Run	Average Precision	R-Precision
Reference	0.4219	0.4126
Reference (NF)	0.3376	0.3361
Otago (BM25)	0.3580	0.3597
Otago (Rocchio)	0.3576	0.3597
Otago (Tuned)	0.3656	0.3573

The following table shows the exact precision of the submitted modules in the form of P@N precision, referring to the average proportion of relevant documents that have been returned after N documents have been returned. For example, a P@5 value of 0.5 means that, on average, 50% (or 2.5) of the first 5 documents returned were relevant.

Another way of plotting the results is the previously described interpolated recall-precision curve. This has the downside of producing occasionally unexpected results due to the smoothing being enough to show improvement in the reference run even at 0.0, despite the fact that improvements don't occur in the first 5 results as shown in the exact precision plot.

Table 2. Exact (P@N) precision

	Reference	Ref (nf)	Otago (BM25)	Otago (Rocchio)	Otago (Tuned)
P@5	0.5	0.5	0.54	0.54	0.52
P@10	0.515	0.435	0.445	0.445	0.485
P@15	0.49	0.4	0.4167	0.4167	0.45
P@20	0.4675	0.385	0.3975	0.3975	0.3975
P@30	0.4367	0.35	0.3583	0.3583	0.3533
P@100	0.3095	0.242	0.226	0.226	0.2175
P@200	0.2327	0.174	0.183	0.1828	0.1788
P@500	0.1224	0.1182	0.1154	0.1154	0.1152
P@1000	0.0636	0.0636	0.0636	0.0636	0.0636

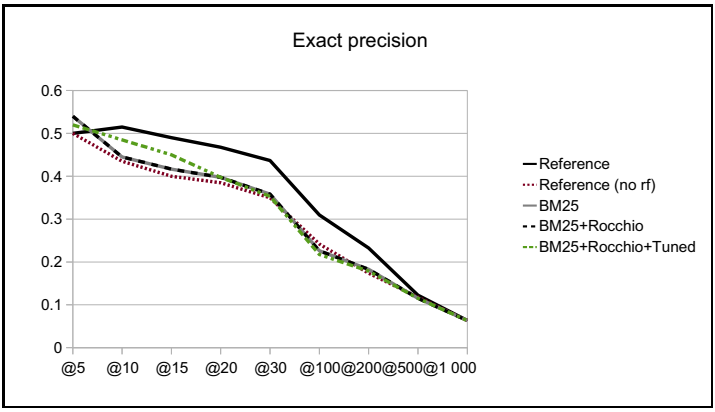


Fig. 1. Comparison of P@N precision of submitted Relevance Feedback modules

Table 3. Interpolated recall-precision

	Reference	Ref (nf)	Otago (BM25)	Otago (Rocchio)	Otago (Tuned)
0.0	0.87	0.8418	0.8481	0.8481	0.8679
0.1	0.6903	0.5741	0.5782	0.5782	0.5916
0.2	0.5877	0.5105	0.494	0.494	0.516
0.3	0.5132	0.433	0.4354	0.4354	0.4518
0.4	0.4718	0.3986	0.3943	0.3943	0.3916
0.5	0.4329	0.3078	0.3537	0.3537	0.3632
0.6	0.3912	0.2703	0.3128	0.3134	0.3305
0.7	0.3557	0.2488	0.279	0.279	0.2977
0.8	0.3015	0.2053	0.241	0.241	0.2407
0.9	0.217	0.1716	0.1761	0.1761	0.1753
1.0	0.1528	0.1291	0.1369	0.1369	0.1368

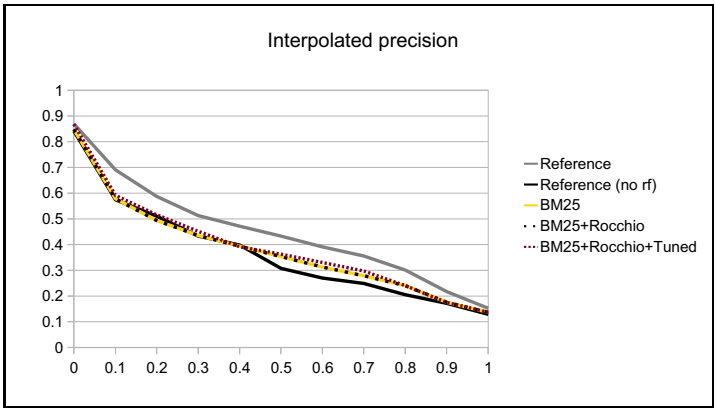


Fig. 2. Recall-precision comparison of Relevance Feedback Modules

In this case we present recall-precision data at 11 points from 0.0 to 1.0 to compare the submitted modules.

6 Conclusion

We have presented the Relevance Feedback track at INEX 2011. Despite the limited pool of participating organisations, the track has provided optimistic results, with on average improved results on INEX 2010.

The same issues that may have been a barrier to participation in INEX 2010 (increased complexity and less flexibility due to the required submission format) have persisted in INEX 2011 and unfortunately the availability of a native code plugin has not mitigated this.

The track will resume in 2012 as part of the CLEF Show Me Your Code track. While the method of participation will remain identical (supplying a relevance feedback module to be tested by an evaluation platform simulating a user) the evaluation tool will be run by participants and the results automatically submitted online. The results (including the standard precision at 10 documents and MAP) will be shown to participants, but only once the results have been successfully submitted. This will allow participants to tune their relevance feedback approaches to the data; however, it will be clear to track organisers that this has been done.

It is hoped that the increased participation of the submitter and CLEF's wider popularity will result in a larger base of submissions.

Acknowledgments. We would like to thank all the participating organisations for their contributions and hard work.

References

1. Buckley, C.: The trec eval IR evaluation package (2004) (retrieved January 1, 2005)
2. Chappell, T., Geva, S.: Overview of the INEX 2010 Focused Relevance Feedback Track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 303–312. Springer, Heidelberg (2011)
3. Goetz, B.: The Lucene search engine: Powerful, flexible, and free, Javaworld (2002), http://www.javaworld.com/javaworld/jw-09-2000/jw-0915-lucene.html
4. Rocchio, J.J.: Relevance feedback in information retrieval. In: Salton, G. (ed.) The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice-Hall Series in Automatic Computation, ch. 14, pp. 313–323. Prentice-Hall, Englewood Cliffs (1971)

Snip!

Andrew Trotman and Matt Crane

Department of Computer Science
University of Otago
Dunedin, New Zealand

Abstract. The University of Otago submitted runs to the Snippet Retrieval Track and the Relevance Feedback tracks at INEX 2011. Snippets were generated using vector space ranking functions, taking into account or ignoring structural hints, and using word clouds. We found that using passages made better snippets than XML elements and that word clouds make bad snippets. In our runs in the Relevance Feedback track we were testing the INEX gateway to C/C++ and blind relevance feedback (with and without stemming). We found that blind relevance feedback with stemming does improve prevision in the INEX framework.

Keywords: Wikipedia, Snippet Generation, Procrastination.

1 Introduction

In 2011 the University of Otago participated in two tracks it had not previously experiment with: the Snippet Retrieval Track and the Relevance Feedback Track. Six snippet runs were submitted and three relevance feedback runs were submitted. This contribution discusses those runs.

For details of the INEX document collection and the "rules" for the tracks the interested reader is referred to the track overview papers in this volume.

2 Snippets

2.1 Runs

A total of six runs were submitted:

First-p: in this run the snippet was the first 300 characters of the first <p> element in the document. This run was motivated by the observation that the start of a Wikipedia article typically contains an overview of the document and is therefore a good overview of the paper. In this run and all submitted runs the snippet was constructed so that it always started at the beginning of a word and ended at the end of a word and all XML tag content was discarded.

Top-tf-paragraph: in this run the snippet was the first 300 characters from the paragraph (<p>) element with the highest sum of query term occurrences (that is, for a two word query it is sum of the tf's of each term).

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 278–282, 2012.

Top-tf-passage : in this run the snippet was the 300 character (word aligned) sliding window with the highest sum of query term occurrences. Since there are usually many such possible windows, the window was centered so that the distance from the start of the window to the first occurrence was the same (within rounding errors) as the distance from the last occurrence to the end of the window (measured in bytes).

These three runs together form experiment 1 in which the aim is to determine whether a snippet is better formed from an element or a passage.

Top-tficf-paragraph: in this run the snippet was the first 300 characters of the paragraph with the highest *tf * icf* weight were $icf_t = \log(C/c_t)$ were C is the length of the collection (in term occurrences) and c is the number of times term t occurs.

Top-tficf-passage: in this run the snippet was the first 300 character (word aligned) sliding window with the highest tf * icf score.

The tficf runs along with the tf runs form experiment 2 in which the aim is to determine the most effective way of choosing a passage or paragraph.

The final run was

KL: in this run the KL-divergence between each term in the document and the collection was used to order all terms in the document. From this ordering the top n were chosen as the snippet so that the snippet did not exceed 300 characters.

This final run forms experiment 3 in which the aim is to determine whether snippets are better form using extractive techniques (phrases) are better than summative techniques (word clouds).

2.2 Results

The preliminary results against the GM metric are presented in Table 1. From this table it can be seen that the passage runs were universally better than the element runs and that tf.icf runs were universally better than tf runs. Our best run was passage-based tf.icf.

No run performed as well as the QUT run that simply took the first 300 characters from the document. Our run first-p took the first paragraph element from the document, but this is not equivalent. Our run did not include the document title, and in some documents the first paragraph was empty. We believe that this demonstrates the importance of putting the title into the snippet (context matters).

Other participants also submitted runs generated as word clouds, and those runs also performed below the median, suggesting that word clouds do not make good snippets.

Table 1. Snippet Track results for University of Otago runs

Rank (of 41)	Run	GM
1	LDKE-1111	0.5705
4	QUTFirst300	0.5416
11	top_tficf_passage	0.5242

Table 1. *(continued)*

27	**top_tf_passsage**	**0.4648**
28	top_tf_p	0.4574
34	top_tficf_p	0.4337
37	first_p	0.4044
39	kl	0.3598

2.3 Observations from Assessing

In total 6 topics were assessed by participants at the University of Otago. A post-assessment debriefing by the four assessors resulted in the following observations:

Snippets that included the title of the document were easier to assess than those that did not. It is subsequently predicted that those runs will generally score better than runs that did not. A recommendation is made to the track chairs to either automatically include the document title in the assessment tool, or to make it clear that the snippet may include the document title.

Snippets that were extractive from multiple parts of the document (included ellipses) generally contained multiple snippets each of which was too short to be useful and collectively not any better.

Snippets made from word clouds were generally instantly dismissible as up-helpful.

Snippets that contained what appeared to be the section / subsection "path" through the document generally took so much space that the remaining space for the extractive snippet was too short for a useful snippet.

2.4 Further Work

If the track is run in 2012 then from the observations it would be reasonable to submit a run that contains the document title, a single snippet extracted from the document, and the title of the section from which the snippet was extracted. The method of extraction is unclear and would depend on the results of the experiments submitted to this track in 2011.

3 Relevance Feedback

The purpose of the Otago relevance feedback runs was twofold: The first purpose was to experiment with the INEX relevance feedback infrastructure. In particular, the infrastructure was written in Java but the search engine Otago uses (ATIRE) is written in C++. The gateway from Java to C++ was provided by INEX.

The second purpose was to determine whether or not blind relevance feedback is an effective method of improving whole-document search results on Wikipedia. To this end the runs submitted by Otago ignored the human assessments and only returned whole documents.

3.1 Runs

A total of three runs were submitted:

BM25: in this run the documents are ranked using BM25 (k1=0.9, b=0.4). No relevance feedback was performed. This runs forms an out-of-the-box baseline. Is it the result of running the untrained ATIRE search engine over the documents.

BM25-RF: in this run the documents are ranked using BM25 (as above), then from the top 5 document the top 8 terms were selected using KL-divergence. These were then added to the query (according to Rocchio's algorithm) and BM25 was again used to get the top results. Terms added to the query had an equal weight to those already there, and terms already in the query could be added. The parameters 5 and 8 were chosen through learning over the training data.

BM25-RF-S: in this run the documents are ranked using BM25, then from the top 5 document the top 8 terms were selected using KL-divergence (as above). Additionally the S-stemmer was used in the initial and second query. Additional to this the parameters for BM25 were learned using a grid search (k1=0.5 b=0.5). Again training was on the INEX supplied training data.

In all runs blind relevance feedback was used and the user's assessments were ignored. As such these runs form a good baseline for ignoring the user.

3.2 Results

A subset of the official INEX published results are presented in Table 1 and the Precision / Recall graph is presented in Figure 1. The focused retrieval results are not presented as whole-document retrieval was used.

From the results, it appears as though relevance feedback has no effect on the performance of the search engine, but stemming does. It is already know that stemming works on the INEX Wikipedia collection, but unexpected that Rocchio Feedback does not. This result needs to be verified as it could be a problem with the run or a problem with the assessment method.

Table 2. INEX Published Results (from INEX)

Precision	Reference	Reference no feedback	Otago BM25	Otago BM25-RF	Otago BM25-RF-S
P@5	0.500	0.500	0.540	0.540	0.520
P@10	0.515	0.435	0.445	0.445	0.485
P@15	0.490	0.400	0.417	0.417	0.450
P@20	0.468	0.385	0.398	0.398	0.398
R-Precision	0.413	0.336	0.360	0.360	0.357

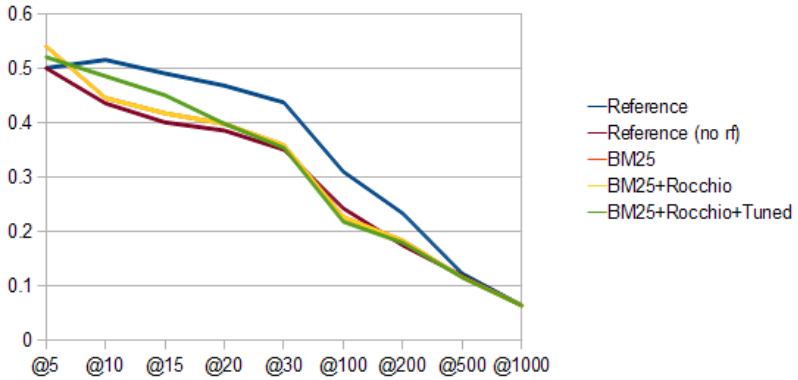


Fig. 1. Official INEX Relevance Feedback Result (from INEX)

3.3 Further Work

If the track is run in 2012 then it is reasonable to build on the baseline by including the user's assessments in the run. This could be done by performing a process similar to run BM25-RF-S at each assessment point and returning the top as-to un-seen document.

However, before any further work is done it is important to understand why relevance feedback does not appear to have an effect on this collection.

4 Conclusions

The University of Otago submitted six snippet runs and three feedback runs. These runs form baselines for experiments in improving the quality of the results in the search engine.

It is not clear why the relevance feedback method had no effect on precision. In further work this will be investigated.

The interested reader is referred to the overview papers of INEX 2011, especially the overview of the snippet and relevance feedback tracks.

Overview of the INEX 2011 Snippet Retrieval Track

Matthew Trappett[1], Shlomo Geva[1], Andrew Trotman[2], Falk Scholer[3], and Mark Sanderson[3]

[1] Queensland University of Technology, Brisbane, Australia
matthew.trappett@qut.edu.au, s.geva@qut.edu.au
[2] University of Otago, Dunedin, New Zealand
andrew@cs.otago.ac.nz
[3] RMIT University, Melbourne, Australia
falk.scholer@rmit.edu.au, mark.sanderson@rmit.edu.au

Abstract. This paper gives an overview of the INEX 2011 Snippet Retrieval Track. The goal of the Snippet Retrieval Track is to provide a common forum for the evaluation of the effectiveness of snippets, and to investigate how best to generate snippets for search results, which should provide the user with sufficient information to determine whether the underlying document is relevant. We discuss the setup of the track, and the evaluation results.

1 Introduction

Queries performed on search engines typically return far more results than a user could ever hope to look at. While one way of dealing with this problem is to attempt to place the most relevant results first, no system is perfect, and irrelevant results are often still returned. To help with this problem, a short text snippet is commonly provided to help the user decide whether or not the result is relevant.

The goal of snippet generation is to provide sufficient information to allow the user to determine the relevance of each document, without needing to view the document itself, allowing the user to quickly find what they are looking for.

The INEX Snippet Retrieval track was run for the first time in 2011. Its goal is to provide a common forum for the evaluation of the effectiveness of snippets, and to investigate how best to generate informative snippets for search results.

2 Snippet Retrieval Track

In this section, we briefly summarise the snippet retrieval task, the submission format, the assessment method, and the measures used for evaluation.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 283–294, 2012.

2.1 Task

The task is to return a ranked list of documents for the requested topic to the user, and with each document, a corresponding text snippet describing the document. This text snippet should attempt to convey the relevance of the underlying document, without the user needing view the document itself.

Each run is allowed to return up to 500 documents per topic, with a maximum of 300 characters per snippet.

2.2 Test Collection

The Snippet Retrieval Track uses the INEX Wikipedia collection introduced in 2009 — an XML version of the English Wikipedia, based on a dump taken on 8 October 2008, and semantically annotated as described by Schenkel et al. [1]. This corpus contains 2,666,190 documents.

The topics have been reused from the INEX 2009 Ad Hoc Track [2]. Each topic contains a short content only (CO) query, a content and structure (CAS) query, a phrase title, a one line description of the search request, and a narrative with a detailed explanation of the information need, the context and motivation of the information need, and a description of what makes a document relevant or not.

To avoid the 'easiest' topics, the 2009 topics were ranked in order of the number of relevant documents found in the corresponding relevance judgements, and the 50 with the lowest number were chosen.

For those participants who wished to generate snippets only, and not use their own search engine, a reference run was generated using BM25.

2.3 Submission Format

An XML format was chosen for the submission format, due to its human readability, its nesting ability (as information was needed at three hierarchical levels — submission-level, topic-level, and snippet-level), and because the number of existing tools for handling XML made for quick and easy development of assessment and evaluation.

The submission format is defined by the DTD given in Figure 1. The following is a brief description of the DTD fields. Each submission must contain the following:

- participant-id: The participant number of the submitting institution.
- run-id: A run ID, which must be unique across all submissions sent from a single participating organisation.
- description: a brief description of the approach used.

Every run should contain the results for each topic, conforming to the following:

- topic: contains a ranked list of snippets, ordered by decreasing level of relevance of the underlying document.

```
<!ELEMENT inex-snippet-submission (description,topic+)>
<!ATTLIST inex-snippet-submission
  participant-id CDATA #REQUIRED
  run-id CDATA #REQUIRED
>
<!ELEMENT description (#PCDATA)>
<!ELEMENT topic (snippet+)>
<!ATTLIST topic
  topic-id CDATA #REQUIRED
>
<!ELEMENT snippet (#PCDATA)>
<!ATTLIST snippet
  doc-id CDATA #REQUIRED
  rsv CDATA #REQUIRED
>
```

Fig. 1. DTD for Snippet Retrieval Track run submissions

- topic-id: The ID number of the topic.
- snippet: A snippet representing a document.
- doc-id: The ID number of the underlying document.
- rsv: The retrieval status value (RSV) or score that generated the ranking.

2.4 Assessment

To determine the effectiveness of the returned snippets at their goal of allowing a user to determine the relevance of the underlying document, manual assessment has been used. The documents for each topic were manually assessed for relevance based on the snippets alone, as the goal is to determine the snippet's ability to provide sufficient information about the document.

Each topic within a submission was assigned an assessor. The assessor, after reading the details of the topic, read through the top 100 returned snippets, and judged which of the underlying documents seemed relevant based on the snippets.

To avoid bias introduced by assessing the same topic more than once in a short period of time, and to ensure that each submission is assessed by the same assessors, the runs were shuffled in such a way that each assessment package contained one run from each topic, and one topic from each submission.

2.5 Evaluation Measures

Submissions are evaluated by comparing the snippet-based relevance judgements with the existing document-based relevance judgements, which are treated as a ground truth. This section gives a brief summary of the specific metrics used. In all cases, the metrics are averaged over all topics.

We are interested in how effective the snippets were at providing the user with sufficient information to determine the relevance of the underlying document, which means we are interested in how well the user was able to correctly determine the relevance of each document. The simplest metric is the mean precision accuracy (MPA) — the percentage of results that the assessor correctly assessed, averaged over all topics.

$$\text{MPA} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \tag{1}$$

Due to the fact that most topics have a much higher percentage of irrelevant documents than relevant, MPA will weight relevant results much higher than irrelevant results — for instance, assessing everything as irrelevant will score much higher than assessing everything as relevant.

MPA can be considered the raw agreement between two assessors — one who assessed the actual documents (i.e. the ground truth relevance judgements), and one who assessed the snippets. Because the relative size of the two groups (relevant documents, and irrelevant documents) can skew this result, it is also useful to look at positive agreement and negative agreement to see the effects of these two groups.

Positive agreement (PA) is the conditional probability that, given one of the assessors judges a document as relevant, the other will also do so. This is also equivalent to the F_1score.

$$\text{PA} = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}} \tag{2}$$

Likewise, negative agreement (NA) is the conditional probability that, given one of the assessors judges a document as irrelevant, the other will also do so.

$$\text{NA} = \frac{2 \cdot \text{TN}}{2 \cdot \text{TN} + \text{FP} + \text{FN}} \tag{3}$$

Mean normalised prediction accuracy (MNPA) calculates the rates for relevant and irrelevant documents separately, and averages the results, to avoid relevant results being weighted higher than irrelevant results.

$$\text{MNPA} = 0.5\frac{\text{TP}}{\text{TP} + \text{FN}} + 0.5\frac{\text{TN}}{\text{TN} + \text{FP}} \tag{4}$$

This can also be thought of as the arithmetic mean of recall and negative recall. These two metrics are interesting themselves, and so are also reported separately. Recall is the percentage of relevant documents that are correctly assessed.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5}$$

Negative recall (NR) is the percentage of irrelevant documents that are correctly assessed.

$$\text{NR} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{6}$$

The primary evaluation metric, which is used to rank the submissions, is the geometric mean of recall and negative recall (GM). A high value of GM requires a high value in recall and negative recall — i.e. the snippets must help the user to accurately predict both relevant and irrelevant documents. If a submission has high recall but zero negative recall (e.g. in the case that everything is judged relevant), GM will be zero. Likewise, if a submission has high negative recall but zero recall (e.g. in the case that everything is judged irrelevant), GM will be zero.

$$\mathrm{GM} = \sqrt{\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \cdot \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}} \tag{7}$$

3 Participation

In this section, we discuss the participants and their approaches.

In the 2011 Snippet Retrieval Track, 44 runs were accepted from a total of 56 runs submitted. These runs came from 9 different groups, based in 7 different countries. Table 1 lists the participants, and the number of runs accepted from them.

Participants were allowed to submit as many runs as they wanted, but were required to rank the runs in order of priority, with the understanding that some runs may not be assessed, depending on the total number of runs submitted. To simplify the assessment process, 50 runs were initially accepted, to match the number of topics. This was achieved by capping the number of runs at 8 runs per partipating institute, and discarding any runs ranked below 8.

Six submissions were later rejected. An additional three submissions did not include the full 50 topics, and are thus uncomparable with the remaining 41 submissions. They have been reported alongside the accepted submissions, but have not been assigned a ranking or included in any analysis.

3.1 Participant approaches

The following is a brief description of the approaches used, as reported by the participants.

Table 1. Participation in the Snippet Retrieval Track

ID	Institute	Runs
14	University of Otago	6
16	Kasetsart University	3
20	Queensland University of Technology	3
23	RMIT University	3
31	Radboud University Nijmegen	6
65	University of Minnesota Duluth	4
72	Jiangxi University of Finance and Economics	8
73	Peking University	8
83	Indian School of Mines, Dhanbad	3

University of Otago. Otago tried three approaches. In the first they used the first paragraph from each document. In the second they slid a window across the text looking for the most relevant (tf.icf) passage. In the third they constructed a word cloud. Their results suggest that word clouds make bad snippets, using document structure is not helpful, and that tf.icf is a reasonable ranking function.

Kasetsart University. Our system retrieves XML elements based on both indices are including Selected Weight and Leaf-Node. In our engine, the Selected Weight is based on the Term Frequency, and the Leaf-Node is based on BM25. In the snippet generation system, we use significant words, the element of relevance from both Selected Weight and Leaf-Node relevance, and then we combine relevance context to the sentences. Afterwards, The sentences with the higher relevance score will be chosen as the retrieval snippet.

Queensland University of Technology. The run 'QUTFirst300' is simply the first 300 characters (excluding the title) of the documents in the reference run.b

The run 'QUTFocused', again using the reference run, ignored certain elements, such as tables, images, templates, and the reference list. The tf-idf values were calculated for the key words found in each document. A 300 character window was then moved along the text, counting the total key words found in each window, weighted by their tf-idf scores. The highest scoring window was found, then rolled back to the start of the sentence to ensure the snippet did not start mid-sentence. The document title was not included in the snippet.

The run 'QUTIR2011A' selects snippets, using the reference run to select the appropriate documents. A topological signature is created from the terms of the query. Snippets are determined as 300 character passages starting from the `<p>` tag that is used to delineate paragraphs in the documents. Signatures are created for these snippets and compared against the original query signature. The closest match is used. The document title is added to the start of the snippet.

RMIT University. The snippet generation algorithm was based on selecting highly ranked sentences which were ranked according to the occurrence of query terms. Nevertheless, it was difficult to properly identify sentence boundaries due to having multiple contributors with different writing styles. The main exception was detected when a sentence included abbreviations such as "Dr. Smith". We did not do an analysis of abbreviations to address this issue in detail.

We processed Wikipedia articles before constructing snippets. Specifically, information contained inside the `<title>` and `<bdy>` was used to narrow the document content. We suggest that snippets should include information of the document itself instead of sources pointing to other articles. Therefore, the Reference section was ignored in our summarisation approaches. The title was concatenated to the leading scored sentences.

We used the query terms listed in the title, and we expanded them by addressing a pseudo relevance feedback approach. That is, the top 5 Wikipedia articles were employed for selecting the first 25 and 40 terms.

Radboud University. Our previous study found that topical language model improves document ranking of ad-hoc retrieval. In this work, our attention is paid on snippets that are extracted and generated from the provided ranked list of documents.

In our experiments of the Snippet Retrieval Track, we hypothesize that the user recognizes certain combinations of terms in created snippets which are related to their information needs. We automatically extract snippets using terms as the minimal unit. Each term is weighted according to their relative occurrance in its article and in the entire Wikipedia. The top K scoring terms are chosen for inclusion in the snippet. The term-extraction based snippets are then represented differently to the user. One is a cluster of words that indicate the described topic. Another is a cluster of semi-sentences that contains the topic information while preserving some language structure.

University of Minnesota Duluth. The run entitled 'p65-UMD_SNIPPET_RETRIEVAL_RUN_3' was created as follows: Our method of dynamic element retrieval was used to generate a rank-ordered list of all elements associated with each article in the reference run. The elements were focused based on correlation, and the highest correlating element was selected as the basis for the snippet. For this particular run, the corresponding element from the original text (rather than the focused element itself) was selected and further processed by examining each sentence of the element, selecting those containing at least one query term, and ordering the sentences by the number of query terms contained in them. The top 300 characters from this text string were reported.

Jiangxi University of Finance and Economics. p72-LDKE-$m_1m_2m_3m_4$, where m_i $(1 \leq i \leq 4)$ equals to 0 or 1, employs four different strategies to generate a snippet. Strategy 1 is dataset selection: using documents listed in reference runs ($m_1 = 0$) or Wikipedia 2009 dataset ($m_1 = 1$). Strategy 2 is snippet selection: using baseline method ($m_2 = 0$) or window method ($m_2 = 1$). According to the baseline method, after the candidate elements/nodes being scored and ranked, only the first 300 characters are extracted as snippet from the element/node has the highest score. Remain part of this snippet are extracted from the successive elements/nodes in case of the precedents are not long enough. While in the window method, every window that contain 15 terms are scored and those with higher scores are extracted as snippets. Strategy 3 is whether using ATG path weight ($m_3 = 1$) or not ($m_3 = 0$) in element retrieval model. The element retrieval model used in our system is based on BM25 and the works about ATG path weight has been published in CIKM 2010. Strategy 4 is whether reordering the XML document according to the reference runs ($m_4 = 0$) or not ($m_4 = 1$) after elements/nodes being retrieved.

Peking University. In the INEX 2011 Snippet Retrieval Track, we retrieve XML documents based on both document structure and content, and our retrieval engine is based on the Vector Space Model. We use Pseudo Feedback

method to expand the query of the topics. We have learned the weight of elements based on the cast of INEX2010 to enhance the retrieval performance, and we also consider the distribution of the keywords in the documents and elements, the more of the different keywords, the passage will be more relevant, and so is the distance of the keywords. We used method of SLCA to get the smallest sub-tree that satisfies the retrieval. In the snippet generation system, we use query relevance, significant words, title/section-title relevance and tag weight to evaluate the relevance between sentences and a query. The sentences with higher relevance score will be chosen as the retrieval snippet.

Indian School of Mines, Dhanbad. Our snippet retrieval task focused on a simple snippet generation technique based on the reference run provided by the task organizers. Instead of indexing the huge Wikipedia XML collection we did some pre-processing. The document collection was parsed using LIBXML2 parser and the XML tags were removed from all the XML documents. Also there were a lot of white spaces and control characters which were stripped off. This text-only version of document collection was used for snippet generation. We observed that the first few lines of each document typically provide an overall introduction to the content of the whole documents. As a naive approach we started with extraction of first 300 characters of each document appearing in the reference run as the snippets. We submitted three runs based on this notion. Each run contained snippets for all the 50 topics with each topic having a maximum of 500 snippets. The first two runs had some file access errors which was rectified in the third run.

4 Snippet Retrieval Results

In this section, we present and discuss the evaluation results for the Snippet Retrieval Track.

Table 2 gives the ranking for all of the runs. The run ID includes the ID number of the participating organisation; see Table 1 for the name of the organisation. The runs are ranked by geometric mean of recall and negative recall.

The highest ranked run is 'p72-LDKE-1111', submitted by the Jiangxi University of Finance and Economics.

Table 3 lists additional metrics for each run, as discussed in Section 2.5. One statistic worth noting is the fact that no run scored higher than 47% in recall, with an average of 35%. This indicates that poor snippets are causing users to miss more than half of all relevant results. Negative recall is high, with no run scoring below 80%, meaning that users are easily able to identify most irrelevant results based on snippets.

Significance tests were performed to determine whether higher ranked systems were significantly better than lower ranked systems. A one-tailed t-test at 95% was used. Table 4 shows, for each submission (shown on the left), whether it is significantly better than each lower ranked run (indicated by "⋆"). The top run is significantly better than runs 14, 17 and 19–41 – 62.5% of all lower-ranked

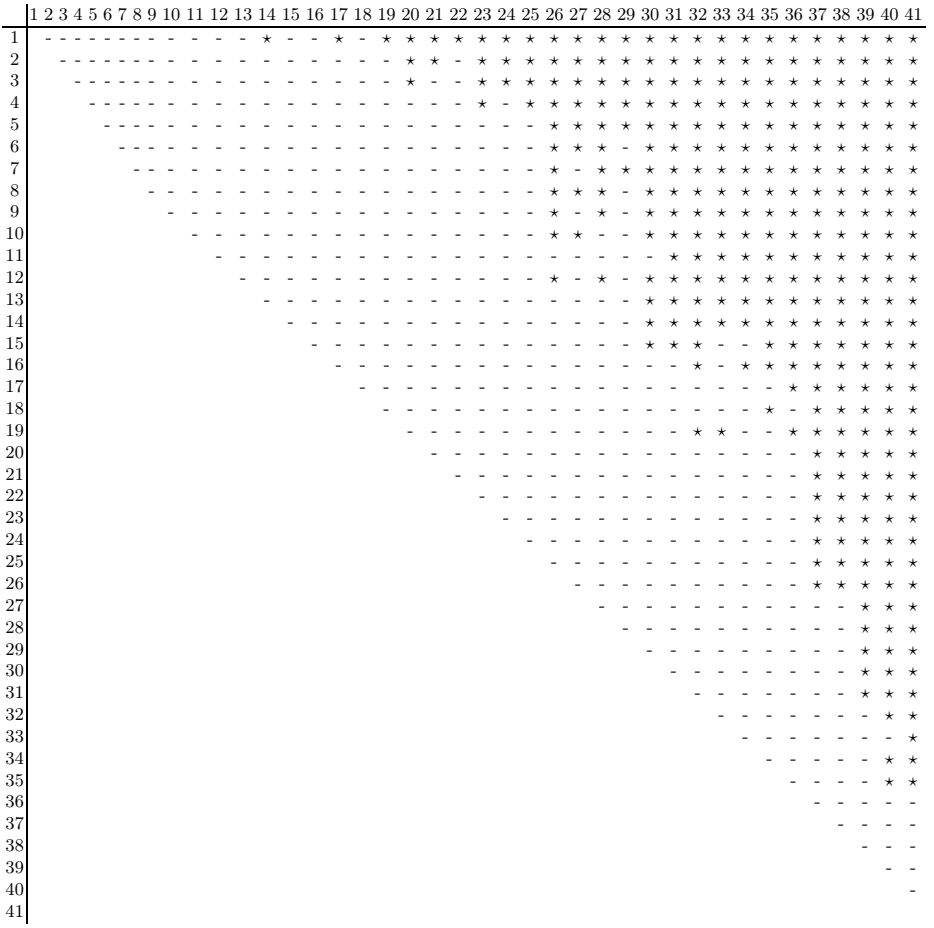
Table 2. Ranking of all runs in the Snippet Retrieval Track, ranked by GM

Rank	Run	Score
1	p72-LDKE-1111	0.5705
2	p23-baseline	0.5505
3	p72-LDKE-0101	0.5472
4	p20-QUTFirst300	0.5416
5	p73-PKU_ICST_REF_11a	0.5341
6	p72-LDKE-1110	0.5317
7	p23-expanded-40	0.5294
8	p72-LDKE-0111	0.5270
9	p65-UMD_SNIPPET_RETRIEVAL_RUN_3	0.5264
10	p20-QUTFocused	0.5242
11	p14-top_tficf_passage	0.5242
12	p23-expanded-25	0.5239
13	p72-LDKE-1121	0.5192
14	p72-LDKE-1101	0.5130
15	p20-QUTIR2011A	0.5122
16	p73-PKU_105	0.5080
17	p73-PKU_102	0.5011
18	p73-PKU_100	0.5001
19	p72-LDKE-1001	0.4919
20	p35-97-ism-snippet-Baseline-Reference-run_01	0.4886
21	p73-PKU_107	0.4805
22	p31-SRT11DocTXT	0.4803
23	p35-98-ism-snippet-Baseline-Reference-run_01	0.4800
24	p72-LDKE-1011	0.4770
25	p73-PKU_106	0.4741
26	p65-UMD_SNIPPET_RETRIEVAL_RUN_4	0.4680
27	p14-top_tf_passsage	0.4648
28	p14-top_tf_p	0.4574
29	p31-SRT11ParsDoc	0.4557
30	p65-UMD_SNIPPET_RETRIEVAL_RUN_1	0.4470
31	p73-PKU_ICST_REF_11b	0.4459
32	p35-ism-snippet-Baseline-Reference-run_02	0.4365
33	p31-SRT11DocParsedTXT	0.4351
34	p14-top_tficf_p	0.4337
35	p65-UMD_SNIPPET_RETRIEVAL_RUN_2	0.4270
36	p31-SRT11ParsStopDoc	0.4157
37	p14-first_p	0.4044
38	p73-PKU_101	0.3956
39	p14-kl	0.3598
40	p31-SRT11ParsStopTerm	0.3458
41	p31-SRT11ParsTerm	0.3392
n/a	p16-kas16-MEXIR-ALL	0.0000
n/a	p16-kas16-MEXIR-ANY	0.0000
n/a	p16-kas16-MEXIR-EXT	0.0000

Table 3. Additional metrics of all runs in the Snippet Retrieval Track (preliminary results only)

Run	MPA	MNPA	Recall	NR	PA	NA
p14-first_p	0.7582	0.6430	0.4641	0.8219	0.3748	0.8292
p14-kl	0.7638	0.6220	0.4115	0.8325	0.3329	0.8313
p14-top_tf_p	0.7684	0.6328	0.4470	0.8187	0.3646	0.8232
p14-top_tf_passsage	0.8022	0.6076	0.3269	0.8884	0.3151	0.8715
p14-top_tficf_p	0.7860	0.6263	0.3888	0.8637	0.3279	0.8587
p14-top_tficf_passage	0.7674	0.6179	0.4058	0.8299	0.3452	0.8364
p20-QUTFirst300	0.7728	0.5919	0.3064	0.8774	0.2727	0.8533
p20-QUTFocused	0.7982	0.5781	0.2446	0.9116	0.2576	0.8715
p20-QUTIR2011A	0.7580	0.6024	0.3716	0.8333	0.2959	0.8247
p23-baseline	0.7702	0.5896	0.3101	0.8692	0.2952	0.8475
p23-expanded-25	0.7988	0.6086	0.3235	0.8938	0.2725	0.8676
p23-expanded-40	0.7690	0.5708	0.2721	0.8695	0.2227	0.8360
p31-SRT11DocParsedTXT	0.8026	0.6047	0.2982	0.9113	0.2900	0.8737
p31-SRT11DocTXT	0.7992	0.6128	0.3231	0.9026	0.3007	0.8721
p31-SRT11ParsDoc	0.7830	0.5704	0.2431	0.8977	0.2171	0.8544
p31-SRT11ParsStopDoc	0.8092	0.6204	0.3513	0.8896	0.3333	0.8672
p31-SRT11ParsStopTerm	0.7830	0.6247	0.3824	0.8670	0.3387	0.8525
p31-SRT11ParsTerm	0.7766	0.6231	0.3866	0.8596	0.3389	0.8514
p35-97-ism-snippet-Baseline-Reference-run_01	0.7958	0.6441	0.4035	0.8848	0.3715	0.8667
p35-98-ism-snippet-Baseline-Reference-run_01	0.7936	0.6233	0.3597	0.8870	0.3319	0.8685
p35-ism-snippet-Baseline-Reference-run_02	0.7652	0.5877	0.3229	0.8525	0.2849	0.8365
p65-UMD_SNIPPET_RETRIEVAL_RUN_1	0.7724	0.5813	0.2904	0.8723	0.2736	0.8500
p65-UMD_SNIPPET_RETRIEVAL_RUN_2	0.7850	0.6207	0.3811	0.8602	0.3498	0.8542
p65-UMD_SNIPPET_RETRIEVAL_RUN_3	0.7976	0.5982	0.3027	0.8937	0.3078	0.8645
p65-UMD_SNIPPET_RETRIEVAL_RUN_4	0.8056	0.6245	0.3534	0.8956	0.3448	0.8706
p72-LDKE-0101	0.8042	0.6165	0.3348	0.8982	0.3161	0.8712
p72-LDKE-0111	0.8090	0.5984	0.2875	0.9093	0.2850	0.8764
p72-LDKE-1001	0.8026	0.6250	0.3706	0.8793	0.3530	0.8697
p72-LDKE-1011	0.7886	0.5732	0.2618	0.8846	0.2398	0.8623
p72-LDKE-1101	0.7580	0.6201	0.4103	0.8300	0.3105	0.8358
p72-LDKE-1110	0.7928	0.6167	0.3731	0.8602	0.3269	0.8612
p72-LDKE-1111	0.7998	0.6076	0.3288	0.8864	0.3050	0.8675
p72-LDKE-1121	0.8054	0.6176	0.3544	0.8809	0.3061	0.8705
p73-PKU_100	0.7808	0.6250	0.3884	0.8617	0.3622	0.8516
p73-PKU_101	0.7892	0.5924	0.2964	0.8883	0.2883	0.8629
p73-PKU_102	0.7656	0.5723	0.2684	0.8762	0.2081	0.8426
p73-PKU_105	0.8144	0.6444	0.3980	0.8909	0.3747	0.8773
p73-PKU_106	0.7772	0.6265	0.3902	0.8627	0.3431	0.8481
p73-PKU_107	0.5908	0.2954	0.0000	0.5908	0.0000	0.6588
p73-PKU_ICST_REF_11a	0.8998	0.4499	0.0000	0.8998	0.0000	0.9386
p73-PKU_ICST_REF_11b	0.8829	0.4414	0.0000	0.8829	0.0000	0.9305
p16-kas16-MEXIR-ALL	0.7726	0.6161	0.3690	0.8633	0.3191	0.8434
p16-kas16-MEXIR-ANY	0.7590	0.6331	0.4347	0.8314	0.3647	0.8301
p16-kas16-MEXIR-EXT	0.7892	0.6279	0.3816	0.8742	0.3522	0.8624

Table 4. Statistical significance of improvement of each run over each lower-ranked run (t-test, one-tailed, 95%)



runs. However, the average is much lower than this – of the 820 possible pairs of runs, there are only 318 (or 38.8%) significant differences. We should therefore be careful when drawing conclusions based on these results.

5 Conclusion

This paper gave an overview of the INEX 2011 Snippet Retrieval track. The goal of the track is to provide a common forum for the evaluation of the effectiveness of snippets. The paper has discussed the setup of the track, and presented the preliminary results of the track. The results show that, for the submitted runs, users are generally able to identify most irrelevant results, but poor snippets are causing them to miss over half of the relevant results, indicating that there is still substantial work to be done in this area.

References

1. Schenkel, R., Suchanek, F.M., Kasneci, G.: YAWN: A semantically annotated Wikipedia XML corpus. In: 12. GI-Fachtagung für Datenbanksysteme in Business, Technologie und Web (BTW 2007), pp. 277–291 (2007)
2. Geva, S., Kamps, J., Lethonen, M., Schenkel, R., Thom, J.A., Trotman, A.: Overview of the INEX 2009 Ad Hoc Track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 4–25. Springer, Heidelberg (2010)

Focused Elements and Snippets

Carolyn J. Crouch, Donald B. Crouch, Natasha Acquilla,
Radhika Banhatti, Sai Chittilla, Supraja Nagalla, and Reena Narenvarapu

Department of Computer Science University of Minnesota Duluth
Duluth, MN 55812
ccrouch@d.umn.edu

Abstract. This paper reports briefly on the final results of experiments to produce competitive (i.e., highly ranked) focused elements in response to the various tasks of the INEX 2010 Ad Hoc Track. These experiments are based on an entirely new analysis and indexing of the INEX 2009 Wikipedia collection. Using this indexing and our basic methodology for dynamic element retrieval [5, 6], described herein, yields highly competitive results for all the tasks involved. This is important because our approach to snippet retrieval is based on the conviction that good snippets can be generated from good focused elements. Our work to date in snippet generation is described; this early work ranked 9th in the official results.

1 Introduction

In 2010, our INEX investigations centered on integrating our methodology for the dynamic retrieval of XML elements [3] with traditional article retrieval to facilitate in particular the retrieval of *good focused elements*—i.e., elements which when evaluated are competitive with those in the top-ten highest ranked results. Earlier work [5, 6] had convinced us that our approach was sound, but the scaling up of the document collection (i.e., moving from the small Wikipedia collection used in earlier INEX competitions to the new, much larger version made available in 2009) clarified an important point for us. The new (2009) Wiki documents are much more complex in structure than their predecessors. Because our methodology depends on being able to recreate the Wiki document at execution time, every tag (of the more than 30,000 possible tags within the document set) must be maintained during processing in order for the xpath of a retrieved element to be properly evaluated. (In earlier work, we had omitted some tags—e.g., those relating to the format rather than structure—for the sake of convenience, but this was no longer possible in the new environment.) Clearly, we needed to analyze the new collection with respect to the kinds of elements we wanted to retrieve (most specifically, the terminal nodes of the document tree). We spent some time on this process and then parsed and indexed the collection based on this analysis, applied our methodology for producing focused elements (see Section 2), and used these elements as the basis for snippet generation (see Section 3). All the experiments described below are applied to this data.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 295–299, 2012.

2 Using Dynamic Element Retrieval for Focused Output

In this section, we describe our methodology for producing focused elements for the major INEX 2009 and 2010 Ad Hoc tasks. A brief overview of the application of this methodology is included in terms of experimental results for these tasks. The queries are taken from the *title* field; negative terms are removed.

2.1 Dynamic Element Retrieval

To retrieve good focused elements in response to a query, we use article retrieval (to identify the articles of interest) combined with dynamic element retrieval [3, 4] (to produce the elements) and then apply a focusing strategy to that element set. A slightly lower-level view of this process follows.

For each query, we retrieve *n* articles or documents. Our system is based on the Vector Space Model [11]; basic functions are performed using Smart [10]. We then use dynamic element retrieval to produce the elements. Dynamic element retrieval builds the document tree at execution time, based on a stored schema representing a pre-order traversal of the document, created when it is parsed, and a terminal node index of the collection. Given an article of interest, our document-building routine, Flex, first identifies its document tree and then *seeds* the tree by connecting each of its terminal nodes to the vector representing that node in the terminal node index. (Here the terminal node represents a leaf of the document tree; the content of this node contributes, along with that of its siblings, to form the element vector of the parent node.) These documents are semi-structured, so all untagged text within an element is collected to become a new, artificial child of that element. (These nodes are present in the terminal node index.) Given all terminal nodes associated with a parent, Flex builds the parent node, and the process continues until each element vector in the document tree has been generated, bottom-up, from the terminal node level to the body node of the article. *Lnu-ltu* term weighting [12], designed to deal with differences in the lengths of vectors, is utilized (with inner product) to produce a rank-ordered list of elements from each document.

2.2 Producing Focused Output

For those tasks requiring focused elements as output, we use one of the three following focusing strategies to remove overlap. The *section strategy* chooses the highest correlating non-body element along a path as the focused element. (Most of these elements turn out in fact to be sections.) The *correlation strategy* chooses the highest correlating element along a path as the focused element, without restriction on element type. And the *child strategy* chooses the terminal element along a path as the focused element (i.e., ignores correlation and always gives precedence to the child rather than the parent). Each artificial node (which represents untagged text) is purged from the list; its purpose is served when the parent node is generated.

2.3 Experiments in Focused Retrieval: INEX 2010

Here we briefly recap the experiments in focused retrieval for 2010 and summarize the results produced when our basic approach is applied to the newly analyzed and indexed Wikipedia collection.

A brief overview of the basic approach, used for each of the three Ad Hoc focused tasks, is reiterated here. Given a query, the top-ranked *n* articles are identified based on article retrieval. (All results reported below are based on the reference run.) Dynamic element retrieval is then used to build the document trees. As the trees are built, bottom up, each *Lnu*-weighted element vector is correlated with the *ltu*-weighted query using inner product. For each tree, a rank-ordered list of elements is produced. This list includes elements representing untagged text in the document, which must be present in order to generate the trees properly but which do not physically exist as elements per se. These elements are removed from the element list along with overlapping elements to produce the set of focused elements for the tree, based on the focusing strategy applied. The focused elements from each tree are then reported, in article order, for evaluation.

Four tasks make up the 2010 Ad Hoc Track. Three of these tasks, i.e., the Relevant-in-Context (RiC), Restricted Relevant-in-Context (RRiC), and Restricted Focused (RF) tasks, all center on focused retrieval. (The fourth task, called the Efficiency task, is more similar to the 2009 Thorough task; it returns a ranked list of elements which may overlap.) For each query, the 2009 RiC task returns, in document order, all its focused elements grouped by document. The 2010 RiC task is similar but can be viewed as a form of snippet retrieval in that the focused elements retrieved should have fewer than 300 irrelevant characters between them. The metrics supplied by INEX for this task are F-Score and T21(300) [8], and final evaluation is given in terms of MAgP. The RRiC task is a variant of RiC in that a maximum of 500 characters per article is retrieved. We use the top-ranked focused element from each article as the source of that character string. This task also uses the F-Score and T21 metrics with MAgP for evaluation. The RF task of 2010 is similar to the Focused Task of 2009; it returns a rank-ordered set of focused elements in response to a query but in this case, only 1000 characters per topic may be returned. Evaluation is in terms of set-based precision over the retrieved characters (char_prec) [8].

All of these 2010 focused retrieval tasks were performed again, using the new organization and indexing of the 2009 Wikipedia collection, for each focusing method (section, child, and correlation). Results were evaluated and compared with those produced by the top-ranked teams in each case. Suffice it to say, in each case, our methods produced a task result that would rank in the top-ten for that task. Significance tests were performed as well. We also reran the 2009 Focused tasks based on the new organization of the data and again, all results ranked in the top ten when evaluated. (See [1, 2, 9] for details.) We plan to produce a paper describing these experiments and their results in the near future. But results to date convinced us that our method of producing focused elements was sound and could be used as a basis for snippet generation.

3 INEX 2011: Snippet Generation

By definition, a snippet is designed to provide the user with information sufficient to determine the relevance of a document without viewing the document itself, thus quickly allowing the user to identify what she is looking for [7]. Given a query, dynamic element retrieval produces, for each document, a rank-ordered list of overlapping elements. Our goal for this year centered on generating snippets based on focused elements. To facilitate this goal, we first apply the correlation focusing strategy to focus the element set and then select the highest ranking focused element as the basis for the snippet representing that document (the raw snippet). The snippet is then generated using the snippet refinement algorithm described below.

Given the raw snippet selected above, for each sentence in the element, initialize the score of the sentence to 0. If the length of the sentence is greater than a defined minimum, for each query term (substantive word type) in the query, increment the sentence score by the frequency of that term in the sentence. The final score of the sentence is the score divided by its length. The sentences are sorted in decreasing order and concatenated in that order to form a paragraph. The first 300 characters of the paragraph form the corresponding snippet.

This approach was used to generate the snippets for the INEX 2011 Snippet Retrieval Track. Evaluation produced a score (based on the geometric mean of recall and negative recall) of 0.5606, which ranked 9 in the list of top 10 scores reported for this track out of the 50 runs recorded.

4 Conclusions

We conclude from the results of the task evaluation that our approach to snippet generation is soundly based. However, this is very early work and much remains to be done. Additional investigation is needed to gain perspective on how good snippets are generated.

References

1. Acquilla, N.: Improving results for the INEX 2009 and 2010 Focused tasks. M.S. Thesis, Department of Computer Science, University of Minnesota Duluth (2011), http://www.d.umn.edu/cs/thesis/acquilla.pdf
2. Banhatti, R.: Improving results for the INEX 2009 Thorough and 2010 Efficiency tasks. M.S. Thesis, Department of Computer Science, University of Minnesota Duluth (2011), http://www.d.umn.edu/cs/thesis/banhatti.pdf
3. Crouch, C.: Dynamic element retrieval in a structured environment. ACM TOIS 24(4), 437–454 (2006)
4. Crouch, C., Khanna, S., Potnis, P., Doddapaneni, N.: The Dynamic Retrieval of XML Elements. In: Fuhr, N., Lalmas, M., Malik, S., Kazai, G. (eds.) INEX 2005. LNCS, vol. 3977, pp. 268–281. Springer, Heidelberg (2006)

5. Crouch, C.J., Crouch, D.B., Bhirud, D., Poluri, P., Polumetla, C., Sudhakar, V.: A Methodology for Producing Improved Focused Elements. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 70–80. Springer, Heidelberg (2010)
6. Crouch, C., Crouch, D., Vadlamudi, S., Cherukuri, R., Mahule, A.: A Useful Method for Producing Competitive Ad Hoc Task Results. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 63–70. Springer, Heidelberg (2011)
7. INEX 2011 Preproceedings, https://inex.mmci.unisaarland.de/static/proceedings/INEX2011preproceedings.pdf
8. INEX website, https://inex.mmci.unisaarland.de/
9. Narendravarapu, R.: Improving results for the INEX 2009 and 2010 Relevant-in-Context tasks. M.S. Thesis, Department of Computer Science, University of Minnesota Duluth (2011), http://www.d.umn.edu/cs/thesis/narendravarapu.pdf
10. Salton, G. (ed.): The Smart System—Experiments in Automatic Document Processing. Prentice-Hall (1971)
11. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Comm. ACM 18(11), 613–620 (1975)
12. Singhal, A., Buckley, C., Mitra, M.: Pivoted document length normalization. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Zurich, Switzerland, pp. 21–29 (1996)

RMIT at INEX 2011 Snippet Retrieval Track

Lorena Leal Bando, Falk Scholer, and James Thom

RMIT University, Melbourne, Australia
{lorena_lealbando,falk.scholer,james.thom}@rmit.edu.au

Abstract. This report describes our participation in the Snippet retrieval track. Snippets were constructed by first selecting sentences according to the occurrence of query terms. We also used a pseudo-relevance feedback approach in order to expand the original query. Results showed that a large number of extra terms may harm sentence selection for short summaries. However, simple heuristics that employ query term occurrence information can benefit considerably sentence retrieval.

1 Introduction

In many IR systems the standard response after submitting a query consists of a ranked list of results. Each one of these results is presented as a summary with three textual key elements: the title, the snippet and the URL. The retrieved documents are returned by the IR system because they have a certain similarity with the users' query terms. However, not all documents in the answer list are likely to actually be relevant for a user. Therefore, users carry out a triage process, selecting which documents they wish to view in full by scanning these key elements. The snippet is generally the most indicative component when users need to review multiple documents for fulfilling their information needs. Snippets are short fragments extracted from the document, and their aim is to provide a glimpse of the document content. Common practices for constructing snippets include the selection of either metadata information, leading sentences of a document, or sentences containing query terms. Our approach focuses on the latter method.

The INEX initiative launched the Snippet Retrieval Track to study not only system retrieval, but also snippet generation effectiveness. We describe our experiments and results for this latter task.

2 Methodology

Given that we did not participate in the system retrieval task, we used a baseline ranked run distributed by the track organizers. It involved, for each test topic, the first 500 Wikipedia articles retrieved by applying the BM25 similarity function ($K1 = 0.7, b = 0.3$). The snippet generation task consists of constructing succinct summaries for those documents. Snippets were limited in terms of length to not exceed 300 characters.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 300–305, 2012.

In the following subsections, we briefly describe the collection and topics for the track. We have divided the conducted experiments in two parts: *query expasion* and *snippet generation*.

2.1 Collection and Topics

Documents for the snippet track are part of the INEX Wikipedia collection. This collection is a snapshot of Wikipedia constructed in 2008 of English articles which are enriched with semantic annotations (obtained from YAGO). We did not use those annotations for query expansion or snippet creation experiments, so all markup was ignored from documents. We removed stopwords and applied the Porter stemming algorithm [4] to the remaining terms.

Each topic of the track includes the following fields: title, castitle, phrasetitle, description and narrative. For our experiments, we only used the title terms as they resemble information requests from typical real users. Stopword removal and stemming were also applied to these terms.

2.2 Query Expansion

Query expansion is a technique that attempts to address the potential vocabulary mismatch between users and authors. That is, users may choose different terms to describe their information needs than authors use when creating documents. Query expansion introduces new and possibly closely related terms to an original query, thus enlarging the set of results. This technique has been explored in terms of retrieval effectiveness of IR systems [2,6,9]. However, we employed query expansion to source extra terms for extractive summarisation approaches that we explain latter.

We followed Rocchio's approach [5] for expanding the initial query.

$$\boldsymbol{Q_1} = \alpha \cdot \boldsymbol{Q_0} + \frac{\beta}{|R|} \sum_{d \in R} \boldsymbol{d} - \frac{\gamma}{|\overline{R}|} \sum_{d \in \overline{R}} \boldsymbol{d} \tag{1}$$

As can be seen in Equation 1, the influence of the original query ($\boldsymbol{Q_0}$), relevant (R) and irrelevant ($\overline{R}$) documents can be adjusted by the α, β, γ parameters, respectively. For our experiments the value of α was set with a low value to only choose terms that were different from the original query ($\alpha = -1000$). The value of γ, in contrast, was set to 0 since we did not have negative feedback (or irrelevant documents) from the ranked list of results. Consequently, the value of β can be set to any non-zero value as this will affect the final result in a constant way, in this case its value was defined as 8. Thus, $\boldsymbol{Q_1}$ will contain a new set of terms.

Given that a reference run was provided, we applied Rocchio's formulation by using the R top ranked documents for each topic. Subsequently, we selected the leading E tokens as expansion terms. We called this set "Rocchio terms". Based on previous experiments using a newswire collection, we fixed R to a value of 5, and tried two settings for E: 25 and 40.

The original query was expanded by concatenating the additional Rocchio terms. However, some refinements were applied to the supplementary terms. Numbers were discarded, except numbers of exactly four digits, since we assumed that those may refer to years.

2.3 Snippet Generation

It has been shown that the presence of query terms in short summaries positively influences the finding of relevant documents [7,8]. If the snippet lacks the key words provided by users, they are generally less able to detect whether the underlying document is relevant or not.

Previous research on extractive summarisation has explored the selection of sentences for succint summaries [1,3]. The advantage of employing sentences is that they convey simple ideas. Following this approach, Wikipedia articles were segmented into sentences. Nevertheless, it was difficult to properly identify sentence boundaries due to having multiple contributors with different writing styles. For example, sentence boundaries become ambiguous when a sentence includes abbreviations such as *"Dr. Smith"*. We did not carry out an analysis of abbreviations to address this issue in detail.

We processed Wikipedia articles before constructing snippets. Specifically, information contained inside the <*title*> and <*bdy*> was used to narrow the document content. We suggest that snippets should include information of the document itself instead of sources pointing to other articles. Therefore, the *Reference* section was ignored in our summarisation approaches. With the remaining text, we scored sentences according to the occurrence of query terms in them by using Equation 2.

$$qb_i = \frac{(\text{number of unique query terms in sentence } i)^2}{\text{total number of query terms}} \tag{2}$$

For all snippets, the title of the document was concatenated to the first ranked sentences. It should be noted that snippets should not exceed 300 characters according to the track requirements. Top sentences were presented in snippets depending on document length. In case an article contained less than 3 sentences, the snippet algorithm only provided the first 100 characters of each sentence. In this setting we assumed that the document was very short and was unlikely to provide any relevant information; therefore the summary was reduced. On the contrary, the leading 150 characters (or less) of the first ranked sentence were displayed, when the document had more than 3 sentences. Subsequent highly scored sentences were cut to 100 characters. This is done until the summary length is reached.

Submitted runs employed the query-biased approach for ranking document sentences in three different settings. The run labelled as "baseline" generated snippets given the title words of each topic, that is, no expansion was applied. The runs defined as "expanded-40" and "expanded-25" used expanded queries with $E = 40$ and $E = 25$ respectively, for calculating the query-biased score.

Using a newswire collection in a former study, we found that 40 extra terms were sufficient for enlarging the original query. Given the differences among collections, we also experimented with another E value by reducing it to 25.

3 Results

Results of our runs are shown in Table 1. For our three submitted runs, the baseline (p23-baseline) not using query expansion performed the best, as measured by the geometric mean of recall and negative recall (GM), the official track measure. Adding more expansion terms (p23-expanded-25 and p23-expanded-40) hurts the performance on the selection of sentences for constructing the summary. However, we observed a slight improvement when a larger number of extra terms are employed. Further analysis will determine whether the differences are statistically significant. Moreover, all runs were more effective than the median run, based on the GM measure.

Table 2 provides results for other metrics that were used in the track, and the rank of our runs according to each metric. We observed that our baseline run without query expansion achieves better performance for many metrics, except for Negative Recall and Negative Agreement. Negative Recall measures the percentage of irrelevant documents correctly assessed, so our technique of exclusively employing query terms requires to be complemented with other heuristics. Moreover, it can be seen that in terms of Recall and Positive Agreement the addition of 40 extra terms to the query benefits sentence ranking on a small scale compared with a reduced number of supplementary terms.

As explained previously, we applied 40 extra terms given that this value performed well in tests using a newswire collection. However, it appears that this tuning parameter varies substantially across collections, for example a good value for newswire articles does not lead to improvements for Wikipedia articles. Thus, more experimentation is required in this regard.

It should be noted that the sentence ranking algorithm ignores the *"References"* section of a Wikipedia article for constructing a summary. However, the process for adding additional terms includes this article section. We assume that this could introduce noisy terms. Further analysis will be conducted to investigate differences when specific parts of a document are employed for expansion.

Table 1. Snippet track final results based on GM

Run	GM	Rank
p23-baseline	0.5505	2
p23-expanded-25	0.5239	12
p23-expanded-40	0.5294	7
Top run	0.5705	1
Median run	0.4805	–

Table 2. Snippet track final results based on MPA (Mean prediction accuracy), MNPA (Mean normalised prediction accuracy), Recall, NR (Negative Recall), PA (Positive agreement) and NA (Negative agreement)

Run	MPA	MNPA	Recall	NR	PA	NA
p23-baseline	0.7958	0.6441	0.4035	0.8848	0.3715	0.8667
Rank	15	2	7	16	3	15
p23-expanded-25	0.7936	0.6233	0.3597	0.8870	0.3319	0.8685
Rank	16	13	20	14	17	11
p23-expanded-40	0.7766	0.6231	0.3866	0.8596	0.3389	0.8514
Rank	27	14	12	33	13	27
Top run	0.8144	0.6444	0.4641	0.9116	0.3748	0.8873
Median run	0.7860	0.6167	0.3544	0.8762	0.3161	0.8587

4 Conclusions

Wikipedia articles are similar to newspaper reports, as they tend to condense the main important points in the leading sentences of those documents. However, in our first approach we observe that simple sentence ranking mechanisms towards a query perform well in terms of GM. For our second approach that employs expanded queries, the snippet may contain a low or null rate of original query terms. We suppose that for this reason assessors did not find the summaries useful or related to the topic. In this regard, we will study the mechanisms for weighing expansion terms gradually, and minimize negative effects of presenting sentences without the initial query terms.

References

1. Brandow, R., Mitze, K., Rau, L.F.: Automatic condensation of electronic publications by sentence selection. Information Processing & Management 31, 675–685 (1995)
2. Buckley, C., Salton, G., Allan, J., Singhal, A.: Automatic query expansion using SMART: TREC 3. In: Overview of the Third Text REtrieval Conference (TREC-3), pp. 69–80 (1995)
3. Luhn, H.P.: The automatic creation of literature abstracts. IBM Journal of Research and Development 2, 159–165 (1958)
4. Porter, M.F.: An algorithm for suffix stripping. Program: Electronic Library and Information Systems 4, 130–137 (1980)
5. Rocchio, J.J.: Relevance feedback in information retrieval. In: The SMART Retrieval System:Experiments in Automatic Document Processing, pp. 313–323 (1971)
6. Salton, G., Buckley, C.: Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science 41, 288–297 (1990)
7. Tombros, A., Sanderson, M.: Advantages of query biased summaries in information retrieval. In: Proceedings of the 21st Annual International ACM SIGIR Conference, pp. 2–10. ACM (1998)

8. White, R.W., Jose, J.M., Ruthven, I.: A task-oriented study on the influencing effects of query-biased summarisation in web searching. Information Processing & Management 39, 707–733 (2003)
9. Xu, J., Croft, W.B.: Query expansion using local and global document analysis. In: Proceedings of the 19th Annual International ACM SIGIR Conference, pp. 4–11. ACM (1996)

Topical Language Model for Snippet Retrieval

Rongmei Li and Theo van der Weide

Radboud University, Nijmegen, The Netherlands

Abstract. In this paper we describe our participation in the INEX 2011 snippet retrieval track. Based on the reference run of ranked documents, we compute topical language models and then derive snippets that will be used for relevance judgment of their corresponding documents. Our snippets are represented as a bag of words or a cluster of (semi) sentences. Our findings are: 1) the relevance of word snippet is difficult to judge; 2) topical language model and stopword removal do improve recall marginally; 3) visual representation of topical words can help understanding the relevance of topical aspects.

1 Introduction

INEX offers a framework for cross comparison among content-oriented XML retrieval approaches given the same test collections and evaluation measures. The INEX snippet retrieval track is to determine how best to generate informative snippets for search results. Such snippets should provide sufficient information to allow the user to determine the relevance of each document, without needing to view the document itself. It simulates part of the common search behavior on many commercial Web search engines (e.g. Google and Yahoo). For instance, the user clicks the link of a relevant result based on the sneak preview of the document contents or the information provided by the corresponding snippet.

As in 2009 and 2010, in 2011 the same English Wikipedia with XML format is used in the snippet retrieval track. The retrieval topics are selected from previous years that were created by the INEX participants to represent real life information needs. As in 2009, each topic consists of five fields. The `<title>` field (Content Only, or CO query) is the same as the standard keyword query. The `<castitle>` field (Content And Structure, or CAS query) adds structural constraints to the CO query by explicitly specifying where to look and what to return. The `<phrasetitle>` field (Phrase query) presents explicitly a marked up query phrase. The `<description>` and `<narrative>` fields provide more information about topical context. Especially the `<narrative>` field is used for relevance assessment.

Following our previous years' work [1], [2], we adopted the language modeling framework for retrieval task. Our snippet retrieval contains two steps: 1) relevant document ranking; 2) snippet extraction. The retrieval result is either a cluster of topical terms or fragments of text extracted from the document content [6].

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 306–314, 2012.

This paper documents our primary and official results in the INEX 2011 snippet retrieval track and focuses on examining the effectiveness of using topical language model.

The rest of the paper is organized as follows. We elaborate the topical language model in section 2. Section 3 briefly introduces how relevant documents are ranked. Various strategies of snippet extraction are given in section 4. The evaluation of retrieval result and analysis are presented in section 5. The paper is concluded in section 6.

2 Topical Language Model

Each Web page or document may consist of several topics. Each topic is described by the terms with a high semantic correlation. These terms occur more often in relevant Web pages than in irrelevant Web pages. For instance, Web pages about *education* will have many more terms like school, department, courses, and exams than Web pages about other topics. At the same time, there are general terms (e.g. *stop words*) that appear in most documents and terms that are too specific (e.g. *acknowledgment*) and appear only in certain relevant documents. They are either less discriminative or too biased. In both cases they contribute less to distinguish a relevant document from others with regards to a query. To eliminate these terms from language models, the so-called parsimonious language model has been introduced [4]. This model can facilitate the computation of topical models capturing the similarity of language usage as the likelihood $P(t|T)$ of a term t being relevant for topic T.

The parsimonious model uses the EM-algorithm to estimate the term distribution $P(t|D)$ in a document D. At the *expectation step*, for each term the expectation score e_t is computed using term frequency $tf(t, D)$ and term probability $P(t|D)$ for that document, and the term probability of the whole document collection $P(t|C)$. The general terms will have a smaller expectation score since they have relatively higher probabilities $P(t|C)$ in the background model. The algorithm is smoothed using the Jelinek-Mercer (JM) method [5] with a fixed smoothing factor μ, which allows us to further reduce the expectation score for the general terms. At the *maximization step*, the expectation score is normalized and compared to a given threshold. Terms having higher score will be preserved in the pruning process. Some general terms are eliminated from the next iteration as their normalized expectation score is low. This selection process continues till the maximized term distribution $P(t|D)$ does not change significantly anymore. Getting rid of terms that are common in general English, the resulting model thus has fewer terms than the standard model with full text indexing. In other words, the parsimonious model preserves specific terms that appear in a document frequently but relatively less often in the whole collection.

$$\text{expectation step}: \quad e_t = tf(t, D) \cdot \frac{\mu P(t|D)}{\mu P(t|D) + (1-\mu)P(t|C)} \tag{1}$$

$$\text{maximization step}: P(t|D) = \frac{e_t}{\sum_t e_t}, \text{i.e. normalize the model} \tag{2}$$

Table 1. Examples of Topical Models

vitiligo pigment disorder cause treatment		**france second world war normandy**	
the	0.036212	the	0.087846
and	0.025080	of	0.032329
vitiligo	0.024885	in	0.018172
in	0.018974	normandy	0.008875
is	0.013187	german	0.007675
skin	0.007773	from	0.006805
patches	0.005832	war	0.006348
be	0.005609	allied	0.006070
for	0.005209	with	0.006006
his	0.004722	operation	0.005828
people	0.004171	world	0.005269
erythema	0.003888	that	0.004972
white	0.003850	forces	0.004887
may	0.003765	1944	0.004308
nalp1	0.003499	day	0.004274
disease	0.003492	june	0.003451
which	0.003292	battle	0.003360
dermatitis	0.003110	british	0.003336
hair	0.003106	germans	0.003034
pmid	0.003095	beach	0.003022
document ID: 65847		**document ID: 6723726**	

We hypothesize that the resulting parsimonious language model preserves the main topics of a document interest and thus name it a topical model. As an example, the language models on the topics *vitiligo pigment disorder cause treatment* and *france second world war normandy* are given in Table 1. They are computed for the English Wikipedia article 65847 and 6723726 separately using the entire Wikipedia 2009 collection as the background collection. Both articles rank the top 1 in its own topic given by the reference document ranking. In this snippet retrieval task, the smoothing factor is 0.9 and threshold is 0.0001 for topical model construction. The resulting topical models consist of a list of terms with the new probability generated by the considered article that excludes general terms.

3 Document Ranking

Our previous study [3] found that a topical language model improves document ranking for ad-hoc retrieval. In this work, our attention is on snippets that are extracted and generated from the provided ranked list of documents. For each query, approximately 500 top relevant documents are pre-retrieved from the Wikipedia 2009 collection using the BM25 ranking function, a well-known high performance state-of-the-art retrieval model. In this paper we examine the

relevance and the rank correspondence between these documents and the snippets that are generated by our approaches in section 4.

4 Snippet Creation

The English Wikipedia is a single Web domain with encyclopedic articles that are written in an objective style, with little redundancy of information between articles. Being different from the Web pages that are created by some people, Wikipedia articles have a clear focus on a single and dedicated topic. In most of cases, these Wikipedia articles are relatively short. In case of relatively longer articles, sub-topics are also rather co-related. With the one-topic observation, the article topic can be modeled by the parsimonious language model. The snippet of an article is the representation of this topic. It is expected to provide the most important information on which the user is able to judge if the underlying article is relevant to their information needs.

In our experiments, we hypothesize that users recognize certain combinations of terms in created snippets which are related to their information need. We automatically extract snippets using terms as the minimal unit. Each term is weighted according to the relative occurrence in its article and in the entire Wikipedia 2009 collection. The top K scoring terms are chosen for inclusion in the snippet. The term-extraction based snippets are then represented differently to the user. One is a cluster of words that indicates the described topic. Another is a cluster of semi-sentences that contains the topic information while preserving some language structure. The detailed snippet creation process is given in the following sub-sections.

4.1 Word Snippet

The topical language model consists of a list of document terms and corresponding probability of document generation (see Table 1). These terms present a small range of topics. The probability presents the degree of relevance of each term to the covered topic. In the example of the topical model for *france second world war normandy*, terms such as Normandy, German, war, operation, forces, battle, have higher probability on the described topic. Based on a list of such terms extracted from a underlying document, the user can have a reasonable judgment on the topic of its original. Following this hypothesis, we can create a word snippet for each relevant document that contains most probable topical terms. An example word snippet of the relevant Wikipedia article (document ID: 21201) for the topic of *nobel prize* (topic ID: 2011011) looks like the following:

```
the nobel in of prize and for to a prizes 2007 peace by physics
awarded are or is have accessed on was chemistry award alfred
swedish with norwegian who nobelprize be org as medicine committee
that laureates shall november his at physiology foundation awards
it literature medal medals 10 not
```

The snippet terms are ordered descendingly based on their probability of document generation in the topical language model. For instance, the term *the* has a higher probability to be generated by document 21201 than the term *nobel* for the topic of *nobel prize*.

4.2 Extracted Document Phrase

The Wikipedia articles are well-tagged with XML fields. These fields have different importance to its topic. The most informative fields include <title>, <category>, <name>, <website>, <description>, <recipient> and <p>. The example document phrases of article (21201) on the topic *nobel prize* are taken from the corresponding XML fields:

```
<title>Nobel Prize</title>
<category>Science and engineering awards</category>
<category>Organizations based in Sweden</category>
<category>Awards</category>
<category>Nobel Prize</category>
<name>Infobox award</name>
<website>http://nobelprize.org</website>
<description>Outstanding contributions in
<recipient>Physics</recipient>
</description>
```

The document phrases within less informative XML fields are filtered out. The example of filtered XML fields are as follows:

```
<id>21201</id>
<timestamp>2008-10-12T01:29:51Z</timestamp>
<username>G913</username>
```

The example concatenated document phrases of article 21201 on *nobel prize* are the following:

```
Nobel Prize Science and engineering awards Organizations based
in Sweden Awards Nobel Prize Infobox award http://nobelprize.org
Outstanding contributions in Physics, Chemistry, Literature,
Peace and Physiology or Medicine. The Sveriges Riksbank Prize in
Economic Sciences in Memory of Alfred Nobel, commonly identified
with the Nobel
```

4.3 Extracted Semi-sentence Snippet

Considering the cognitive impact and the described topics, we can bring the document structure into our word snippet to create another representation. Instead of being ordered by probability of document generation, the terms with higher probability are selected from chosen document paragraphs while scanning through them. We would like to compare how the different representations affect the relevance judgment of the original document. An example semi-sentence snippet of Wikipedia article 21201 looks like the following:

```
nobel prize science and awards in sweden awards nobel prize award
nobelprize org outstanding contributions in physics chemistry
literature peace and physiology or medicine the sveriges riksbank
prize in economic sciences in memory of alfred nobel identified
with the nobel prize is awarded for outstanding contributions in
economics the nobel
```

4.4 Stopword Removal

A well-focused document snippet should contain much relevant information of its original document but as little irrelevant information as possible. Due to the limited relevant information provided by a single original document, the resulting topical model of one document still has some stop words (e.g. *and*, *in*, *the*) that should be removed from the final result. The example of word snippet with stopword removal of the same article looks like the following:

```
nobel prize prizes 2007 peace physics awarded accessed chemistry
award alfred swedish norwegian nobelprize org medicine committee
laureates shall november physiology foundation awards literature
medal medals 10 mathematics 2006 discovery sciences five
discoveries economics members december made years nobels copyright
sweden three pauling linus nominations academy won october
stockholm shared
```

5 Results

We submit 6 runs that are officially manually evaluated by human beings. All runs use reference document ranking to retrieve snippets of the corresponding ranked documents. They are named in the following way:

first field: abbreviation of tasks (namely SRT11)
second field: used strategy

- ParsTerm: ranked topical terms from topical language model
- ParsStopTerm: ranked topical terms with stopword removal
- ParsDoc: extracted semi-sentence snippet that contains only topical terms
- ParsStopDoc: extracted semi-sentence snippet that contains only topical terms and without stop words
- DocParsedTXT: extracted document phrases that removes punctuation
- DocTXT: extracted document phrases

5.1 Evaluation Metrics

All results of snippet retrieval are evaluated by participants manually. The assessment result of snippet retrieval is compared with the existing document-based relevance judgments, which are treated as a ground truth. The purpose of the evaluation is to examine how effective the snippets are at providing the user with sufficient information to determine the relevance of the underlying document.

Table 2. Evaluation results of all official runs

runs	performance metrics							
	MPA	PA	NA	Recall	NR	MNPA	GM	rank/all
SRT11DocParsedTXT	0.8026	0.2900	0.8737	0.2982	0.9113	0.6047	0.4351	33/41
SRT11DocTXT	0.8092	0.3333	0.8672	0.3513	0.8896	0.6204	0.4803	22/41
SRT11ParsDoc	0.7992	0.3007	0.8721	0.3231	0.9026	0.6128	0.4557	29/41
SRT11ParsStopDoc	0.7988	0.2725	0.8676	0.3235	0.8938	0.6086	0.4157	36/41
SRT11ParsStopTerm	0.7690	0.2227	0.8360	0.2721	0.8695	0.5708	0.3458	40/41
SRT11ParsTerm	0.7830	0.2171	0.8544	0.2431	0.8977	0.5704	0.3392	41/41

1. Mean Precision Accuracy (MPA) can be considered the raw agreement between two assessors - one who assessed the actual documents (i.e. the ground truth relevance judgments), and one who assessed the snippets.
2. Positive agreement (PA) is the conditional probability that, given one of the assessors judges a document as relevant, the other will also do so.
3. Negative agreement (NA) is the conditional probability that, given one of the assessors judges a document as irrelevant; the other will also do so.
4. Recall is the percentage of relevant documents that are correctly assessed.
5. Negative recall (NR) is the percentage of irrelevant documents that are correctly assessed.
6. Mean normalized prediction accuracy (MNPA) is the arithmetic mean of recall and negative recall. It calculates the rates for relevant and irrelevant documents separately, and averages the results, to avoid relevant results being weighted higher than irrelevant results.
7. The primary evaluation metric, which is used to rank the submissions of all participants, is the geometric mean of recall and negative recall (GM).

A high value of GM requires a high value in recall and negative recall, i.e. the snippets must help the user to accurately predict both relevant and irrelevant documents. If a submission has high recall but zero negative recall (e.g. in the case that everything is judged relevant), GM will be zero. Likewise, if a submission has high negative recall but zero recall (e.g. in the case that everything is judged irrelevant), GM will be zero.

5.2 Evaluation Results

Based on GM performance, SRT11DocTXT is our best run, which is the collection of document sentences from the pre-defined most interesting elements. It ranks position 22 out of all 41 submissions. Our worst run is the word snippet (SRT11ParsTerm) with 0.3392 GM score and ranks the last of all submissions.

Our best run (SRT11DocTXT) ranks 16 out of 41 in case of MNPA score (0.6204) while SRT11ParsTerm remains the worst.

SRT11DocParsedTXT has the 2nd highest NR with 91.13% accuracy on predicting irrelevant documents among all submissions. However, its recall ranks only 32th of 41 with 29.82% accuracy on predicting relevant documents.

Among all of our submissions, assessors are most certain about the positive snippet result of our best run (SRT11DocTXT) according to MPA score (0.8092) and PA score (0.3333). The same run is also the 2nd that wins the raw agreement between two assessors compared to all 41 submissions. Our worst run (SRT11ParsTerm) ranks 23th of 41 on raw agreement (MPA=0.7830). However, the PA of the worst is nearly the lowest except the other run having 0.2081 PA score.

In regard to NA, our best run has relatively good agreement on irrelevance judgment of assessors with 14th rank among all 41 submissions. Instead of our worst run, another run (SRT11ParsStopTerm) is the most difficult one for irrelevance judgment. It ranks 35th of all 41 submissions with 0.8360 NA score.

5.3 Analysis

During the evaluation process, we find that word snippet is the most difficult one to judge. The reasons include: 1) some general terms and stop words are still in the snippet so that the topical context is rather scattered; 2) other assessors may not agree with our assumption of "one-article one-topic". As a result, it is very difficult for them to "create" a single "story" out of a bag of words; 3) many queries (topics) in this task are very specific and have high restriction on the scope of topic. The bag of words cannot provide sufficient information on the information requirement.

The evaluation difficulty of word snippets is also reflected in the PA and NA score. They are relatively low compared to other submissions. It is not surprising that these runs get lowest recall and good NR.

Different from word snippets, the relevance judgment on (semi)-sentence snippet or document phrases has lower controversy among assessors. Based on the same document ranking, non-semantic information (e.g. punctuation) is able to reduce wrong rejection while improving recall.

The topical language model and stopword removal have little positive influence on recall. It can focus the assessor's attention on the topic. However, it hurts recall when the query has too much restriction.

During the assessment, we noticed that the representation of snippet results affects the assessor's attention and understanding on the relevance of topical aspects. For instance, when the article title is represented as a separate paragraph with highlighting as "title", the snippet is more easily accepted than when it is in line with the rest of the description.

Additionally, we also found that when the to-be-judged snippets contain too little words or too little language structure (e.g. word snippet), the agreement of assessors is too low to be trusted.

6 Conclusion

In this paper, we present our results for the snippet retrieval track of INEX 2011. We use the reference document ranking to extract snippets. We compute the so-called "topical language model" for each article and construct final snippets

at the granularity of words, semi-sentences, and document phrases. Our best result is *extracted document phrases* while the worst run is *word snippet*. The topical model and stopword removal contribute little to retrieval performance. One reason for the failure of word snippet is due to the difficulty of judgment. Good snippet representation and language structure have positive impact on relevance judgment.

References

1. Li, R., van der Weide, T.: Extended Language Models for XML Element Retrieval. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) INEX 2010. LNCS, vol. 6932, pp. 89–97. Springer, Heidelberg (2011)
2. Li, R., van der Weide, T.: Language Models for XML Element Retrieval. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 95–102. Springer, Heidelberg (2010)
3. Hiemstra, D., Li, R.M., Kamps, J., Kaptein, R.: Parsimonious Language Models for a Terabyte of Text. In: Proceedings of TREC (2007)
4. Hiemstra, D., Robertson, S., Zaragoza, H.: Parsimonious Language Models for Information Retrieval. In: Proceedings of SIGIR (2004)
5. Zhai, C.X., Lafferty, J.: A Study of Smoothing Methods for Language Models Applied to Information Retrieval. ACM Trans. on Information Systems 22(2), 179–214 (2004)
6. Liu, X.Y., Croft, W.B.: Passage Retrieval Based on Language Models. In: Proceedings of CIKM (2002)

JUFE at INEX 2011 Snippet Retrieval Track

Dexi Liu*, Changxuan Wan, Guoqiong Liao, Minjuan Zhong, and Xiping Liu

School of Information Technology, Jiangxi Key Laboratory of Data and Knowledge Engineering, Jiangxi University of Finance and Economics, Nanchang, China
{Dexiliu,lewislxp}@gmail.com, wanchangxuan@263.net, liaoguoqiong@163.com, lucyzmj@sina.com

Abstract. Jiangxi University of Finance and Economics (JUFE) submitted 8 runs to the Snippet Retrieval Track at INEX 2011.This report describes an XML snippet retrieval method based on Average Topic Generalization (ATG) model used by JUFE. The basic idea of the ATG is that different element in an XML document plays different role and hence should has distinguishing importance. The ATG model sets a weight automatically to each element according to its tag or path in the XML document. Then, the BM25EW model based on the ATG is proposed to retrieve and rank the relevant elements in an XML document collection. All windows in the most relevant elements are scored and those windows with higher scores are extracted as snippets. By comparing with the runs under different strategies, the performance are discussed and analyzed in detail.

Keywords: XML snippet retrieval, ATG model, BM25EW model, Window.

1 Introduction

In XML information retrieval, queries performed on XML search engines usually return far more results than a user could ever imagined. Although all XML search engines are try to return the relevant results and place the most relevant results first, lots of irrelevant results are still returned. Query-oriented XML document summarization can help with this problem. In addition to help users directly access the required information, a snippet, a kind of summary, also contributes to decide whether a result is relevant to a query or not and improve retrieval efficiency greatly.

The goal of the Snippet Retrieval Track is to determine how to generate the best snippet in a XML document. Such snippet can provide sufficient information to allow the users to determine the relevance of its underlying document, instead of viewing the document itself, which can help the users find what they want quickly. In this work, a snipped retrieval method based on the Average Topic Generalization (ATG) model is proposed, which got the first grade in INEX 2011 Snippet Retrieval Track. In the method, the elements in an XML document are weighted automatically firstly, then the most relevant elements to the query are retrieved by using the BM25EW model, and finally the snippets are extracted from the retrieved elements.

* Corresponding author.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 315–324, 2012.

The rest of the paper is organized as follows. In Section 2, we introduce an XML snippet retrieval method based on the ATG model, including the simple description of the ATG model, the BM25EW model based on ATG and the method of XML snippet extraction. Experiment results are discussed and analyzed in Section 3. In Section 4, we conclude our works and present the future works.

2 XML Snippet Retrieval Method Based on the ATG Model

The ATG model [1] is a tag or path weight assignment model. It not only reduces heavy burden from experts, but also solves the problem induced by subjective guess and misjudgments. The ATG model is based on the following observation that, in an XML document, if an element summarizes the major contents of the whole document, it is more important in the document and hence has greater impact on the search results than the elements which summarize minor contents. For example, the *title*, *keywords*, *abstract*, and *section title* in a document are more generally related to the document topic than the *paragraph*, *figure*, *table*, and *reference*. Therefore, if a query term occurs in the elements of *title*, *keywords*, *abstract*, or *section title* in some documents, these documents or elements are commonly more relevant to the query. Nevertheless, it is not easy to infer just from the literal words of tags or paths representing *title*, *keywords*, and so on.

In an XML document, the content in every element is related to the document topic in a certain extent and it can be regarded as a kind of generalization (summary) of the entire document. It is evident that the elements with different tags/paths or different elements with the same tags/paths have different generalization strengths. The ATG weight of tag t (or path p) means the average strength that all elements with tag t (or path p) generalize the document that it occurs, which is measured on the whole collection. Statistically, the more documents are used, tag weights calculated by ATG model are more reasonable. However, only 250 XML documents are sampled for the ATG weight because it is difficult to finish one of the steps in the ATG model, singular value decomposition, on the whole collection containing millions of documents.

Based on the ATG model, an XML snippet retrieval strategy is put forward in this paper, which is divided into two stages, i.e., XML element retrieval and snippet extraction.

2.1 The BM25EW Model Based on ATG

The BM25EW model is an improved BM25 model, which makes full use of element weights (tag weights or path weights) based on the ATG model. In general, Okapi BM25 [2] is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, which is defined as Formula (1):

$$s(D) = \sum_{t \in Q} \log\left(\frac{N+1}{df_t + 0.5}\right) \cdot \frac{(k_1+1) f_{D,t}}{f_{D,t} + k_1\left(1 - b + b\frac{|D|}{avgdl}\right)} \cdot \tag{1}$$

where, N is the number of documents in the collection, df_t is the number of documents containing term t, $f_{D,t}$ is the frequency of term t in document D, $|D|$ is the length of D in words, $avgdl$ is the average document length in the text collection from which documents are drawn, and k_1 and b are free parameters.

In XML retrieval, instead of the documents, the elements are retrieved. So, the elements can be regarded as a kind of special documents when we use BM25 model. The main difference between the BM25EW model and the classical BM25 model is how to calculate the frequency of term t in element E or document D. Considering the element weights, in the BM25EW model, $f_{D,t}$ in Formula (1) is replaced by $f_{E,t}$, which is defined as Formula (2):

$$f_{E,t} = \sum_{leaf \in E} tf_{leaf,t} EW(leaf) . \tag{2}$$

where $tf_{leaf,t}$ is the frequency of term t in the leaf node $leaf$, and $EW(leaf)$ refers to the path weight or tag weight of $leaf$, which is computed based on the ATG model. We set parameters k_1=1.2 and b=0.3 in this work.

2.2 XML Snippet Retrieval and Extraction

XML Element Retrieval. We use BM25EW model mentioned above to retrieve XML elements. The queries used in this work are generated automatically from *title* and *phrasetitle* fields in the given topics. For example, the query in Fig. 1 is expressed as {*nobel*, *prize*, "*nobel prize*"}. We limit the returned answer elements as {*section*, *bdy*, *caption*, *sec*, *ss*1, *ss*2, *ss*3, *ss*4, *ss*5, *p*}.

```
<inex-topic-file>
<topic id="2011011" ct_no="11">
<title>Nobel prize</title>
<castitle>//article[about(., Nobel prize)]</castitle>
<phrasetitle>"Nobel prize"</phrasetitle>
<description>information about Nobel prize</description>
<narrative>I need to prepare a presentation about the Nobel prize. Therefore, I want to collect information about it as much as possible. Information, the history of the Nobel prize or the stories of the award-winners for example, is in demand. </narrative>
</topic>
…
</inex-topic-file>
```

Fig. 1. An example of query topic

Snippet Extraction: The basic idea of snippet extraction is as following:

(1) If the XML document have the *title* field, the *title* field is selected priorly as a part of the snippets since the title of a document can illustrate clearly the relevance between the document and the query;

(2) If a fragment (a sentence or a sequential of terms with definite length) can reflect the query, and also can describe the XML elements or the entire XML document content, it is suitable to be a part of the snippets;

(3) If a fragment has been contained in other fragments in the snippet, it is not necessary to extract it;

(4) By analyzing terms occurrence, we can get a great deal of valuable information, such as the relationship between a fragment and a query, the relationship between a fragment and the context of XML document/elements, and information coverage degree among different fragments.

Let XML element $e=<t_1, \ldots, t_n>$, $1 \le k \le n$, query $Q=\{q_1, \ldots, q_m\}$, $1 \le i \le m$, where n and m are the number of terms in e and Q respectively, and t_k in e occurs in leaf node $leaf_{t_k}$.

According to the basic idea of snippet extraction, there are three factors to determine whether term t is suitable to be a snippet: the relevance between t and query Q, the informativity of t and the weight of the element where t occurs.

In this work, the relevance between t and Q is determined by their distance in e. If t occurs close to a query term q, it means t and q are relative. The closer t and q occur, the more relative they are. Formula(3) is used to measure the relevance between t_k and Q.

$$relevance(Q, t_k) = \sum_{j=1}^{n} \frac{\delta_{t_j}}{|j-k|+1}. \tag{3}$$

$$\text{where } \delta_{t_j} = \begin{cases} 1, \text{ if } t_j \in Q \\ 0, else \end{cases}$$

The informativity of term t is computed by its inverse document frequency (IDF). The element weight, i.e., $EW(leaf_{t_k})$, is gotten from the ATG model.

The scores of candidate snippets are calculated after the three factors are obtained. Supposing $window_i=<t_i, t_{i+1}, \ldots, t_{i+l-1}>$ is a window in e with the length of l, the score of the $window_i$ depends on the contained terms, and it is defined as Formula(4).

$$score(window_i) = \frac{1}{l} \sum_{k=i}^{i+l-1} relevance(Q, t_k) \cdot IDF_{t_k} \cdot EW(leaf_{t_k}). \tag{4}$$

Before the $window_i$ is added to snippet s, it is necessary to check whether the content of $window_i$ is already contained in s. In this work, 2-gram coverage rate is used to measure the information coverage degree. Supposing $s=<t_1, \ldots, t_L>$is a snippet, the coverage rate of s to $window_i=<t_i, t_{i+1}, \ldots, t_{i+l-1}>$ is defined as Formula (5).

$$Cover(s, window_i) = \frac{\text{the number of same 2-gram between } window_i \text{ and } s}{l\text{-}1} \tag{5}$$

The snippet extraction algorithm is shown as Fig. 2.

3 Experiment Results and Analysis

In this section, we present and discuss the evaluation results of our runs in the Snippet Retrieval Track of INEX 2011.

Input: data set C, query Q, snippet length L=300, the number of relevant documents to be returned M=500.

Output: snippets set S, each element in S including score r related to query, XML document no. d, and snippet s guiding users for relevant judgment.

1 $S=\phi$;
2 Sample from C and compute path and tag weights using the ATG model.
3 Retrieve and rank elements based on BM25EW model. The result is denoted by E.
4 For each element $e \in E$
5 Supposing the score of e is r, the id of the document where e occurs is d. Let s equal to the title field of document d.
6 If the snippet of document d does not exist in S, insert $\zeta=(r, d, s)$ to S.
7 If length of s is less than L then continue step 8 to step 11.
8 Rank all windows in e based on formula (4). The window size is set to 15 in this work.
9 For each $window \in e$
10 If the information coverage of s to $window$ less than 0.6, namely $cover(s,window)<0.6$, the $window$ is appended to s.
11 If length of s is less than L then continue step 9 to step 11
12 If the number of elements in S less than M, or the length of any snippet is less than L, continue step 4 to step12

Fig. 2. Snippet retrieval algorithm

3.1 Introduction of Snippet Retrieval Track in INEX 2011 [3]

The task of Snippet Retrieval Track is to return a ranked list of documents for the requested topic, and together with each document, a corresponding snippet. The users can get the relevance of the underlying document based on snippets, instead of viewing the document. Each run is allowed to return no more than 500 documents for each topic, together with a maximum of 300 characters per snippet. The INEX Wikipedia collection introduced in 2009 (called as Wikipedia 2009 in this paper) with 2,666,190 documents is used and the topics are selected from the INEX 2009 Ad Hoc Track. Furthermore, 500 reference documents for each topic are retrieved by the BM25 model, which allows participants without using their own search engines to generate snippets directly from these documents. We denote the XML documents returned by the reference runs as Reference-Run-dataset.

3.2 Evaluation Metrics

To determine the effectiveness of the returned snippets, the method of manual assessment is used. Each topic in a submission is assigned to an assessor. After understanding the details of the topic, the assessor reads through the top 100 returned

snippets, and judges the relevance of the underlying documents based on the snippets he/her reads. Submissions are evaluated by comparing the agreements between the snippet-based relevance judgments and the existing document-based relevance judgments. If the agreement of the snippet-based relevance judgments is higher, the snippets are more effective to help the users to make correct judgments and hence the performance of the corresponding snippet retrieval system is better.

Given a query, a document may be classified as "relevant" or "not relevant" with it. For a real "relevant" document, the assessor may either make a correct judgment and label it as "relevant" after reading the snippet, or make a wrong judgment by labeling it as "not relevant" due to the misleading of the snippets. In a similar way, for a real "not relevant" document, the assessor may also make a correct or a wrong judgment according to the snippets. The metrics used in this track are as following:

Mean Precision Accuracy (MPA):

$$MPA = \frac{TP + TN}{TP + FP + FN + TN} \tag{6}$$

Positive Agreement (PA):

$$PA = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{7}$$

Negative Agreement (NA):

$$NA = \frac{2 \cdot TN}{2 \cdot TN + FP + FN} \tag{8}$$

Mean Normalized Prediction Accuracy (MNPA):

$$MNPA = 0.5 \frac{TP}{TP + FN} + 0.5 \frac{TN}{TN + FP} \tag{9}$$

Recall and Negative Recall (NR):

$$\mathrm{Re}\,call = \frac{TP}{TP + FN} \tag{10}$$

$$NR = \frac{TN}{TN + FN} \tag{11}$$

Geometric Mean of recall and negative recall averaged over all topics (GM):

$$GM = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{TN + FP}} \tag{12}$$

where *TP* refers to the number of the "relevant" documents judged correctly by the assessor, *TN* is the "not relevant" document number with correct judgment, *FP* stands for the "relevant" document number with wrong judgment by the assessor, and *FN* means the "not relevant" document number with wrong judgment.

Among above metrics, the Geometric Mean of recall and negative recall is the primary evaluation metric, which is used to rank the submissions.

3.3 Experiment Setting and the Results

As a contrast, 8 runs were submitted, named p72-LDKE-$m_1m_2m_3m_4$, where m_i ($1<i<4$) denote four different strategies when a snippet is generated. Strategy 1 is dataset selection: using Reference-Run-dataset (m_1=0) or Wikipedia 2009 dataset (m_1=1). Strategy 2 is snippet selection: using baseline method (m_2=0) or window method described in Fig. 2 (m_2=1). According to the baseline method, after the candidate elements being scored and ranked, only the first 300 characters are extracted as the snippet from elements with highest score. The remain part of this snippet are extracted from the successive elements in the ranked candidate element list in case of the precedents are not long enough. While in the window method, every window that contain 15 terms are scored and those with higher scores are extracted as a part of the snippet. Strategy 3 is whether the ATG element weight (tag weight, m_3=1 or path weight, m_3=2) is used or not (m_3=0) in the processes of element retrieval and snippet extraction. The element retrieval model used in our system is based on BM25EW. Strategy 4 is whether reordering the XML documents according to the reference runs (m_4=0) or not (m_4=1) after elements being retrieved.

The top 10 runs according to GM metric are listed in Tab.1, in which our runs are highlighted. The evaluation results by other metrics for our runs are listed in Tab. 2.

Based on the analysis of the evaluation results, we can draw the following six conclusions.

Table 1. The top 10 runs in the Snippet Retrieval Track, ranked by GM score

Rank	Run	GM Score
1	p72-LDKE-1111	0.5705
2	p23-baseline	0.5505
3	p72-LDKE-0101	0.5472
4	p20-QUTFirst300	0.5416
5	p73-PKU_ICST_REF_11a	0.5341
6	p72-LDKE-1110	0.5317
7	p23-expanded-40	0.5294
8	p72-LDKE-0111	0.527
9	p65-UMD_SNIPPET_RETRIEVAL_RUN_3	0.5264
10	p20-QUTFocused	0.5242

Table 2. The evaluation results by other metrics for our runs (numbers in the brackets are the ranks according to corresponding metrics)

Run	MPA	MNPA	Recall	NR	GM	PA	NA
p72-LDKE-1111	0.7582(39)	0.643(3)	0.4641(1)	0.8219(40)	0.5705(1)	0.3748(1)	0.8292(39)
p72-LDKE-0101	0.759(38)	0.6331(4)	0.4347(3)	0.8314(37)	0.5472(3)	0.3647(4)	0.8301(38)
p72-LDKE-1110	0.7684(33)	0.6328(5)	0.447(2)	0.8187(41)	0.5317(6)	0.3646(5)	0.8232(41)
p72-LDKE-0111	0.7674(34)	0.6179(19)	0.4058(6)	0.8299(39)	0.527(8)	0.3452(10)	0.8364(34)
p72-LDKE-1121	0.786(21)	0.6263(8)	0.3888(10)	0.8637(27)	0.5192(13)	0.3279(18)	0.8587(21)
p72-LDKE-1101	0.7638(37)	0.622(15)	0.4115(4)	0.8325(36)	0.513(14)	0.3329(16)	0.8313(37)
p72-LDKE-1001	0.7726(29)	0.6161(23)	0.369(19)	0.8633(28)	0.4919(19)	0.3191(20)	0.8434(31)
p72-LDKE-1011	0.8022(9)	0.6076(27)	0.3269(26)	0.8884(12)	0.477(25)	0.3151(22)	0.8715(5)

Conclusion 1. The proposed snippet retrieval method based on the ATG weights obtains high performance. The extracted snippet can guide users to determine the relevance of each document, instead of viewing the whole document. In p72-LDKE-1111, the four strategies are set as: selecting Wikipedia 2009 dataset as the collection, extracting those windows with higher scores as snippets, using tag weight as element weight, and sorting the elements by BM25EW.

Conclusion 2. The XML element retrieval model BM25EW, which uses the ATG model to obtain the tag weights, outperforms BM25 model according to GM evaluation metric. It can be shown from two aspects. The one is that GM value of p72-LDKE-1111 (0.5705), using the BM25EW model directly on Wikipedia 2009 dataset, is much higher than that on Reference-Run-dataset (p72-LDKE-0111, 0.527); the other is that the GM value decreases from 0.5705 to 0.5317 (p72-LDKE-1110) after reordering the results of BM25EW according to Reference-Run-dataset.

Conclusion 3. The score strategy based on window granularity (p72-LDKE-1111 and p72-LDKE-1101) is effective, which scores each window in elements first and then select those with higher score as the returned snippets. Compared with the direct extraction of the former *L* bytes of the elements (p72-LDKE-1011and p72-LDKE-1001), the gained GM values through window strategy are increased from 0.477 and 0.4919 to 0.5705 and 0.513 respectively.

Conclusion 4. The performance on XML element retrieval and snippet extraction is significantly improved due to the adoption of the element weight. The GM value gained by the strategy without the element weight is 0.513 (p72-LDKE-1101), while it increases to 0.5705 after using the tag weight as the element weight (p72-LDKE-1111). Interestingly, the path weight gets higher performance for the Ad Hoc task on Wikipedia 2008 dataset, but for snippet retrieval task on Wikipedia 2009 dataset, the performance almost does not change after using the path weights. Their GM values are 0.5192 (p72-LDKE-1121) and 0.513 (p72-LDKE-1101) respectively. As the introduction of the ATG model, only 250 XML documents are sampled from the collection when calculating the ATG weights. However, after analyzing on the dataset, we find that the path in the sample of 250 documents from Wikipedia 2009 dataset cannot cover the path in the whole collection, which leads to most path weight missing. Here we give the quantitative description:

Supposing T is the whole tag set in dataset C and $T' \subseteq T$ is the tag set in the sampling dataset. If T' is used to replace T, some tags may not be accessed for they are not existed in T'. We name this part of tags is not covered by T'. The tag coverage rate of T' on C is defined as follows:

$$Cover(T', C) = \sum_{d \in C} |T'_d| \Big/ \sum_{d \in C} |T_d| \tag{12}$$

where $|T_d|$ is the total number of tags in document d and $|T'_d|$ refers to the number of tags covered by T'.

In a similar way, $|P_d|$ is the total number of the path in document d, $|P'_d|$ refers to the number of paths covered by P'. The path coverage rate of P' on C is also defined as:

$$Cover(P',C) = \sum_{d \in C} \left| P_d' \right| \Big/ \sum_{d \in C} \left| P_d \right| \tag{13}$$

Table 3 is the path and tag coverage rate of the sampled 250 documents on Wikipedia 2008 dataset and Reference-Run-dataset (a subset of Wikipedia 2009, including 24,971 documents). It can be seen that the path coverage rate only 6.44% on Reference-Run-dataset, which can explain why the path weight has almost no effect in our runs.

Table 3. Coverage rate of tags/paths in the sampling dataset

	T	T'	$\sum_{d \in C} \lvert T_d \rvert$	$\sum_{d \in C} \lvert T_d' \rvert$	tag coverage
Wikipedia 2008	1258	33	9,000,149	8,259,998	91.78%
Reference-Run-dataset	9469	299	1,891,176	1,406,898	74.39%
	P	P'	$\sum_{d \in C} \lvert P_d \rvert$	$\sum_{d \in C} \lvert P_d' \rvert$	path coverage
Wikipedia 2008	66210	53	12,329,436	10,120,278	82.08%
Reference-Run-dataset	2503335	184	4,267,833	274,913	**6.44%**

Conclusion 5. The retrieval performance on Reference-Run-dataset (p72-LDKE-0101, 0.5472) is better than on the entire collection of Wikipedia 2009 (p72-LDKE-1101, 0.513) when the element weight is not used. However, it remains to be further analyzed to find out why the element weight cannot improve the snippet retrieval performance on Reference-Run-dataset (p72-LDKE-0111, 0.527). A possible reason is that the BM25EW, which should be used on the whole dataset, is not appropriate for Reference-Run-dataset, which is the retrieval result of the BM25 model.

Conclusion 6. Compared with other runs, XML snippet retrieval based on the ATG model has the advantage in guiding users to judge the relevant documents. The goal of a Web or an XML search engine is to find more relevant documents instead of filtering irrelevant documents. If the snippet can help the users to determine the "relevant" documents correctly, the reading burden will be reduced effectively. Two metrics, Recall and PA, both measure the judging agreement of the "relevant" documents between the snippet assessors and the document assessors (standard answer), by which p72-LDKE-1111 ranked first respectively. However, the strategy proposed in this paper is not competitive when guiding the users to judge the "irrelevant" documents.

4 Conclusions and Further Work

This paper describes an XML snippet extraction strategy based on the ATG model consisting of three steps. Firstly, the tag or path weight is assigned according to the

ATG model. Secondly, based on the BM25EW model, the XML elements are retrieved and ranked. Thirdly, the windows with fixed length in the elements are scored, and those windows with higher values are returned to user under the constraint of less information redundancy. The experiment results from INEX 2011 evaluation confirm the effectiveness.

In the further, we will try to find out a suitable parameter training method in the ATG model and an appropriate windows size setting in snippet extraction. Furthermore, how to apply effectively the path weights will also be explored.

Acknowledgements. This paper is supported by Natural Science Foundation of China (No. 60803105, 60763001 and 61173146) and Science & Technology Project of Department of Education of Jiangxi Province (No.GJJ09649).

References

1. Liu, D., Wan, C., Chen, L., Liu, X.: Automatically Weighting Tags in XML Collection. In: 19th ACM Conference on Information and Knowledge Management, pp. 1289–1292. ACM Press, New York (2010)
2. Robertson, S.E., Walker, S., Beaulieu, M.: Okapi at TREC–7: Automatic Ad Hoc, Filtering, VLC and Interactive Tracks. In: 7th Text Retrieval Conference, pp. 253–264. NIST Special Publication 500-242 (1999)
3. Trappett, M., Geva, S., Trotman, A., Scholer, F., Sanderson, M.: Overview of the INEX 2011 Snippet Retrieval Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 283–294. Springer, Heidelberg (2012)

Indian School of Mines at INEX 2011 Snippet Retrieval Task

Sukomal Pal and Preeti Tamrakar

Dept of CSE, Indian School of Mines, Dhanbad India
{sukomalpal,tamrakarpreeti89}@gmail.com

Abstract. This paper describes the work that we did at Indian School of Mines, Dhanbad towards Snippet Retrieval for INEX 2011. During official submissions, we pre-processed the XML-ified Wikipedia collection to a simplified txt-only version. This collection and the reference document run were used as inputs to a simple Snippet Retrieval system that we developed. We submitted 3 runs to INEX 2011. Post submission we apply a passage retrieval technique based on a Language Modelling approach for snippet retrieval. The performance of our submissions at the INEX SR task was moderate, but promising enough for further exploration.

1 Introduction

The usual way of interacting with an IR system is to enter a specific information need expressed as a query. In response, the system provides a ranked list of retrieved documents. For each of these retrieved documents, the user is typically provided by the system a title and a few sentences from each of these documents. These few sentences are called a *snippet* and considered as an "excerpt" of the document so that the user can decide which of the retrieved documents are more likely to meet his/her information need. Ideally, a snippet should be short yet informative enough so that this decision can be made without having to refer to the full document text.

1.1 Snippet Retrieval (SR)

Retrieving the snippets from the whole documents is known as *Snippet Retrieval*. The goal of the Snippet Retrieval track is to study the methods of generating informative snippets for search results. Such snippets should provide sufficient information to allow the user to determine the relevance of each document, without needing to view the document itself. In other words, snippets should be a short summary of the document.

However the summary can be generated in two following ways as given in Manning and Schütze [1]:

a) Static Summary: These are always the same regardless of the query. A static summary generally comprises either or both a subset of the document and

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 325–330, 2012.

metadata associated with the document. The simplest form of summary takes the first two sentences or 50 words of a document, or extracts particular zones of a document, such as the title and author.

b) Dynamic Summary: These are customized according to the user's information need as deduced from a query. Dynamic summaries display one or more 'windows' on the document, aiming to present the pieces that have the most utility to the user in evaluating the document with respect to their information need. Dynamic summaries are generally regarded as greatly improving the usability of IR systems, but they present a complication for IR system design.

Though summary generation techniques are often used for snippet retrieval and the problems are seen and attacked from the same angle, they are not the same. While summaries are always coherent text, snippets are not necessarily always. Snippets can be a set of meaningful phrases without forming complete sentence(s).

1.2 SR in of INEX

This is the first year of SR track at INEX. The organizers provided a test collection consisting of documents and topics for the track.

Participants are asked to either use their own system to retrieve a ranked set of snippets *or* can use the result of a reference run provided by the organizer as a seed to their snippet retrieval system [2].

We took the second approach i.e. we found snippets from the set of documents returned by the reference run. We submitted 3 runs in INEX 2011 Snippet Retrieval track.

The paper is organized as follows. In the next section, we review the works done in this field. In section 3 we describe test data followed by our approach. We discuss results in section 4. Finally we conclude with scope of future work.

2 Related Work

Summary generation dates back to summarization activities of early TRECs (Text REtrieval Conferences) [3]. The documents to be summarized were articles of the Wall Street Journal (WSJ). In order to decide which aspects of the articles would provide utility to generating a summary, their characteristic were examined in a small scale study. The methodology that was followed involved examining 50 randomly selected articles from the collection and attempting to extract conclusions about the distribution of important information within them. Their title, headings, leading paragraph, and the overall structural organization were studied. This sample collection was used for experimentation with various system parameters, in order to approximate the best settings for the summarization system. Although the sample of the documents was small, there was a strong uniformity in the characteristics of the samples that allowed for a generalization of the conclusions to the entire collection.

Tombros and Sanderson [4] presented the first in-depth study showing that properly selected query-dependent snippets are superior to query-independent summaries with respect to speed, precision, and recall with which users can judge the relevance of a hit without actually having to follow the link to the full document. In this work, we take the usefulness of query-dependent result snippets for granted but could produce only the static snippets during official submissions.

3 Data Used

Track organizers provided the test collection with documents, topics and a document level refernce run.

3.1 Documents

The INEX Snippet Retrieval 2011 corpus is a part of the whole Wikipedia XML collection containing 2, 666, 190 XML-ified documents. This is a XML version of the English Wikipedia based on dump taken on October 8, 2008 and semantically annotated. Uncompressed, the corpus size is about 50.7 GB containing images and other multimedia data [5].

3.2 Topics

The topics were recycled from INEX 2009 Ad Hoc track avoiding the 'easy topics'. The topic set provided contained 50 short *Content-Only (CO)* queries (2011011 − 2011060) comprising topic id, title, castitle, phrasetitle, description and narrative.

3.3 Reference Run

Our focus was to see the performance of snippet retrieval for a given document run. We therefore used the result of the document retrieval the organizers already provided. The reference document run (*2011_SR_reference_run*) was generated using BM25 (parameters $K_1 = 0.7b = 0.3$, s-stemmer). The run contained a list of 500 documents per topic ranked in decreasing order of probability of relevance as computed by the IR system at the document level.

4 Approach

We took the reference run as our starting point. However we were not in a position to handle the whole INEX collection having very detailed and lengthy annotations. We considered the text-only version of the documents for snippet generation. We divided the task in the following two steps:

4.1 xml-to-txt Conversion

The document collection was parsed using LIBXML2[1] parser and the XML tags were removed from all the XML documents. Also there were a lot of white spaces and control characters which were stripped off.

4.2 Snippet Generation

For each document per topic in the reference run, we generated one single snippet. We observed that the first few lines of each document provide an overall introduction to the content of the whole documents. As a naive approach we started with extraction of first 300 characters of each document appearing in the reference run as the snippets. We submitted three runs based on this notion. Each run contained snippets for all the 50 topics with each topic having a maximum of 500 snippets. The first two runs had some file access errors which was rectified in the third run.

Post-submission, however, we applied passage retrieval technique from the XML documents (without *xml-to-txt* conversion) using Language modelling approach implemented in *Lemur-indri 5.2* [2] for snippet generation (default *Dirichlet smoothing* with $\mu = 2500$), which are yet to be evaluated and compared with our official submissions.

5 Results

As a naive approach, our results were moderate as per the geometric mean of recall and negative recall which was used as the official metric of INEX 2011 (shown in Table 1).

Table 1. Geometric Mean of Recall and Negative Recall

Run-id	Score	Rank
p35-97-ism-snippet-Baseline-Reference-run_01	0.4886	20/41
p35-98-ism-snippet-Baseline-Reference-run_01	0.4800	23/41
p35-ism-snippet-Baseline-Reference-run_02	0.4365	32/41
p72-LDKE-1111 (BEST performer)	0.5705	1/41

Table 2 describes our performance on other metrics where the metrics are defined as follows [2]:

$$\begin{aligned} \text{MPA} &= \text{Mean Prediction Accuracy} \\ &= \text{fraction of results correctly predicted, average over all topics} \\ &= (TP + TN)/(TP + FN + TN + FP) \end{aligned}$$

[1] `http://www.xmlsoft.org`
[2] `www.lemurproject.org/indri`

Table 2. Our Performance based on other metrics

Run-id or Rank	MPA	MNPA	Recall	NR	GM	PA	NA
p35-97-ism-snippet-Baseline-Reference-run_01	0.8056	0.6245	0.3534	0.8956	0.4886	0.3448	0.8706
rank (out of 41)	4	12	22	7	20	11	8
p35-98-ism-snippet-Baseline-Reference-run_01	0.8042	0.6165	0.3348	0.8982	0.4800	0.3161	0.8712
rank (out of 41)	6	22	24	5	23	21	7
p35-ism-snippet-Baseline-Reference-run_02	0.8090	0.5984	0.2875	0.9093	0.4365	0.2850	0.8764
rank (out of 41)	3	30	36	3	32	32	2

$$
\begin{aligned}
\text{MNPA} &= \text{Mean Normalised Prediction Accuracy} \\
&= \text{average of the relevant and irrelevant results correctly predicted averaged over all topics} \\
&= 0.5 * TP/(TP + FN) + 0.5 * TN/(TN + FP) \\
\text{Recall} &= \text{fraction of relevant documents, averaged over all topics} \\
&= TP/(TP + FN) \\
\text{NR} &= \text{Negative recall} \\
&= \text{fraction of irrelevant documents, averaged over all topics} \\
&= TN/(TN + FP) \\
\text{PA} &= \text{Positive agreement} \\
&= \text{conditional probability of agreement between snippet assessor and document assessor (i.e. ground truth), given that one of the two judged relevant} \\
&= 2 * TP/(2 * TP + FP + FN) \\
\text{NA} &= \text{Negative agreement} \\
&= \text{conditional probability of agreement between snippet assessor and document assessor (i.e. ground truth), given that one of the two judged irrelevant} \\
&= 2 * TN/(2 * TN + FP + FN)
\end{aligned}
$$

Overall, with a minimal setup our performance was encouraging. Our post-submission effort based on a passage retrieval using Language modelling however needs evaluation and comparison with other submitted runs which we look forward to do.

6 Conclusion

In this paper we described a simple approach of extracting a few lines from the top of documents as snippets. Though this is not the best snippets for all documents, often they provide a quick overview on the detailed content of the documents. Our initial results were quite promising and can act as the baseline.

We need to explore other techniques and compare them with the baseline. After official submissions, we have attempted to apply passage retrieval technique based on Language Modelling approach. Once evaluation scripts are available we can evaluate the results obtained and tune the parameters used towards further improvement. Also we plan to explore other techniques and models for snippet retrieval in the coming days.

References

1. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge (1999)
2. Trappett, M., Geva, S., Trotman, A., Scholer, F., Sanderson, M.: Overview of the INEX 2011 Snippet Retrieval Track. In: Geva, S., Kamps, J., Schenkel, R. (eds.) INEX 2011. LNCS, vol. 7424, pp. 283–294. Springer, Heidelberg (2012)
3. Harman, D.: The text retrieval conferences (trecs). In: Proceedings of a Workshop on held at Vienna, Virginia: May 6-8, TIPSTER 1996, 373–410. Association for Computational Linguistics, Stroudsburg (1996)
4. Tombros, A., Sanderson, M.: Advantages of query biased summaries in information retrieval. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1998, pp. 2–10. ACM, New York (1998)
5. Schenkel, R., Suchanek, F.M., Kasneci, G.: Yawn: A semantically annotated wikipedia xml corpus. In: BTW, pp. 277–291 (2007)

PKU at INEX 2011 XML Snippet Track

Songlin Wang, Yihong Hong, and Jianwu Yang*

Institute of Computer Sci. & Tech., Peking University,
Beijing 100871, China
{wang_sl05,hongyihong,yangjw}@pku.edu.cn

Abstract. Snippets are used by almost every text search engine to complement ranking scheme in order to effectively handle user's searching, which are inherently ambiguous and whose relevance semantics are difficult to assess. In this paper, we present our participation in the INEX 2011 Snippet track. A efficiently retrieval system has been used, by which the semi-structured of document has been considered, and we present a snippet generate system, which effectively summarize the query results and document content, according to which users can quickly assess the relevance of the query results.

Keywords: XML Retrieval, XML-IR, Snippet, Query.

1 Background

Each result in the results list delivered by current search engines contains a short document snippet, and the snippet gives the user a sneak preview of the document contents. Accurate snippets allow users to make good decisions about which results are worth accessing and which can be ignored.

INEX 2011 XML Snippet track contains two parts, XML document retrieval and snippet generation task. XML retrieval presents different challenges of retrieval in text documents due to the semi-structured nature of the data, the goal is to take advantage of the structure of explicitly marked up documents to provide more focused retrieval results. The goal of the snippet generation is to generate informative snippets for search results; such snippets should provide sufficient information to allow the user to determine the relevance of each document, without needing to view the document itself.

The corpus is a subset of the Wikipedia corpus with 144,625 documents, and the snippet retrieval track will use the INEX Wikipedia collection. Topics will be recycled from previous ad hoc tracks. Participating organizations will submit a ranked list of documents, and corresponding snippets. Each submission should contain 500 snippets per topic, with a maximum of 300 characters per snippet.

This paper is organized as follows. In section 2, we show the retrieval system. Section 3 describes the approach taken to the snippet task. The paper ends with a discussion of future research and conclusion in section 4.

* Corresponding author.

S. Geva, J. Kamps, and R. Schenkel (Eds.): INEX 2011, LNCS 7424, pp. 331–336, 2012.

2 Retrieval System

In snippet track, we direct two parts, the document retrieval and the snippet generation. We first retrieve each element of articles using CO queries. The retrieval model is based on Okapi BM25 scores between the query model and the document (element) model that is smoothed using Dirichlet and Jelinek-Mercer priors.

2.1 Retrieval Model

The Vector Space Model is the basic model of this research. The vector space model represents each document and query as an n-dimensional vector of unique terms. The terms are weighted based on their frequency within the document. The relationship of a document to a query is determined by the distance between the two vectors in vector space. The closer the two vectors, the more potentially relevant the document is (cosine similarity is one of the measures used to compute the distance).

Our retrieval engine is based on the Vector Space Model. We used a matrix to represent the document, and vector for each element of the document.

2.2 BM25 Score Formula

Our system is based on the BM25 weights function.

$$Score(Q, e_i) = \sum_{j=1}^{m} IDF(q_j) \times \frac{tf(q_j, e_i) \times (k_1 + 1)}{tf(q_j, e_i) + k_1 \times (1 - b + b \times \frac{dl}{avgdl})} \quad 2\text{-}1$$

$$IDF(q_j) = log \frac{N - n(q_j) + 0.5}{n(q_j) + 0.5} \quad 2\text{-}2$$

With:

- $tf(q_j, \mathrm{e_i})$: the frequency of keyword q_j in elemente$_\mathrm{i}$.
-N: the number of articles in the collection.
-$n(q_j)$: the number of articles containing the term q_j.
-$\frac{dl}{avgdl}$: the ratio between the length of element e_i and the average element length.
-k_1 and b: the classical BM25 parameters where we set k_1=0.25, b=0.15 based on the experiments.

Parameter k_1 is able to control the term frequency saturation. Parameter b allows setting the importance of $\frac{dl}{avgdl}$.

2.3 Document Score Based on Subtree

Each document has lots of elements, and the formula for the document is defined as formula [2-3] and [2-4].

$$Score(d) = D(m) \cdot \sum_{i=1}^{m} Score(Q, e_i) \cdot elementweight(Q, e_i) \quad 2\text{-}3$$

$$D(m) = \begin{cases} D_{single} & (m = 1) \\ D_{multiple} & (m > 1) \end{cases} \quad 2\text{-}4$$

With:

-$Score(Q, e_i)$: the score of element e_i in document.
- $elementweight(Q, e_i)$: the weight of element e_i in the document.
-$D(m)$: the reduced via the element decay.

2.4 Weight of the Element

We used the INEX2010 croup as a learning set and split the elements into two part, relevant elements and irrelevant elements, and we can get each element weight using information gain.

$$Gain(e) = -\sum_{i=1}^{2} P(c_i) log\ P(c_i) \quad 2\text{-}5$$

With:

-c_i : relevant or irrelevant classification.

2.5 Pseudo Feedback

Since synonymy has an impact on the recall of most information retrieval systems, we use Pseudo Feedback method to expand the query of the topics in this track.

First a query with the recognized phrase is submitted to the retrieval system, and the system will do the first run to rank document and pick the top ranked 50 documents. These top ranked documents are assumed to be relevant by the retrieval system and are combined with the original query through query expansion to do the second run. The retrieval system presents newly ranked documents to the user. And the score of the words that expanded form this method is defined as formula [2-6] and [2-7].

$$Score(w|Q) = \sum idf(w) idf(q_i) log \frac{df}{|D|+1} \quad 2\text{-}6$$

$$idf(w) = log \frac{N - n(w) + 0.5}{n(w) + 0.5} \quad 2\text{-}7$$

With:

-$|D|$: the number of terms in document *D*.
-df: the frequency of term *w* in document *D*.

2.6 The Distribution of Keywords

The passage contain more different keywords will be more relevant. The position of the keywords also will impact the relevant between query and document. We used the SLCA algorithm to get the smallest subtrees contain all keywords, and in this subtree, we calculated the score of query Q to the position x by formula 2-9.

$$score(i,Q) = \sum_{k=1}^{|Q|} c(kw_k, j) exp\,[\frac{-(i-k)^2}{2\partial^2}] \qquad 2\text{-}8$$

With:

-$c(kw_k, j)$: the value of keyword kw_k in the position j.

For example,

```
<sec>
    <st> the mountains in Indian</ st>
    <p> …(list of mountains and their distribute)</p>
</sec>
```

We can make sure that the content in element <p> is the retriever need. When one element contains keywords of the query, its brother node is also important. When generate the snippet of the document, the weight of this kind element (<p> as above example) should be increased.

3 Snippet Generation

In the snippet generation system, we used query relevance, significant words, title/section-title relevance and tag weight to evaluate the relevance between sentences and a query. The sentences with higher relevance score will be chosen as the retrieval snippet.

3.1 Query Relevance

The relevance between a query and sentences largely depends on the query terms in a sentence. Function [3-1] is one example to calculate the relevance.

$$Score_{query}(s) = queryC(s) * \sum_{i=1}^{n} Occ(q_i, s) * Weight(q_i) \qquad 3\text{-}1$$

$$Weight(q_i) = idf(q_i) \qquad 3\text{-}2$$

With:

- $queryC(s)$: the number of query terms category in sentence s.
- $Occ(q_i, s)$: the occurrence frequency of query terms q_i in sentence s.

- $Weight(q_i)$: the weight of query term q_i.
- $idf(q_i)$: the IDF value of term q_i in the database.

3.2 Significant Words[3]

The frequency of significant words was used to help evaluate the relevance between a query and a sentence. A word is defined as a significant word if it is a non-stop word and its term frequency is larger than a threshold *T*. *T* is defined as function [3-3].

$$T = 7 + I * 0.1 * |L - n| \quad 3\text{-}3$$

With:

- *n*: the number of sentences in the document.
- *L*: *L* is 25 for *n* < *25* and 40 for *n* > *40*.
- *I*: *I* is 0 for 25<=n<=40 and 1 otherwise

The significant words score is based on the function [3-4]

$$Score_{sw}(s) = SW/TSW \quad 3\text{-}4$$

Where *SW* is the number of significant words in the sentence and *TSW* is the total number of words in the sentence.

3.3 Title Relevance

Since the title is the best summary of a document, the sentence with more title terms is highly probably relevant to the query. The title score of the sentence can be calculated using the function [3-5].

$$Score_{title}(s) = T/N \quad 3\text{-}5$$

Where *T* is the number of title words in the sentence *s* and *N* is the number of title words.

3.4 Sentence Score

The function [3-6] is the formula used to calculate the sentence score.

$$Score_{rel}(s) = \alpha Score_{query}(s) + \beta Score_{sw}(s) + \gamma Score_{title}(s) \quad 3\text{-}6$$

Where$\alpha = 0.7,\ \beta = 0.15,\ \gamma = 0.15$ based on the experiment.

Additionally, we use a section-title weight and a tag-weight calculated in the document retrieval system to help evaluate the sentence relevance. If a sentence is in a section whose title contains the query words, the sentence may be more relevant to the query. The function [3-7] synthesizes all the factors mentioned above, and generates the final sentence relevance score.

$$Score(s) = \left(1 + \mu * stW(s) + \sigma * tagW(s)\right) * Score_{rel}(s) \quad 3\text{-}7$$

With

-*stW(s)*: The frequency of query terms in the section-title contain sentence s.
-*tagW(s)*: The tag weight of the tag contain the sentence s.

4 Conclusions

From the special focus on exploiting structural characteristics of XML document collections, we retrieve XML documents based on both document structure and content. We have learned the weight of elements based on the cast of INEX2010 to enhance the retrieval performance, and we also consider the distribution of the keywords in the documents and elements. In the snippet generation system, we use query relevance, significant words, title/section-title relevance and tag weight to evaluate the relevance between sentences and a query. The sentences with higher relevance score will be chosen as the retrieval snippet.

Acknowledgment. The work reported in this paper was supported by the National Natural science Foundation of China Grant 60642001 and 60875033.

References

1. Fuhr, N., Lalmas, M., Malik, S., Kazai, G. (eds.): INEX 2005. LNCS, vol. 3977. Springer, Heidelberg (2006)
2. Weerkamp, W.: Optimizing structured document retrieval using focused and relevant in context strategies. Master's thesis, Utrecht University, The Netherlands (2006)
3. Wang, C., Jing, F., Zhang, L., Zhang, H.-J.: Learning Query-Biased Web Page Summarization. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management (2007)
4. Goldsteiny, J., Kantrowitz, M., Mittal, V., Carbonelly, J.: Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1999)
5. Chuang, W.T., Yang, J.: Extracting Sentence Segments for Text Summa-rization: A Machine Learning Approach. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2000 (2000)

Author Index