RMIT University

Celebrating 120 years

ADCS 2007

Proceedings of the Twelfth Australasian Document Computing Symposium

10 December 2007

Edited by
Amanda Spink, Andrew Turpin and Mingfang Wu

NICTA

Proceedings of the Twelfth Australasian Document Computing Symposium

The Melbourne Zoo,
10 December 2007

Published by
School of Computer Science and Information Technology,
RMIT University,
Melbourne VIC 3001, Australia.

Editors

Amanda Spink
Andrew Turpin
Mingfang Wu

ISBN: 978 0 646 48437 2

`http://www.cs.rmit.edu.au/~aht/adcs2007`

Proceedings of the Twelfth Australasian Document Computing Symposium
Melbourne
10 December 2007

Chair's Preface

These proceedings contain the papers of the Twelfth Australasian Document Computing Symposium hosted by the School of Computer Science and Information Technology at RMIT University, and held at the Melbourne Zoo.

The two keynote talks, ten papers and nine posters reflect the breadth of interest of the Australian research community in the area of document computing. It is also a highlight of ADCS this year that we are not only collocated with The Australasian Language Technology Workshop 2007, but are sharing a paper session, keynote talk, and social functions with the Australian natural language research community.

Of the 18 full papers submitted, 10 were accepted for presentation as full papers (55%), 5 were accepted as posters (28%), and 3 were rejected (17%). Of the 6 short papers submitted, 4 were accepted for presentation as posters (67%).

All full papers received at least three anonymous reviews by experts in the area, and every short paper received at least two anonymous reviews by experts in the area. Dual submissions were explicitly prohibited.

The members of the program committee and extra reviewers deserve special thanks for their professional reviews all received in the short time required for this ADCS conference. Reviewers not listed among the program committee include: Peter Bailey, Shlomo Geva, Alexander Krumpholz, Jose Lay, Ben Martin, Johvan Pehcevski, and Tom Rowlands.

We would also like to thank RMIT School of Computer Science and IT, and NICTA (Victoria) for their generous sponsorship of the event.

The symposium includes many formal presentations, but perhaps its greatest benefit lies in the opportunity it provides for document computing practitioners and researchers to get together and informally share ideas and enthusiasm.

Symposium Co-Chairs

Andrew Turpin	RMIT University
Mingfang Wu	RMIT University

Program co-chairs

Amanda Spink	Queensland University of Technology
Andrew Turpin	RMIT University

Program Committee

Peter Bruza	Queensland University of Technology
Bob Colomb	University of Technology Malaysia
Stijn Dekeyser	University of Southern Queensland
Peter Eklund	University of Wollongong
David Hawking	CSIRO, Canberra
Rob McArthur	CSIRO, Canberra
Alistair Moffat	The University of Melbourne
Gitesh K. Raikundalia	Victoria University
Falk Scholer	RMIT University
Saied Tahagohghi	RMIT University
Jamie Thom	RMIT University
Anne-Marie Vercoustre	INRIA, France
Anh Vo	The University of Melbourne
William Webber	The University of Melbourne
Ross Wilkinson	CSIRO, Canberra
Mingfang Wu	RMIT University
Justin Zobel	NICTA

ADCS Advisory Committee

Peter Bruza	Queensland University of Technology
Judy Kay	The University of Sydney
David Hawking	CSIRO, Canberra
Alistair Moffat	The University of Melbourne
Amanda Spink	Queensland University of Technology
Ross Wilkinson	CSIRO, Canberra
Justin Zobel	NICTA

Contents

Keynote I

Keynote II (Joint with ALTW2007)

Papers

Posters

Keynote Talk I

How to Evaluate Information Retrieval: Why is it Receiving Attention Now?

My talk is comprised of three parts. PART ONE: I will introduce a couple of novel information access services that we provide at FreshEye, the Japanese Web portal run by NewsWatch, Inc. NewsWatch was founded by Toshiba in 1996, and was bought by Yahoo! JAPAN in 2006. PART TWO: I will then discuss why information retrieval evaluation is receiving a lot of attention in the research community now, and mention some challenges, including the relevance data incompleteness issue, and the possibility of evaluating online, nontraditional information access services like the ones I have mentioned in PART ONE. PART THREE: I will describe the ongoing activities at NTCIR, which is an international information retrieval evaluation effort for Asian languages. The latest tasks cover complex question answering, cross-language information retrieval, opinion extraction and patent mining/translation. I will conclude the talk by urging you to submit a paper to EVIA 2008, the Second International Workshop on Evaluating Information Access, which will be held together with NTCIR-7 in Tokyo.

About Tetsuya Sakai

Tetsuya Sakai is the Director of the Natural Language Processing Laboratory at NewsWatch, Inc. which runs the Japanese Web portal FreshEye. In 1993, He received his Master's degree from Waseda University and joined Toshiba Corporate R&D Center. He received his Ph.D from Waseda University in 2000 for his work on information retrieval and filtering systems. From 2000 to 2001, he was a visiting researcher at the University of Cambridge Computer Laboratory. In February 2007, he left Toshiba to join NewsWatch, Inc. He has received several awards from the Information Processing Society of Japan (IPSJ) and the Institute of Electronics, Information and Communication Engineers (IEICE). He is currently the Asian Regional Representative of ACM SIGIR, and is on the steering committee of Asia Information Retrieval Symposium (AIRS) and the editorial board of the international journal "Information Retrieval".

Keynote Talk II

Measures of Measurements: Robust Evaluation of Search Systems

A good search system is one that helps a user to find useful documents. When building a new system, we hope, or hypothesise, that it will be more effective than existing alternatives. We apply a measure, which is often a drastic simplification, to establish whether the system is effective. Thus the ability of the system to help users and the measurement of this ability are only weakly connected, by assumptions that the researcher may not even be aware of. But how robust are these assumptions? If they are poor, is the research invalid? Such concerns apply not just to search, but to many other data-processing tasks. In this talk I introduce some of the recent developments in evaluation of search systems, and use these developments to examine some of the assumptions that underlie much of the research in this field.

About Justin Zobel

Professor Justin Zobel is leading the Computing for Life Sciences initiative within National ICT Australia's Victorian Laboratory. He received his PhD from the University of Melbourne and for many years was based in the School of CS&IT at RMIT University, where he led the Search Engine group. He is an Editor-in-Chief of the International Journal of Information Retrieval, an associate editor of ACM Transactions on Information Systems and of Information Processing & Management, and was until recently Treasurer of ACM SIGIR. In the research community, he is best known for his role in the development of algorithms for efficient text retrieval. He is the author of "Writing for Computer Science" and his interests include search, bioinformatics, fundamental data structures, and research methods.

Multimedia Web Searching on a Meta-Search Engine

Dian Tjondronegoro & Amanda Spink

Faculty of Information Technology
Queensland University of Technology
2 George St, Brisbane, QLD 4001

dian@qut.edu.au, ah.spink@qut.edu,au

Bernard J. Jansen

College of Information Science and Technology
The Pennsylvania State University
State College PA 16802 USA

jjansen@psu.edu

Abstract *This paper provides preliminary results from a major study of multimedia Web searching by Dogpile meta-search engine users, including queries and session characteristics, and changes or differences in image, video and audio searching. The results are compared with multimedia Web searching studies from 1997 to 2002. Image and sexual queries are dominant in multimedia Web searching. The paper provides important implications for the design of multimedia information retrieval systems.*

Keywords

1. Introduction

Tracking the trends in users' interactions with Web search engines has emerged an important area of research within information retrieval (IR). Trends in multimedia Web searching are playing an important role in new designs for multimedia Web retrieval systems. Next generation multimedia search engines need to support user's needs and searching behavior [3, 8]. Research shows that users still find it difficult to develop keywords and queries as most multimedia retrieval needs are often not describable using exact keywords or sample images/sounds [8]. Users' satisfaction on image search is often compromised, as they often cannot formulate better search queries [4].

Many studies have analyzed different aspects of Web query data logs, including multimedia Web search [6, 9]. A study of multimedia Web search behavior using the Excite dataset 1997-2001 [5] has found that multimedia queries are generally longer and mostly performed with multimedia interface buttons that allow users to select the particular type of media to search. Tjondronegoro, et al., (in press) found that: (1) few major Web search engines offer multimedia searching and (2) multimedia Web search functionality is generally limited. Despite the increasing level of interest in multimedia Web search, those few Web search engines offering multimedia Web search, provide limited multimedia search functionality. Keywords are still the only means of multimedia retrieval, while other methods such as "query by example" are offered by less than 1% of Web search engines examined. The study reported in this paper is part of an ongoing research project to investigating Web searching behavior [2, 7, 9, & 11]. Major studies of Web user behavior are significant for the development of more effective multimedia IR systems.

In this paper, we report findings from a study of trends in multimedia Web searching by Dogpile users in 2006, including queries and session characteristics, and changes or differences in image, video and audio searching. To assess the new trends and changes in Web multimedia searching, we compare our 2006 preliminary findings with multimedia Web searching trends from 1997 to 2002 [1, 5]. Our paper provides implications for the development of the next generation of multimedia Web search engines.

2. Research design

Our research goals were to examine (1) multimedia Web searching by Dogpile users in 2006, (2) queries and session characteristics, and changes or differences in image, video and audio searching, and (3) compare our findings previous studies of multimedia Web searching trends from 1997 to 2002 [1, 5].

2.1 Dogpile web query log fields

Dogpile (http://www.dogpile.com/) is a meta-search engine, which combines results from multiple search engines. Searches are based on the exact terms entered by user as a query. The Dogpile transaction log used in this paper consists of 1,228,330 queries that are combined from the multimedia tabbed interfaces, namely image, audio, and video. The data collection date was 15 May 2006 with 128,305 search sessions. Each query log record contains 6 fields:

- *Identification*: anonymous code assigned by Dogpile server to a user machine
- *IP:* user machine's Internet address which can be used to uniquely identify users
- *Cookie*: this number is assigned from the first time

a user is connected to the search engine until the user left a session, therefore is used as sessions identification.

- *Time of Day*: the time (in hours, minutes, and seconds) that a particular query is submitted
- *Query:* the user terms entered into the query box (e.g. “Web search”)
- *Organic/Sponsored:* the number of clicks on the organic and sponsored links (indicating that the link is followed up the user and may be deemed as relevant or interesting). Organic links are naturally returned by the search engine (based on the search algorithm), while sponsored links are paid by advertisers.

To obtain human (not agent) sessions, all sessions with more than 101 queries submitted (in a session) were deemed to be conducted by agents. This threshold is used, as it is almost 50 times greater than the reported mean search session for human Web searchers [5].

2.2 Quantitative analysis

Analyses were conducted at multiple levels, using the following metrics:

- *Terms level* - any string of characters bounded by some delimiter such as white space.
- Q*uery* level – a query is a string list of one or more entered terms.
  - ○ *Query length* is measured by counting the number of terms in the query.
  - ○ *Query complexity* examines the query syntax, including the advanced searching techniques such as Boolean and other query operators.
- S*ession* level – a session is the entire sequence of queries entered by a searcher with a given data sampling method.
  - ○ *Session length* is measured by the number of queries per searcher as each searcher is given a unique identifier within the log, namely the IP address.
- *Results pages viewed* level – a results page is the list of results returned by a search engine in response to a query, either organic or sponsored. The result page viewing patterns of Web searchers are analyzed through the number of results pages viewed.
- *Click-through level* – from the results page, a searcher may click on a URL to visit one or more results from the results listing, a method which is often referred as page view analysis. When a link is being followed through by users, we can assume that the result is relevant.

3. Results

After being filtered, we focused our study on 127,613 human Dogpile search sessions.

5.1 Search sessions

Table 1 shows a comparison of sessions and query length in multimedia searching. Image search dominate (i.e. 60% of multimedia sessions), 32% were audio sessions and 18% were video queries. Compared to the 1997 - 2001 Excite Web log trend study [1], this proportion is still consistent; averaged over 3 years, 50% of multimedia search is for image, 28% for video, and 22% for audio. However, video is becoming even less dominant, as it is still the most heavy - band media and probably due to the growing popularity of independent streaming video hosts, such as *YouTube,* which have their own internal search system.

The mean audio’s session duration is the highest at 30.3 minutes, while video is 20.7 minutes, and image is 20.3 minutes. The mean queries per session show that users have entered more queries on video search (2.9) followed by audio (2.4) and image (2.1) respectively. Compared to 1997-2001 on ‘mean session duration’ and ‘queries per session’ depicted in Table 1, the majority of users in 2001 and 2006 equally are not spending more than 30 minutes per session, while the average session duration has not changed much (i.e. 200s or 16% increase in image, 400s or 22% decrease in audio, and 150s or 13% increase in video).

We also examined the percentage of session duration ranging between less than 5 minutes and more than 5 hours. The findings are presented by Table 2 to show the actual percentages and the overall distribution.

The distribution is almost equivalent between image, audio and video, that is “less than 5 minutes’ session duration being the highest (i.e. greater than 58% of the total durations for each type of media), while each of the other session duration length categories being less than 10%.

5. 2 Terms per multimedia query

We studied the mean terms per query for all identified audio, video and image queries. A total of 1,849,410 terms were used in 1,129,041 multimedia queries, the average terms per query for audio search is slightly longer being 3.1, followed by audio and video at 2.3 terms per query equally. This is most likely because audio search queries usually contain terms from songs title, which are generally longer. Query length for multimedia searching generally ranges between 1 to 4 terms. Two terms are most commonly used. Table 3 shows the typical length of multimedia search query and the occurrences based on our Web log data.

Variables	Multimedia by Humans	Image	Audio	Video
Total sessions	127,813 (100%)	76,155 (60%)	41,335 (32%)	23,945 (18%)
Average Duration per Session (in seconds)	1561.6 (26 mins)	1217.1 (20.3 mins)	1820.6 (30.3 mins)	1242.3 (20.7 mins)
Maximum Duration Per Session (in seconds)	86235 (23.9 hours)	86235 (23.9 hours)	85441 (23.73 hours)	86218 (23.9 hours)
Standard Deviation for Duration of Sessions (in seconds)	5673.9 (94.5 mins)	4965.76 (82.8 mins)	6027.39 (100.5 mins)	5244.4 (87.4 mins)
Average Queries Per Session	2.4	2.1	2.4	2.9
Maximum Queries Per Session	86	65	73	86
Standard Deviation for Queries Per Session	2.75	2.03	2.8	3.9

Table 1. Comparison of audio, video and images search sessions.

Session Duration	Occurrences in Image Search (%)	Occurrences in Audio Search (%)	Occurrences in Video Search (%)
Less than 5 minutes	50,819 (66.7%)	24,154 (58.4%)	15,772 (65.9)
5 to 10 minutes	7,323 (9.6%)	3819 (9.2)	2373 (9.9)
10 to 15 minutes	4,171 (5.5%)	2425 (5.9)	1324 (5.5)
15 to 30 minutes	5,802 (7.6%)	3873 (9.4)	1921 (8.0)
30 to 60 minutes	3,413 (4.5%)	2917 (7.1)	1198 (5.0)
1 to 2 hour	1,790 (2.4%)	1525 (3.7)	553 (2.3)
2 to 3 hour	775 (1%)	707 (1.7)	232 (1)
3 to 4 hour	523 (0.7%)	509 (1.2)	146 (0.6)
4 to 5 hour	389 (0.5%)	452 (1.1)	116 (0.5)
More than 5 hours	1,150 (1.5%)	954 (2.3)	360 (1.5)
Total	76,155	41,335	23,945

Table 2. Session duration analysis in multimedia search.

Query Length	Occurrences in Image Search	Occurrences in Image Search (%)	Occurrences in Audio Search	Occurrences in Audio Search (%)	Occurrences in Video Search	Occurrences in video Search
1	31,748	23.9	13,663	15.2	9,575	22.8
2	49,094	36.9	22,601	25.2	15,200	36.1
3	28,424	21.4	18,621	20.7	8,869	21.1
4	13,744	10.3	14,685	16.3	4,713	11.2
5	6,014	4.5	9,531	10.6	2,164	5.1
6	2,390	1.8	5,552	6.2	956	2.3
7	918	0.7	2,857	3.2	397	0.9
8	386	0.3	1,250	1.4	65	0.2
9	180	0.1	613	0.7	58	0.1
>=10	135	0.1	489	0.5	55	0.1
Total	133,033		89,862		42,052	

Table 3: Query length per session statistics in multimedia search.

The distribution statistics of query length in video and image search is almost equivalent. Query length of greater than or equal to 6 is rarely used with percentage of less than 2%. In audio search, however, query length of 5 and 6 are more often used with percentages of 10.6% and 6.2% respectively.

Compared to 1997-2002 study, the query length distribution is almost the same for image, audio and video search. There are two notable differences: 1) in audio search 2002, query length of greater than or equal to 5 happened much less than 2006; 2) in image search 2002, the percentage of query length equal to 9 is 27% (which was deemed as an anomaly of the data collection).

5.3 Click-through analysis

Table 4 show the top 10 terms by which users found many relevant results as indicated by how many times they are entered by users (not necessarily unique) and the number of results viewed (i.e. clicked). All of top 10 terms in image queries are sexual. In audio queries, we found most of them are either song title or artist (or performer).

Image			Audio			Video		
Total Queries	562,380		Total Queries	370890		Total Queries	208115	
Query	Frequency (%)	Organic Links Followed	Query	Frequency (%)	Organic Links Followed	Query	Frequency (%)	Organic Links Followed
pussy	0.08	498	ridin dirty	0.07	427	pussy	0.63	1371
boobs	0.04	251	ridin	0.06	335	boobs	0.30	652
sex	0.04	237	shakira	0.06	318	hentai	0.23	514
hentai	0.03	196	eminem	0.06	343	sex	0.21	468
porn	0.03	181	50 cent	0.06	362	porn	0.17	370
tits	0.02	176	dani california	0.06	266	preteen	0.15	320
paris hilton	0.02	134	temperature	0.04	231	tits	0.15	344
milf	0.02	129	panic at the disco	0.04	191	paris hilton	0.14	310
ass	0.02	122	ms new booty	0.04	197	milf	0.14	315
penis	0.02	113	sean paul	0.04	199	ass	0.12	260

Table 4. Top 25 most occurring and followed up multimedia query terms.

Table 4 suggests that some queries co-occurred in multiple media searches. An analysis of overlapping queries (i.e. unique terms which are used in multiple multimedia types searches) shows that there are 12,730 overlapping queries that are used both in image and video searches. There are 5,742 queries co-occurring in image and audio searches, and 5,754 co-occurring in video and audio. Compared to the 2002 study of frequently occurring terms [1], this study demonstrates that the information need for multimedia Web searching remains consistent. In this new study, we have extended the statistics with the frequencies of followed-up links for each of the popular queries. The numbers have shown that the frequencies of followed up links are proportionally the same with the frequencies of the particular query is entered (i.e. video is much higher than image, and audio is slightly higher than image).

The number of search sessions can be sorted based on the most frequent into: image, audio, and video (highest to lowest). Whereas, the number of links clicked by users (which indicate the impact of the search results) can be sorted into: video, audio, image (highest to lowest). This phenomenon can be interpreted as an interesting search behavior in multimedia domain. It can show that users need to spend more time browsing, clicking and viewing the search results in image, audio, and video (respectively) to find the most relevant file, thereby reducing the number of sessions, which can be seen as new search process.

7. Conclusion

Most of image and video search is for sexual information, while music is the most popular genre for audio. Thus, it will be important for multimedia Web search engines to provide better and real-time support for automated content-based censorship (e.g. skin-based pornographic images identification) to protect children from accessing offensive materials. On the other side of the coin, it is highly likely that sex will remain to be the most popular multimedia genre being sought after by users if search engines would like to maximize the profits advertisements and sponsored links. Our future work aims to conduct a more thorough study using larger scale data and other data analysis techniques to investigate what type of other genres that attract users most (aside from sexual and music or movies).

References

[1] B. J. Jansen, A. Spink and J. Pedersen. An analysis of multimedia searching on AltaVista. In *Proc. SIGMM International Workshop on Multimedia Information Retrieval, Berkeley, CA, 2003.*

[2] B. J. Jansen, A. Spink and T. Saracevic. Real life, real users, and real needs: A study and analysis of user queries on the Web. *Information Processing and Management*, Volume 6, pages 207-227, 2000.

[3] M. L. Kherfi, D. Ziou and A. Bernardi, Image Retrieval from the World Wide Web: Issues, Techniques and Systems. *ACM Computing Surveys (CSUR)*, Volume 36, pages 35-67, 2004.

[4] S. McDonald and J. Tait. Search strategies in content-based image retrieval. In *26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), Amsterdam, Netherlands, 2003*

[5] S. Ozmutlu, A. Spink and H. Ozmutlu, Multimedia web searching trends: 1997- 2001. *Information Processing and Management*, Volume l Number 39, pages 611-621, 2003.

[6] C. Silverman, M. Henzinger, H. Marais and M. Morris. Analysis of a very large web search engine query log. *ACM SIGIR Forum*, Volume 33, 1999.

[7] A. Spink and B. J. Jansen. *Web Search: Public Searching of the Web.* Dordrecht: Springer.

[8] A. Spink. A user-centered approach to evaluating human interaction with web search engines: an exploratory study. *Information Processing & Management*, Volume 38, pages 401-426, 2002.

[9] A. Spink, B. J. Jansen, D. Wolfram and T. Saracevic. From e-sex to e-commerce: web search changes. *IEEE Computer*, Volume 35, pages 107-109, 2002.

[10] D. Tjondronegoro and A. Spink. Search engine Multimedia functionality. *Information Processing and Management* (In Press).

[11] D. Wolfram, A. Spink, B. J. Jansen and T. Saracevic. Vox populi: the public searching of the web. *Journal of the American Society for Information Sciences and Technology*, Volume 52, Number 12, pages 1073-1074, 2001.

Can Requests-for-Action and Commitments-to-Act be Reliably Identified in Email Messages?

Andrew Lampert †‡
†CSIRO ICT Centre
North Ryde NSW Australia
Andrew.Lampert@csiro.au

Cécile Paris
CSIRO ICT Centre
North Ryde NSW Australia
Cecile.Paris@csiro.au

Robert Dale
‡Centre for Language Technology
Macquarie University
NSW Australia
rdale@ics.mq.edu.au

Abstract *This paper reports on the results of an exploratory annotation task where three coders classified the presence and strength of Requests-for-Action (requests) and Commitments-to-Act (promises) in workplace email messages. The purpose of our annotation task was to explore levels of human agreement to establish whether this is a repeatable task that lends itself to automation. The results from our annotation task suggest that there is relatively high agreement about which sentences embody Requests-for-Action (*$\kappa = 0.78$*), but poorer agreement about Commitments-to-Act (*$\kappa = 0.54$*). Analysis of cases of coder disagreement highlighted several areas of systematic disagreement which we believe can be addressed through refining our annotation guidelines. Given this scope for improving agreement, we believe the results presented here are encouraging for our intention to perform larger-scale annotation work leading to automation of the detection and classification of Requests-for-Action and Commitments-to-Act in email communication.*

Keywords Email, document workflow, document management, Speech Acts, task management

1 Introduction

It is well documented that users routinely use email for managing requests and commitments in the workplace (e.g., [15, 7]). With the volume of email ever increasing, previous studies have highlighted that when facing the need to manage multiple ongoing tasks through email, users regularly feel overloaded [21, 3].

As a consequence of feelings of overload, many users struggle to give appropriate attention to tasks hidden in email that require action. The problem is severe enough that some people have even advocated giving up on managing the mountain of pending email tasks altogether by declaring ”Email Bankruptcy”[1] and wiping the slate clean. Of course, this serves only to restart the process, and all too soon important requests and promises are likely to be unintentionally overlooked or ignored in the noise of an overloaded inbox.

Our research aims to help email users by providing them with tools that can automatically detect Requests-for-Action and Commitments-to-Act within their incoming and outgoing email messages. As we explain in Section 4.1 and 4.2, we define a Request-for-Action to be a sentence that creates some form of obligation for an email recipient to do something (a request), and a Commitment-to-Act as some form of commitment from the email sender to perform some action (a promise). An example Request-for-Action from our data is *Please call when you have a chance.*; an example Commitment-to-Act is *I'll keep you posted on any changes.*

Requests-for-Action and Commitments-to-Act in email are particularly interesting because they occur in the context of textual conversations. Because they form part of a conversation, email messages differ from most other forms of written documents that have been more widely studied in the IR and NLP communities. One such difference is that email messages involve interaction between two or more participants, and as a result, the structure of email messages and conversations includes patterns borrowed from verbal conversation. We believe this structure can be usefully exploited to improve the efficiency and effectiveness of managing tasks in email. An example is the role of adjacency pairs of Speech Acts across email messages, which we believe can be used to help associate requests or promises with relevant responses from later email messages within the same conversation.

Proceedings of the 10th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

[1] See, for example,
http://www.wired.com/culture/lifestyle/news/2004/06/63733

There are also a variety of ways in which information about Requests-for-Action and Commitments-to-Act can be used to augment and improve existing tools for managing email, some of which we have presented previously [12]. We hope to couple information about Requests-for-Action and Commitments-to-Act with notions of conversation structure to provide tools that allow users to be more efficient and effective in identifying, prioritising, acting on and monitoring actionable content in email messages.

In the work we present in this paper, we focus on an experiment involving human coders manually identifying Requests-for-Action and Commitments-to-Act in workplace email messages. To automatically identify these actions is a non-trivial language processing task, because of the variety of surface forms that people can use to phrase their requests and promises. We want to investigate how closely humans agree in interpreting requests and promises in email. The problem of ambiguous surface forms is closely related to challenges encountered when automatically classifying Speech Acts associated with utterances. Similarly, difficulties that arise in recognising implicit, indirect Speech Acts must also be resolved when attempting to identify Requests-for-Action or Commitments-to-Act that are expressed implicitly or whose interpretation depends on other aspects of the email message context such as relationships between the sender and recipient, or the previous conversation history.

Our paper is structured as follows. First, in Section 2, we describe related work that has looked at email from an action-oriented perspective, and explain how our work both builds on and differs from these activities. In Section 3, we describe how we have extracted and processed data from the Enron Email Corpus for use in our pilot annotation task that is described in Section 4. We then present the results of our annotation experiment in Section 5 and discuss the levels of human agreement in identifying Requests-for-Action and Commitments-to-Act in Section 6 before making some concluding remarks and indicating our directions for future work in Section 7.

2 Related Work

In considering email from a conversational point of view, our work differs from the approach of many existing search engines and email systems which routinely treat email messages as simple bags-of-words, ignoring any conversational structure. The idea of identifying and exploiting patterns of communicative acts in textual conversation is, however, not new. Two decades ago, Flores and Winograd [22] proposed that workplace workflow can be seen as a process of creating and maintaining networks of conversations in which requests and promises lead to successful completion of work. Our research is attempting to build on these influential ideas to provide intelligent, automated assistance to email users. The work presented in this paper represents some early exploratory steps towards this goal.

While our approach differs from most existing email software, we are certainly not alone in looking at email from an action-oriented point of view. In particular, there is a growing body of research that has taken ideas from Speech Act Theory [1, 19] and applied them to analysing and enhancing email communication. This existing work differs as to whether the Speech Acts (or more properly, Speech-Act-inspired units) should be annotated at the message level (as in [5, 14]) or at the utterance/sentence level (as in [6]). Our thesis is that a single email message may contain multiple Requests-for-Action and Commitments-to-Act on a range of tasks and topics, and thus our work focuses on sentence-level classification. It is possible that another classification unit is also appropriate, such as clause-level annotation, but for simplicity the annotation task we report on here is performed at the sentence level.

The most similar work to our own is that of Corston-Oliver et. al. [6], whose SmartMail system attempted to automatically extract and reformulate action items from email messages for the purpose of adding them to a user's to-do list. While their task is similar in nature, our work differs in a number of aspects. First, Corston-Oliver et. al. only attempt to identify tasks that ”looked like an appropriate item to add to an on-going ’to do’ list”. In particular, they note that factual questions are not annotated as tasks because responding fulfills any associated obligation. In contrast, we annotate factual questions, and indeed any non-rhetorical questions, as Requests-for-Action, because some level of obligation to respond is imposed on the recipient, and thus the user may benefit from tools that can aid in identifying and tracking such requests.

A more significant distinction is that we are focused on identifying both Commitments-to-Act and Requests-for-Action, due to our broader ideas about tasks that a user may want to monitor. While Corston-Oliver et. al. do include a *promise* category in their annotation taxonomy, no mention is made of its use, and no definition is given for what they consider to be *promises*. Additionally, they restrict coders to applying a single tag to each sentence, meaning that a sentence cannot embody both a *task* and a *promise*. In the results of our annotation task, we see that this is a potentially artificial restriction.

Our motivation for considering Commitments-to-Act in addition to Requests-for-Action (which subsume the *tasks* that Corston-Oliver et. al. were identifying) is well explained by Murray's observations borne out of her ethnographic research into the use of electronic messaging at IBM in the early 1990s [16]. In that work, she notes that “They [managers] would like to be able to track outstanding promises they have made, promises made to them, requests they've made that have not been met and requests made of them that they have not fulfilled.” More recent studies of email usage by Bellotti et. al. in their work on the TaskMaster system [3] also identified similar problems with “keeping track of lots of concurrent actions: One’s own to-dos and to-dos one expects from others” using existing email clients. To allow users to keep track of actions in this way requires an ability to identify both Requests-for-Action (requests) and Commitments-to-Act (promises) in email messages.

Work by Kushmerick and Lau [11] is also similar to our work in terms of both their focus on ongoing activities that occur via email, and their aim to provide high-level overviews of ongoing activities to help users manage these tasks more effectively. The application of their ideas is, however, restricted to repeated, structured processes such as those that occur with e-commerce transactions, whereas our interest lies beyond these in the more informal exchange of Requests-for-Action and Commitments-to-Act that occurs in task-oriented workplace email conversations.

3 Preparation of Email for Annotation

To prepare for our pilot annotation task, we manually extracted 54 email messages containing 310 sentences, which we stored in MySQL database tables. Email messages were selected to represent a variety of syntactic styles of expressing possible Requests-for-Action and Commitments-to-Act. In particular, we attempted to select examples of sentences representing both explicit and implicit Requests-for-Action and Commitments-to-Act, and sentences with and without explicit task addressivity (i.e., sentences that do and do not address Requests-for-Action to a specific, named recipient).

3.1 Email Corpus

Our set of 54 messages for the pilot annotation task were selected from a random subset of the Enron Email corpus [9]. For this exploratory stage of annotation, selected messages were constrained to be of less than 12 sentences, to reduce the effort associated with coding longer messages.

We used the version of the Enron corpus released as a MySQL database dump[2] by Andrew Fiore and Jeff Heer at the University of California, Berkeley. This database version of the corpus has had a substantial amount of processing performed on the contents of the raw Enron corpus. This processing includes removing duplicate email messages and normalizing names of senders and recipients. The end result is a corpus of just over 250,000 email messages. Like all publicly released versions of the Enron Email corpus, no attachments are included with any of the email messages.

3.2 Processing Email Body Text

We pre-processed the text in the body of each email message to try to remove email signatures and all quoted or forwarded email content, leaving only text written by the author of the current email message. We will refer to such original content written by the current email sender as *author text*.

We used the Jangada software [4] to identify signature blocks and reply lines. Unfortunately, in using this software, we identified similar shortcomings to those identified recently by Estival et. al. [8] who used Jangada on their own email corpus. In particular, Jangada did not accurately identify forwarded or reply material in the email messages we used for our annotation task. We posit that at least one factor in the poor performance of Jangada is due to it being trained on data from Usenet newsgroups, which generally uses different syntactic markers for forwarded and quoted material. We believe this is a significant factor in the systematic errors that Jangada makes in failing to identify quoted reply and forwarded content represented in the style used by Microsoft Outlook. Unfortunately, Outlook is the most common email client used to compose messages in the Enron corpus.

Unlike Estival et. al., whose work we only became aware of after running our pilot annotation, we did not develop our own document parser for email messages. Given a lack of other available email processing tools, we used the Jangada software despite its shortcomings, and allowed coders to flag processing errors during the annotation process.

3.3 Sentence Splitting

Because our annotation is performed at the sentence level, once we had attempted to remove quoted reply content, forwarded content and email signatures, we segmented the body of each email message into sentences. For this purpose, we used the SentParBreak sentence and paragraph segmenter [18]. SentParBreak uses heuristic rules for identifying the boundaries of sentences and paragraphs. We have not yet attempted to refine these rules for email data, and instead used SentParBreak without modification. We applied SentParBreak to the bodies of all 250,000 email messages in our corpus and produced just over 3 million sentences of probable author text. Due to the Jangada processing errors, however, we know that some proportion of these sentences actually contain quoted reply or forwarded email content.

Although we haven’t formally evaluated the performance of the SentParBreak as a sentence segmenter, we allowed coders to flag sentence segmentation errors during the annotation process, in the same way as we did for Jangada processing errors.

4 Annotation Task

The purpose of our pilot annotation task was to explore the levels of human agreement in identifying Requests-for-Action and Commitments-to-Act within email messages. We gave three coders (the authors of this paper) a set of annotation guidelines, described in Section 4,

[2] Available at: http://bailando.sims.berkeley.edu/enron/enron.sql.gz

and asked them to independently classify Requests-for-Action and Commitments-to-Act within a corpus of 310 sentences from 54 email messages.

The annotation guidelines were written by the first author with comments received from the other authors. The set of guidelines was finalised before any annotation took place.

Our pilot annotation task was performed using a custom web-based tool, developed using Ruby-on-Rails. This annotation tool displays email messages from the Enron Email database using a look and feel that approximates the way email messages are displayed in Microsoft Outlook. Each email message was presented with the usual preceding header fields and values: *From, Date, To, Cc, Bcc, and Subject*. Below the header information, in the email content pane, the author text of the email message was presented as a sequence of sentences, one per line. Paragraph breaks in the original email were represented by a single uncodable blank line in the annotation tool.

For each sentence, coders were required to select annotation values from a number of aligned drop-down menus. Using these menus, coders performed three actions for each sentence:

1. First coders indicated whether the sentence expressed a Request-for-Action for the Specified Recipient, and, if so, whether the Request-for-Action was weak, medium or strong. (See below for an explanation of Specified Recipient.)

2. Next, coders indicated whether the sentence expressed a Commitment-to-Act from the sender, and if so, whether the Commitment-to-Act was weak, medium or strong.

3. Finally, coders could optionally flag any processing errors with the sentence. Flaggable processing errors included sentence segmentation problems, and the inclusion of quoted or forwarded email material (non author text).

Coders were instructed to annotate each sentence in the context of the original email message, rather than in isolation. At the top of each message, one recipient to whom the email message was originally sent – the Specified Recipient – was noted, and coders were instructed to approach the annotation task from the point of view of that person. For the purposes of our pilot annotation task, the first non-sender recipient of the message was chosen as the Specified Recipient. The ’non-sender’ constraint was introduced to counter cases where the first recipient was actually the sender (presumably copying their own email to themself for action or recall purposes). Where no explicit recipients were identifiable, as in the case of an email message whose recipients are all Bcc’d, coders were instructed to annotate the email message from the point of view of any recipient. In general, Requests-for-Action and Commitments-to-Act in such messages tend to be addressed to all recipients, meaning that annotating from a the point of view of a generic recipient is acceptable. Coders were instructed not to mark any Requests-for-Action directed explicitly to recipients other than the Specified Recipient as Requests-for-Action.

To explain or comment on any aspect of the annotation task coders were able to make notes using a comments field for each email message. This comments field was also used in combination with the *Other* category of processing errors to highlight processing or display problems other than segmentation and author text related issues. Coders were instructed to use the comments field to explain any annotation decisions which they felt were conditional or context-sensitive. For example, if a coder’s decision depended on potentially ambiguous interpretation of the email message or its context, they were instructed to explain the basis for their annotation.

Finally, we also noted to coders that it was possible for a single sentence to contain both a Request-for-Action and a Commitment-to-Act: e.g., *Please send the document today, so I can get comments back to you by Monday*. The annotation tool made it possible for any sentence to be annotated as both a Request-for-Action and a Commitment-to-Act.

4.1 Requests-for-Action

A Request-for-Action places some form of obligation on the recipient to respond or act. An example Request-for-Action is: *Do you have an outage calendar for 2002?*.

A simple test for identifying a Request-for-Action is: Is this sentence asking me to do something? Examples of actions can include, but are not restricted to:

- answering a question, in email or otherwise;
- forwarding the message to a new recipient; or
- performing some action in the real world, such as preparing a document or gathering some data.

Instructions to coders were to annotate any sentence that carried an expectation that the Specified Recipient of the email message should respond or take some action as a Request-for-Action.

As we explained in Section 4, coders were also asked to indicate the strength of each Request-for-Action they identified. The different strengths were explained as follows:

- **Strong:** Action or response from the Specified Recipient is considered important and/or mandatory.
- **Medium:** The sender expects a response or action from the Specified Recipient.

- **Weak:** Action or response from the Specified Recipient is optional or conditional; the sender would find it reasonable if the Specified Recipient took no action.
- **None:** No Request-for-Action is expressed.

4.2 Commitments-to-Act

A Commitment-to-Act represents a promise from the author that action will be taken. An example Commitment-to-Act is: *Enron has committed to processing and paying any expenses incurred prior to transaction closing*.

A simple test for identifying a Commitment-to-Act is: Is this sentence promising to do something?

Commitments-to-Act occur both when an action is to be taken by the sender, or when the sender promises action on behalf of another person. The reason for including such indirect Commitments-to-Act is based on the intuition that such delegated promises might occur frequently because of the hierarchical nature of many workplaces, and are likely to be an important part of workplace conversations for action. An example of an indirect Commitment-to-Act is: *Peter will call to let you know the final arrangements*.

Coders were instructed to annotate any sentence that carried an expectation that the sender of the email message will take responsibility for some action being taken as a Commitment-to-Act.

As for Requests-for-Action, coders were also asked to indicate the strength of each Commitment-to-Act that they identified. The different strengths were explained as follows:

- **Strong:** Action from the Sender is considered important and/or mandatory.
- **Medium:** The Specified Recipient expects a response or action from the Sender.
- **Weak:** Action or response from the Sender is optional or conditional; the Specified Recipient would find it reasonable if no action was taken.
- **None:** No Commitment-to-Act is expressed.

4.3 Audience Types

For each email message in our pilot annotation corpus, we also manually classified the nature of the recipient audience. Each email was classified with one of the following message types:

- **Single Recipient:** Addressed to a single recipient. We also consider email messages that are addressed specifically to a single recipient but Cc'd or Bcc'd to another recipient to belong to this group.
- **Closed Group:** Addressed explicitly to a specified group of recipients. Each recipient must be identifiable from the email headers.
- **Broadcast:** Addressed to an unspecified group of recipients, such as a group alias or mailing list.

We used information about the audience type of email messages in analysing intercoder agreements for our annotation task, as we describe below in Section 5.

5 Results and Discussion

The results of our pilot annotation are shown in Table 1 and Table 2. Note that all κ values referred to in this section, apart from the pairwise κ values in Tables 1 and 2, refer to agreement between all three coders calculated using the generalization of Cohen's Kappa to more than two coders, as specified by Krippendorff [10].

The results in Table 1 show pairwise and three-way interannotator agreement between coders for our pilot annotation task. Separate *binary* and *strength* κ values are given for each measurement.

Binary agreement refers to interannotator agreement about which sentences contain a Request-for-Action, ignoring any indication of strength. To calculate these κ scores, we collapsed all three strengths of annotated Requests-for-Action into a single Request-for-Action class. Thus, disagreement about the strength of a Request-for-Action is ignored in binary κ scores.

Strength agreement, which is always lower than binary agreement, refers to interannotator agreement for the more fine-grained strength categories of Requests-for-Action. Thus, it represents agreement between coders over which sentences contain a Strong, Medium, Weak or No Request-for-Action. Disagreement about the strength of an identified Request-for-Action is considered a complete disagreement for the strength κ scores.

The results in Table 2 similarly show both pairwise and three-way interannotator agreement between coders for classifying Commitments-to-Act. The same separate measures of binary and strength agreement are used.

Finally, both Table 1 and Table 2 also show separate κ scores for subsets of the annotation corpus, grouped according to the email audience type (see Section 4.3).

From these results, we can draw several tentative conclusions about human agreement for identifying Requests-for-Action and Commitments-to-Act in email:

1. There is good agreement ($\kappa = 0.78$) about which sentences embody a Request-for-Action.
2. There is some tentative agreement ($\kappa = 0.60$) about the strength of Requests-for-Action.
3. There is poorer agreement about which sentences embody a Commitment-to-Act ($\kappa = 0.54$) and poor agreement about the strength of those commitments ($\kappa = 0.37$).

	Coder A & B		*Coder A & C*		*Coder B & C*		*3-Way κ*	
Message Types	Binary	Strength	Binary	Strength	Binary	Strength	Binary	Strength
All	0.83	0.57	0.79	0.66	0.74	0.56	0.78	0.60
Single Recipient	0.85	0.54	0.79	0.66	0.80	0.55	**0.81**	0.58
Closed Group	0.79	0.55	0.79	0.58	0.66	0.61	0.75	0.58
Broadcast	0.82	0.66	0.77	0.70	0.70	0.53	0.76	0.63

Table 1: Pairwise and 3-way Kappa agreements for classifying Requests-for-Action

	Coder A & B		*Coder A & C*		*Coder B & C*		*3-Way κ*	
Message Types	Binary	Strength	Binary	Strength	Binary	Strength	Binary	Strength
All	0.45	0.30	0.62	0.44	0.55	0.37	0.54	0.37
Single Recipient	0.51	0.28	0.76	0.54	0.52	0.41	**0.60**	0.41
Closed Group	0.14	0.06	0.51	0.32	0.41	0.19	0.35	0.19
Broadcast	0.52	0.41	0.52	0.40	0.66	0.41	0.57	0.41

Table 2: Pairwise and 3-way Kappa agreements for classifying Commitments-to-Act

4. The level of agreement appears to vary depending upon the audience type of the email message.

A particularly interesting aspect of our results is the variation in interannotator agreements across the different audience types that we identified in Section 4.3. As can be seen both in Tables 1 and 2, agreement between our three coders about the presence and strength of both Requests-for-Action and Commitments-to-Act is highest for Single Recipient email messages, and lowest for Closed Group email messages. In the case of Commitments-to-Act, this difference is particularly marked. Analysing the cases of disagreement has not yet revealed a recurring reason for these differences. Some coders did make observations that the strength of Requests-for-Action may be determined, to some extent, by the probability that it would apply to the Specified Recipient when an email message has multiple recipients. It's unclear how to objectively judge such a probability, but perhaps observations such as this shed some light on the reduced agreement for Closed Group email messages.

In addition to the annotations made regarding Requests-for-Action and Commitments-to-Act, we allowed users to flag segmentation errors, as noted in Section 3.3. Although we don't consider this method a rigorous way to evaluate the SentParBreak sentence splitter, it is interesting to note that the results from this error flagging suggest that segmentation error for our pilot annotation data is at least 10%. This is a much greater error rate than the 0.997352 precision and 0.995093 recall results that were apparently achieved using SentParBreak over the Genia corpus [17]. Such a difference in performance serves to highlight the need for standard NLP tools like sentence segmenters to be re-trained, re-tuned or otherwise tailored when working with email data, due to differences in the nature of textual content.

6 Error Analysis

In analysing cases of disagreement for classifying Requests-for-Action and Commitments-to-Act, it is clear that a significant proportion of problems can be solved by improving the annotation guidelines. Below we discuss several highly-represented classes of disagreement between coders.

6.1 Conditional Offers

One systematic source of disagreement between coders in identifying Commitments-to-Act was how to classify sentences that embody an *offer*. An example from one of our email messages is: *If you wish, I could provide you questions in advance to maximize your time*. Some coders classified sentences such as these as containing only a Request-for-Action for the recipient, while others judged that it contained both a Request-for-Action for the recipient and a Commitment-to-Act for the speaker, who implicitly promises to make good on their offer if the recipient accepts or acts on it.

This difference of opinion is actually not surprising when we look into the Speech Act literature. Offers were originally assigned to either be directive acts (which in our terminology are Requests-for-Action) or commissive acts (Commitments-to-Act) [19, 1]. Subsequent studies, however, have differed greatly over whether to categorise *offers* as commissive acts (e.g., [13]), directive/requestive acts (e.g. [20]), or some hybrid category in-between these (e.g. [2]).

Although the issue is still under discussion for our future annotation tasks, we are currently leaning towards specifying that *offers* should not be annotated as Commitments-to-Act, since no obligation to act is enacted until the recipient accepts the offer. At that point, we would classify the acceptance as a Request-for-Action from the original recipient. Doing so relies on the conversational nature of email communication and being able to identify a response to an offer, perhaps using adjacency pairing cues. Another possibility is to introduce the idea of a conditional Commitment-to-Act, which would qualify Commitments-to-Act in a manner orthogonal to the strength (i.e., we could have strong, medium or weak conditional Commitments-to-Act). As an aside, we could similarly have conditional Requests-for-Action. Regardless of how we decide to proceed, our annotation guidelines will be updated to provide specific instructions to coders specifying how *offers* should be annotated. This will improve our κ agreements considerably, since almost 40% of our disagreements over the presence of Commitments-to-Act are for sentences containing *offers*.

6.2 Implicit Requests

Another common source of disagreement occurs for implicit requests. In our pilot annotation corpus, these occur frequently as statements about meetings. As an example, one email from our corpus contains the following sentence: *Ken wants to have a meeting this afternoon in regard to California from a PR standpoint.* Our coders disagreed as to whether this statement represents a request to attend a meeting, and thus a Request-for-Action. In the original email this sentence is followed by an explicitly directive sentence: *Please let me know if you will be able to attend*. This second sentence is clearly a Request-for-Action, but it creates some confusion over the status of the first sentence. In the absence of the second directive sentence, the first sentence should, we would argue, be interpreted as a Request-for-Action, in essence acting as an indirect Speech Act. When followed by the second sentence, however, this interpretation as an indirect Speech Act is no longer necessary, as the Request-for-Action has moved to the explicit directive sentence. For our pilot annotation task, we directed coders to annotate each sentence in the context of the entire email message, which would therefore indicate that the first sentence should not be annotated as a Request-for-Action. Clearly, however, our annotation guidelines need to provide more specific guidance to coders for how to interpret such situations.

6.3 Relationship Context

In our pilot annotation task, we did not include any information that indicated the organisational hierarchy or relationships between senders and recipients. This lack of specified relationship context for each email message sometime lead to ambiguity in interpreting Requests-for-Action and Commitments-to-Act.

Some coders noted that differences in the relationship between the sender and the Specified Recipient (specifically, differences in the “importance” of the sender) would affect the strength they associated with Requests-for-Action or Commitments-to-Act.

One example sentence on which coders commented is: *You might like to organize the paper from a broad overview of the electrical market in the west (including basic descriptions, timelines, fundamentals etc.) down to a specific description of what you did at Enron.* Two coders commented that their interpretation of the strength of any Request-for-Action would be different depending on the organisational relationship between sender and receiver. If this represented, for example, a manager assigning a task to a direct report, then the sentence would be interpreted to have a different strength than if it were a Request-for-Action from a peer.

7 Conclusions and Future Work

We regard our current results as encouraging for automation of this classification task.

Based on analysis of the results from our pilot annotation task, we are working to refine our annotation guidelines and the annotation task in a number of ways. In addition to the issues discussed in Section 6, it is clear from our interannotator agreement results that we need to improve the way we define and distinguish strength for both Requests-for-Action and Commitments-to-Act. Our current plans in this regard are to revise our finer-grained classification of Requests-for-Action and Commitments-to-Act to capture conditionality and explicitness rather than require our coders to directly annotate the strength of requests and promises.

Once we have refined our annotation guidelines, we plan to run a larger annotation task using a wider pool of annotators. We hope to use our corpus of manually annotated data to bootstrap and refine automated classifiers and to iteratively use these classifiers to apply active learning techniques to select maximally informative email data for future stages of manual annotation.

We also plan to resolve some limitations of our existing email pre-processing steps. In particular, rather than removing non-author text such as quoted and forwarded material from email messages that we present to coders, we plan to include it as content that can be viewed but not annotated. This will be necessary as we expand our work to begin looking at aspects of conversational structure, so that coders can correctly identify features such as adjacency pairs that require knowledge of the email conversation history.

With reference to the interesting variations in interannotator agreement across different Audience Types, it is difficult to say anything conclusive, given the relatively small amount of data annotated in our pilot study. This is something that we plan to explore further in future annotation work.

Overall, given the scope for refining our annotation guidelines together with the current levels of human agreement, we believe that humans can reliably identify and classify Requests-for-Action and Commitments-to-Act in email messages, and thus that our task is repeatable.

Acknowledgements We thank Jeroen Geertzen for the use of his multi-annotator Kappa calculator[3] in calculating three-way interannotator agreement values for our annotation data.

References

[1] John L Austin. *How to do things with words*. Harvard University Press., 1962.

[2] Kent Bach and Robert M Harnish. *Linguistic Communication and Speech Acts*. The MIT Press, 1979.

[3] Victoria Bellotti, Nicolas Ducheneaut, Mark Howard and Ian Smith. Taking email to task: The design and evaluation of a task management centred email tool. In *Computer Human Interaction Conference*, CHI, Ft Lauderdale, Florida, USA, April 5-10 2003.

[4] Vitor R Carvalho and William W Cohen. Learning to extract signature reply lines from email. In *Proceedings of First Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, July 30-31 2004.

[5] William W Cohen, Vitor R Carvalho and Tom M Mitchell. Learning to classify email into ”speech acts”. In Dekang Lin and Dekai Wu (editors), *Conference on Empirical Methods in Natural Language Processing*, pages 309–316, Barcelona, Spain, 2004. Association for Computational Linguistics.

[6] Simon H Corston-Oliver, Eric Ringger, Michael Gamon and Richard Campbell. Task-focused summarization of email. In *ACL-04 Workshop: Text Summarization Branches Out*, July 2004.

[7] Nicolas Ducheneaut and Victoria Bellotti. E-mail as habitat: an exploration of embedded personal information management. *Interactions*, Volume 8, Number 5, pages 30–38, September/October 2001.

[8] Dominique Estival, Tanja Gaustad, Son Bao Pham, Will Radford and Ben Hutchinson. Author profiling for english emails. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 263–272, Melbourne, Australia, July 2007.

[9] Bryan Klimt and Yiming Yang. Introducing the enron corpus. In *First Conference on Email and Anti-Spam (CEAS)*, 2004.

[10] Klaus Krippendorff. *Content Analysis: An Introduction to its Methodology*. Sage Publications, Beverley Hills, CA, USA, 1980.

[11] Nicholas Kushmerick and Tessa Lau. Automated email activity management: An unsupervised learning approach. In *Intelligent User Interfaces Conference (IUI)*, San Diego, USA, 2005.

[12] Andrew Lampert. Managing obligations and commitments in email. Presentation at NGS07 - the Second HCSNet Workshop of the Next Generation Search Priority Area, Sydney, Australia, 19-20 July 2007.

[13] Geoffrey N Leech. *Principles of Pragmatics*. Longman Publishing Group, 1983.

[14] Anton Leuski. Context features in email archives. In *Proceedings of IRiX workshop at SIGIR'05*, 2005.

[15] Wendy E Mackay. More than just a communication system: Diversity in the use of electronic mail. In *ACM conference on Computer-supported cooperative work*, pages 344–353, Portland, Oregon, USA, 1988. MIT, ACM Press.

[16] Denise E Murray. *Conversation for Action: The Computer Terminal As Medium of Communication*, Volume 196 pages. John Benjamins Publishing Co, 1991.

[17] Scott Piao. Sentparbreaker web page, 2007. `http://text0.mib.man.ac.uk:8080/scottpiao/sent_detector`, Accessed: 5/10/2007.

[18] Scott S L Piao, Andrew Wilson and Tony McEnery. A multilingual corpus toolkit. In *Proceedings of AAACL*, 2002.

[19] John R Searle. *Speech Acts : An Essay in the Philosophy of Language*. Cambridge University Press, 1969.

[20] Amy B M Tsui. *English Conversation*. Oxford University Press, 1994.

[21] Steve Whittaker and Candace Sidner. Email overload: exploring personal information management of email. In *ACM Computer Human Interaction conference*, pages 276–283. ACM Press, 1996.

[22] Terry Winograd and Fernando Flores. *Understanding Computers and Cognition*. Ablex Publishing Corporation, Norwood, New Jersey, USA, 1st edition, 1986. ISBN: 0-89391-050-3.

[3] http://www.cosmion.net/jeroen/software/kappa/

A Bottom-up Term Extraction Approach for Web-based Translation in Chinese-English IR Systems

Chengye Lu, Yue Xu and Shlomo Geva
School of Software Engineering and Data Communications
Queensland University of Technology
Brisbane, QLD 4001, Australia
(c.lu,yue.xu,s.geva)@qut.edu.au

ABSTRACT

The extraction of Multiword Lexical Units (MLUs) in lexica is important to language related methods such as Natural Language Processing (NLP) and machine translation. As one word in one language may be translated into an MLU in another language, the extraction of MLUs plays an important role in Cross-Language Information Retrieval (CLIR), especially in finding the translation for words that are not in a dictionary.

Web mining has been used for translating the query terms that are missing from dictionaries. MLU extraction is one of the key parts in search engine based translation. The MLU extraction result will finally affect the transition quality.

Most statistical approaches to MLU extraction rely on large statistical information from huge corpora. In the case of search engine based translation, those approaches do not perform well because the size of corpus returned from a search engine is usually small. In this paper, we present a new string measurement and new Chinese MLU extraction process that works well on small corpora.

Keywords

Cross-language Information retrieval, CLIR, query translation, web mining, OOV problem, term extraction

1. INTRODUCTION

As more and more documents written in various languages become available on the Internet, increasingly users wish to explore documents that were written in their native language rather than English. Cross-language information retrieval (CLIR) systems enable users to retrieve documents written in more than one language through a single query. Obviously translation is needed in the CLIR process. The common approach is to translate the query into the document language using a dictionary. Dictionary based translation has been adopted in cross-language information retrieval because bilingual dictionaries are widely available, dictionary based approaches are easy to implement, and the efficiency of word translation with a dictionary is high. However, because of the vocabulary limitation of dictionaries, very often the translations of some words in a query can't be found in a dictionary. This problem is called the Out of Vocabulary (OOV) problem.

The OOV problem usually happens when translating Multiword Lexical Units (MLUs) such as proper names, phrases or newly created words. As the length of input queries is usually short, query expansion does not provide enough information to recover the missing words. Furthermore, very often the OOV terms are key terms in a query. In particular, the OOV terms such as proper names or newly created technical terms carry the most important information in a query. For example, a query "SARS, CHINA" may be entered by a user in order to find information about SARS in China. However SARS is a newly created term and may not be included in a dictionary which was published only a few years ago. If the word SARS is left out of the translated query or translated incorrectly, it is most likely that the user will practically be unable to find any relevant documents at all.

Web mining has been used for OOV term translation [4; 5; 7; 9; 13]. It is based on the observation that there exist web pages which contain more than one language. Investigation has found that, when a new English term such as a new technical term or a proper name is introduced into another language (target language), the translation of this term and the original English term very often appear together in documents written in the target language in an attempt to avoid misunderstanding. Mining this kind of web pages can help discover the translation of the new terms. Some earlier research already addressed the problem of how those kinds of documents can be extracted by using web search engines such as Google and Yahoo. Popular search engines allow us to search English terms for pages in a certain language, e.g., Chinese or Japanese. The result returned by a web search engine is usually a long ordered list of document titles and summaries to help users locate information. Mining the result lists is a way to find translations to the unknown query terms. Some studies[3; 13] have shown that such approaches are rather effective for proper name translation. To distinguish such approaches from other web mining based translation approaches, we call those approaches "search engine based translation approaches". Search engine based approaches usually consist of three steps:

1. Document retrieval: use a web search engine to find the documents in target language that contain the OOV term in original language. For example, when finding the translation of an English term in Chinese, the English term will be put in the search engine and ask the search engine return Chinese result only. Collect the text (i.e. the summaries) in the result pages returned from the web search engine.
2. Term extraction: extract the meaningful terms in the summaries where the OOV term appears. Record the terms and their frequency in the summaries. As a term in one language could be translated to a phrase or even a sentence, the major difficulty in term extraction is how to extract correct MLUs from summaries.
3. Translation selection: select the appropriate translation from the extracted words. As the previous step may produce a long list of terms, translation selection has to find the correct translation from the terms.

Step 2 and step 3 are the core steps of search engine based translation. In this paper, we present our contribution to term extraction and translation selection. Specifically, we introduce a statistics based approach to extraction of terms to improve the precision of the translations.

2. Previous Work

In this section, we briefly review some statistical based approaches on term extraction. Detailed analysis of these approaches will be given in the evaluation section.

Term extraction is mainly the task of finding MLUs in the corpus. The concept of MLU is important for applications that exploit language properties, such as Natural Language Processing (NLP), information retrieval and machine translation. An MLU is a group of words that always occur together to express a specific meaning. The minimal size of a MLU should be 2. For example, compound nouns like *Disney Land*, compound verbs like *take into account*, adverbial locutions like *as soon as possible*, idioms like *cutting edge*. In most cases, it is necessary to extract MLUs, rather than words, from a corpus because the meaning of an MLU is not always the compositional of each individual word in the MLU. For example, you cannot interprete the MLU 'cutting edge' by combining the meaning of 'cutting' and the meaning of 'edge'.

Finding MLUs from the summaries returned by a search engine is important in search engine based translation because a word in one language may be translated into a phrase or even a sentence. If only words are extracted from the summaries, the later process may not be able to find the correct translation because the translation might be a phrase rather than a word.

For Chinese text, a word consisting of several characters is not explicitly delimited since Chinese text contains sequences of Chinese characters without spaces between them. Chinese word segmentation is the process of marking word boundaries. The Chinese word segmentation is actually similar to the extraction of MLUs in English documents since the MLU extraction in English documents also needs to mark the lexicon boundaries between MLUs. Therefore, term extraction in Chinese documents can be considered as Chinese word segmentation. Many existing systems use lexical based or dictionary based segmenters to determine word boundaries in Chinese text. However, in the case of search engine based translation, as an OOV term is an unknown term to the system, these kinds of segmenters usually cannot correctly identify the OOV terms in the sentence. Incorrect segmentation may break a term into two or more words. Therefore, the translation of an OOV term cannot be found in a later process. Some researchers suggested approaches that are based on co-occurrence statistics model for Chinese word segmentation to avoid this problem [4; 5; 8; 9; 13].

One of the most popular statistics based extraction approaches is to use mutual information [6; 12]. Mutual information is defined as:

$$MI(x,y)=\log_2 \frac{p(x,y)}{p(x)p(y)}=\log_2 \frac{Nf(x,y)}{f(x)f(y)} \qquad (1)$$

The mutual information measurement quantifies the distance between the joint distribution of terms *X* and *Y* and the product of their marginal distributions [1]. When using mutual information in Chinese segmentation, *x, y* are two Chinese characters; *f(x), f(y), f(x,y)* are the frequencies that *x* appears, *y* appears, and *x* and *y* appear together, respectively; N is the size of the corpus. A string *XY* will be judged as a term if the *MI* value is greater than a predefined threshold.

Chien [6] suggests a variation of the mutual information measurement called significance estimation to extract Chinese keywords from corpora. The significance estimation of a Chinese string is defined as:

$$SE(c)=\frac{f(c)}{f(a)+f(b)-f(c)} \qquad (2)$$

Where *c* is a Chinese string with *n* characters; *a* and *b* are two longest composed substrings of *c* with length *n-1*; *f* is the function to calculate the frequency of a string. Two thresholds are predefined: *THF* and *THSE*. This approach identifies a Chinese string to be a MLU by the following steps. For the whole string c, if f(*c)>THF*, *c* is considered a Chinese term. For the two (*n*-1)-substrings *a* and *b* of *c*, if *SE(c)>=THSE*, both *a* and *b* are not a Chinese term. If *SE(c)<THSE,* and *f(a)>>f(b)* or *f(b)>>f(a)* , *a* or *b* is a Chinese term, respectively. Then for each *a* and *b*, the method is recursively applied to determine whether their substrings are terms.

However, all mutual information based approaches have the problem of tuning the thresholds for generic use. Silva and Lopes suggest an approach called Local Maxima to extract MLU from corpora to avoid using any predefined threshold [12]. The equation used in Local Maxima is known as SCP defined as:

$$SCP(s)=\frac{f(s)^2}{\frac{1}{n-1}\sum_{i=1}^{n-1} f(w_1 ... w_i)\, f(w_{i+1} ... w_n)} \qquad (3)$$

S is an n-gram string, *w*$_{1,...,}$*w*$_i$ is the substring of *S*. A string is judged as an MLU if the SCP value is greater or equal than the SCP value of all the substrings of *S* and also greater or equal than the SCP value of its antecedent and successor. The antecedent of S is an (n-1)-gram substring of *S*. The successor of S is a string that S is its antecedent.

Although Local Maxima should be a language independent approach, Cheng et al.[5] found that it does not work well in Chinese word extraction. They introduced context dependency (CD) used together with the Local Maxima. The new approach is called SCPCD. The rank for a string is using the function:

$$SCPCD(s)=\frac{LC(s)RC(s)}{\frac{1}{n-1}\sum_{i=1}^{n-1} freq(w_1 ... w_i)\, freq(w_{i+1} ... w_n)} \qquad (4)$$

S is the input string, *w1..wi* is the substring of *S*, *LC()* and *RC()* are functions to calculate the number of unique left(right) adjacent characters of *S*. A string is judged as a Chinese term if the SCPCD value is greater or equal than the SCPCD value of all the substrings of *S*.

3. PROPOSED APPROACH

The term extraction approaches listed above have been used on large corpus. However, in our experiments, the performance of those approaches is not always satisfactory in search engine based

OOV term translation approaches. As described in introduction, web search engine result pages are used for search engine based OOV term translation. In most cases, only a few hundreds of top results from the result pages are used for translation extraction. As a result, the corpus size for search engine based approaches is quite small. In a small collection, the frequencies of strings very often are too low to be used in the approaches reviewed in Section 2. Moreover, the search engine results are usually part of a sentence, which makes the traditional Chinese word segmentation hard to be applied in this situation. That is why many researchers [4; 5; 7; 9; 13] try to apply statistical based approaches on search engine base translation for term extraction.

In this section, we describe a term extraction approach specifically designed for the search engine based translation extraction, which uses term frequency change as an indicator to determine term boundaries and also uses the similarity comparison between individual character frequencies instead of terms to reduce the impact of low term frequency in small collections. Together with the term extraction approach, we also describe a bottom-up term extraction approach that can help to increase the extraction quality.

3.1 Frequency Change Measurement

The approaches mentioned in Section 2 use a top-down approach that starts with examining the whole sentence and then examining substrings of the sentence to extract MLUs until the substring becomes empty. We propose to use a bottom-up approach that starts with examining the first character and then examines super strings. Our approach is based on the following observations for small document collections:

Observation 1: In a small collection of Chinese text such as a collection of search engine result pages, the frequencies of the characters in a MLU are similar. This is because in a small collection of text, there are a small number of MLUs, the characters appearing in one MLU may not appear in other MLUs. On the other hand, some times MLUs with similar meanings will share similar characters and those characters are unlikely to be used in other unrelated MLUs. For example, 戰機 (Fighter Aircraft) and 戰鬥機 have the same meaning in Chinese. They share similar Chinese characters. Therefore although the term's frequency is low, the individual characters of the term might still have relatively high and also similar frequencies. The high frequency can help in term extraction.

Observation 2: When a correct Chinese term is extended with an additional character, the frequency of the new term very often drops significantly.

According to Observation 1, the frequencies of a term and each character in the term should be similar. We propose to use the root mean square error (RMS error) given in Equation (5) to measure the similarity between the character frequencies.

$$s = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad (5)$$

For a given Chinese character sequence, x_i is the frequency of each character in the sequence, $\bar{x}$ is the average frequency of all the characters in the sequence. Although the frequency of a string is low in small corpora, the frequencies of Chinese characters still have relatively high values. According to Observation 1, if a given sequence is an MLU, the characters in the sequence should have a similar frequency, in other words, s should be small. If the frequencies of all the characters in a Chinese sequence are equal, then $s = 0$. Because s represents the average frequency error of individual characters in the sequence, according to observation 1, in an MLU, the longer substring of that MLU will have smaller average frequency error.

According to our observation 1, an MLU can be identified by Equation 5. However, as Equation 5 only measures the frequency similarity between individual characters, any character combinations may be identified as MLU if their frequencies are similar; even when they are not occurring together. To avoid this problem, we introduce sequence frequency *f(S)* into the formula. Therefore, if the characters are not occurring together, they won't be considered as a sequence and therefore *f(S)* = 0. Thus any character combinations can be identified if they appear together as a sequence in the corpra.

Finally, we combine the sequence frequency and the RMSE measurement together. We designed the following equation to measure the possibility of *S* being a term:

$$R(S) = \frac{f(S)}{s+1} = \frac{f(S)}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} + 1} \qquad (6)$$

Where, *S* is a Chinese sequence; *f(S)* is the frequency of *s* in the corpus. We use $s+1$ as the denominator instead of using s to avoid 0 denominators.

Let *S* be a Chinese sequence with *n* characters; S= $a_1a_2\ldots a_n$. And *S'* is a substring of *S* with length *n*-1; S' = $a_1a_2\ldots a_{n-1}$.

According to observation 1, we should have:

- If *S* is an MLU, we will have *f(S)* ≈ *f(S')*.
- If S is an MLU, the longer is S, the smaller the average mean square error is.

Therefore, in the case *S'* is a substring of *S* with length *n*-1, we would have$\sigma<\sigma'$. As a result we will have *R(S)>R(S')*. In another case where *S'* is a substring of *S* and S' is an MLU while *S* is not. In other words, *S* has an additional character to an MLU. In this case, we will have *f(S) <f(S')* and the frequency of the additional character makes the RMSE value larger, so$\sigma>\sigma'$.Therefore, *R(S) <R(S')*.

In summary, for a string S and its substring S', the one with higher R value would most likely be an MLU. Table 1 gives the R value of each possible term in a Chinese sentence chosen from a small collection of summaries returned from a search engine: "隱形戰機/是/一種/靈活度/極差/的/戰機" ("/" indicates the lexicon boundary given by a human).

Table 1 Chinese strings and *R(S)*

String *S*	*R(S)*
隱形	26.00
隱形戰	0.94

戰機	2.89
戰機是	0.08
一種	0.44
一種靈	0.21
靈活	2.00
靈活度	2.00
靈活度極	1.07
極差	0.8
極差的	0.07
戰機	2.89

This example clearly shows that if a Chinese MLU has an additional character, its R value will be significantly smaller than the R value of the MLU. For example, R(一種)=0.44>R(一種/靈)=0.21, R(靈活)=R(靈活度)=2.00>R(靈活度極)=1.07. It is reasonable that if we segment the Chinese sentence at the position that the string's R value drops greatly. For the example sentence, it would be segmented as: "隱形/戰機/是/一種/靈活度/極差/的/戰機" by the proposed method. The only difference between the human segmented sentence and the automatic segmented sentence is that "隱形戰機" (Stealth Fighter) is segmented into two words "隱形" (Stealth) and "戰機" (Fighter) by the proposed method. However, this is still an acceptable segmentation because those two words are meaningful.

3.2 A Bottom-up Term Extraction Strategy

As mentioned in Section 3.1, the top-down strategy is firstly to check whether the whole sentence is an MLU, then reduce the sentence size by 1 and recursively check sub sequences. It is reported that over 90% of meaningful Chinese terms consist of less than 4 characters [9], and on average, the number of characters in a sentence is much larger than 4. Obviously, a whole sentence is unlikely to be an MLU. Therefore, checking the whole sentence for an MLU is unnecessary. In this section, we describe a bottom-up strategy that extracts terms starting from the first character in the sentence. The basic idea is to determine the boundary of a term in a sentence by examining the frequency change (i.e., the change of the *R* value defined in Equation (6)) when the size of the term is increasing. If the *R* value of a term with size *n*+1 drops compared with its largest sub term with size *n*, the sub term with size *n* is extracted as an MLU. For example, in Table 1, there is a big drop between the *R* value of the third term "靈活度" (2.00) and its super term "靈活度極" (1.07). Therefore, "靈活度" is considered as an MLU.

The following algorithm describes the bottom-up term extraction strategy:

Algorithm BUTE(*s*)

Input: *s*=$a_1a_2\ldots a_n$ is a Chinese sentence with n Chinese characters

Output: M, a set of MLUs

```
1. Check each character in s, if it is a stop character such as 是 (is, are), 的(of) ，了…, remove it from s. After removing all stop characters, s becomes $a_1a_2\ldots a_{m,}$ m≤n.
2. Let b=2, e=2, and M= $f$
3. Let $t_1= a_ba_2\ldots a_{e,}$  $t_{2=}$ $a_ba_2\ldots a_{(e+1).}$
   If R(t1) >R(t2), then M=M ∪ (t1), b=e+1.
4. e=e+1, if e+1>m, return M, otherwise go to step 3.
```

The algorithm makes the sub sequence uncheckable once it is identified as an MLU (i.e., b=e+1 in step 3 ensures that the next valid checkable sequence doesn't contain t_1 which was just extracted as an MLU). However, when using the bottom-up strategy described above, some longer term might be missed since the longer term contains several shorter terms. As showed in our example, "隱形戰機" (Stealth Fighter) consists of two terms "隱形" and "戰機". When using bottom-up strategy, "隱形戰機" would not be extracted because the composite term has been segmented into two terms. To avoid this problem, we set up a fixed number w which equals specifies the maximum number of characters to be examined before reducing the size of the checkable sequence. The modified algorithm is given below:

Algorithm BUTE-M(*s*)

Input: *s*=$a_1a_2\ldots a_n$ is a Chinese sentence with n Chinese characters

Output: M, a set of MLUs

```
1. Check each character in s, if it is a stop character such as 是，了，的…, remove it from s. After removing all stop characters, s becomes $a_1a_2\ldots a_{m,}$ m≤n.
2. Let b=2, e=2, First-term = true, and M= $f$
3. Let $t_1= a_ba_2\ldots a_{e,}$  $t_{2=}$ $a_ba_2\ldots a_{(e+1).}$
   If R(t1) >R(t2),
     then M:=M ∪ {t1}
         If First-term = true
           then first-position:= e and First-term:= false
   If e-b+1 ≥ $w$
      then e:=first-position, b:=e+1, First-term:=true.
4. e=e+1, if e+1>m, return M, otherwise go to step 3
```

In algorithm **BUTE-M**, the variable first-position gives the ending position of the first identified MLU. Only when w characters have been examined, the first identified MLU will be removed from the next valid checkable sequence, otherwise the current sequence is still being checked for a possible MLU even it contains an extracted MLU. Therefore, not only the term "隱形" and "戰機" will be extracted but also the longer term "隱形戰機" (Stealth Fighter) will be extracted.

3.3 Translation selection

Translation selection is relatively simple compared with term extraction. The translation of a word in a source language is typically determined according to the ranking of the extracted terms. Each of the terms is assigned a rank, usually calculated based on term frequency and term length[13]. The term with the highest rank in the extracted term list is selected as the translation of the English term.

As we have described in other papers [9; 10], the traditional translation selection approaches select the translation on the basis of word frequency and word length [4; 13]. We have suggested an approach to finding the most appropriate translation from the extracted word list regardless of term frequency. In our scheme even a low frequency word will have a chance to be selected. Our experiments in that paper show that in some cases, the most appropriate translation is the low frequency word. In this paper, we only give a brief description of our translation selection technique. The reader is referred to [9] for a more complete discussion.

The idea of our approach is to use the translation disambiguation technology to select the translation from the extracted term list. As the extracted terms are from the result set returned by the web search engine, it is reasonable to assume that those terms are relevant to the English query term that was submitted to the web search engine. If we assume all those terms are translations of the English term, we can apply the translation disambiguation technique to select the most appropriate term as the translation of the English term. We also introduced a filtering technique in our approach to minimize the length of the extracted term list.

In our approach, the correct translation will be selected using a simple translation disambiguation technique that is based on co-occurrence statistic. We use the total correlation which is one of several generalizations of the mutual information to calculate the relationship between the query words.

Our modified total correlation equation is defined as

$$C(x_1x_2x_3..x_n)=\log_2\frac{f(x_1x_2x_3..x_n)+1}{(f(x_1)+1)(f(x_2)+1)...(f(x_n)+1)} \quad (7)$$

Here, x_i are query words, $f(x_i)$ is the frequency that the query word x_i appears in the corpus, $f(x_1x_2x_3...x_n)$ is the frequency that all query words appears in the corpus. For each word frequency, we add 1 because we want to avoid 0 appearing in the equation when a word's frequency is 0.

The frequency information required by equation 7 can be easily collected from local corpus.

4. EVALUATION

We have conducted experiments to evaluate our proposed query translation approach. Using the algorithm described in Section 3.2, we firstly extract Chinese terms from the summaries returned by a search engine with an English term as the query. These Chinese terms are considered the candidate translations of the query English term. We then use the method described in Section 3.3 to select the most appropriate translation. The web search engine that we used in the experiments is Google, and the top 300 summaries returned were used for later processing. The English queries entered into Google will be enclosed by double quotation to ensure Google only returns result with exact phrases. Also specified result pages written in Chinese.

4.1 Test set

140 English queries from the NTCIR6 CLIR task were used. Query terms were first translated using Yahoo's online dictionary. (http://tw.dictionary.yahoo.com/). The remaining OOV terms which could not be translated were used to evaluate the performance of our web based query translation approach described in Section 3.2 and 3.3 by comparing with other approaches. There are 69 OOV terms altogether. The correct translation of each OOV term is given by NTCIR.

4.2 System setup

In our experiments, we evaluated the effectiveness of term extraction approaches in OOV translation. All the approaches described in section 2.1 were used in the experiment. The abbreviations are: MI for Mutual information, SE for the approach introduced by Chien, SCP for the Local Maxima introduced by Silva and Lopes, and SCPCD for the approach introduced by Jenq-Haur Wang et al.. Our extraction approach with BUTE-M extraction strategy is abbreviated as SQUT

The OOV term is translated via the following steps:

1. From the result pages downloaded from Google, use the 5 different term extraction approaches to produce 5 Chinese term lists.
2. For each term list, remove a term if it can be translated to English by Yahoo's online dictionary. This leaves only OOV terms.
3. From each remaining term list which contains only OOV terms, select the top 20 terms as translation candidates. Select the final translation from the candidate list using our translation selection approach described in 3.3.

Finally we have 5 sets of OOV translations, as shown in the Appendix.

4.3 Results and discussion

For the 69 OOV terms, by using the 5 different term extraction approaches, we obtained the translation results shown in Table 2. Details of the translation are showed in appendix.

As we were using the same corpus and the same translation selection approach, the difference in translation accuracy is the result of different term extraction approaches. Thus we can claim that the approach with the higher translation accuracy has higher extraction accuracy.

As we can see from table 2 below, SQUT has the highest translation accuracy. SCP and SCPCD provided similar performance. The approaches based on mutual information provided lowest performance.

Table 2. OOV translation accuracy

	Correct	Accuracy (%)
MI	30	43.5
SE	41	59.4
SCP	53	76.8
SCPCD	52	75.4
SQUT	59	85.5

4.3.1 Mutual information based approaches

In the experiment, MI based approach cannot determine the Chinese term boundaries well. The term lists produced by MI based approaches contain a large number of partial Chinese terms. It is quite often that partial Chinese terms were chosen as the translation of OOV terms. Some partial Chinese terms selected by our system are listed in table 3

Table 3 Some Extracted terms by MI

OOV Terms	Extracted terms	Correct terms
Embryonic Stem Cell	胚胎幹細	胚胎幹細胞
consumption tax	費稅	消費稅
Promoting Academic Excellence	卓越發	卓越發展計畫

Mutual information based term extraction approaches, such as MI and SE, are affected by many factors. These approaches rely on the predefined thresholds to determine the lexicon boundaries. Those thresholds can only be adjusted experimentally. Therefore, they can be optimized in static corpus. However, in OOV term translation, the corpus is dynamic. The result pages returned from search engine will be different for different term query entered. It is impossible to optimized thresholds for generic use. As a result, the output quality is not guaranteed.

In addition, mutual information based approaches seem unsuitable in Chinese term extraction. As there are no word boundaries between Chinese words, the calculation of MI values in Chinese are based on Chinese characters but not words as it does in English. The average high school graduate in the U.S. has a vocabulary of 27,600 words [11], while the cardinality of the commonly used Chinese character set is under 3000 [2]. Since Chinese characters have much higher frequencies than English words, one Chinese character will be used in many MLUs while an English word will have less chance to be used in Multiple MLUs. As a result, an English MLU will have much higher MI value than a Chinese MLU. The subtle differences in MI values between Chinese MLUs and non-MLUs make the thresholds hard to tune for generic use.

Some filtering techniques are used in SE to minimize the affect of thresholds. In our experiment, there is 17.2% improvement in translation accuracy. Obviously the improvement comes from the higher quality of extracted terms. However, the limitation of thresholds is not avoidable.

4.3.2 Local Maxima based approaches

Without using thresholds, local maxima based approaches have much better flexibility than MI based approaches, achieving higher translation accuracy in our experiment. In comparison, the SCP approach tries to extract longer MLUs while the SCPCD approach tries to extract shorter ones. The translation of "Autumn Struggle", "Wang Dan", "Masako" and "Renault" are all 2-character Chinese terms. SCPCD can extract the translation with no problem while SCP always has trouble with them. As over 90% of the Chinese terms are short terms, this is a problem for SCP in Chinese term extraction. In the mean time, SCPCD has trouble in extracting long terms. Overall, the two local maxima based approaches have similar performance. However, since in our experiment, most of the translations of OOV terms are long terms, SCP's performance is a little better than that of SCPCD.

Local maxima based approaches use string frequencies in the calculation of $\frac{1}{n-1}\sum_{i=1}^{n-1} f(w_1...w_i)\, f(w_{i+1}...w_n)$. In a small corpus, the frequency of a string becomes very low which makes the calculation of string frequencies less meaningful. Local Maxima based approaches are not effective in a small corpus. In comparison, our approach calculates the difference between character frequencies. In a small corpus, characters still have a relatively high value. As a result, our approach performs better than Local Maxima based approaches in small corpora. For example, local maxima based approaches were unable to extract the translation of "Nissan Motor Company" because the corpus is too small - Google only returns 73 results for the query "Nissan Motor Company".

4.3.3 SQUT Approach

Most of the translations can be extracted by the SQUT algorithm. As our approach monitors the change in R to determine a string to be an MLU instead of using the absolute value of R, it does not have the difficulty of using predefined thresholds. In addition, the use of single character frequencies in RMSE calculations makes our approach usable in small corpora. Therefore, we have much higher translation accuracy than MI based approaches and also about 10% improvement over Local Maxima based approaches.

However, the SQUT algorithm has difficulty in extracting the translation of "Wang Dan". In analyzing the result summaries, we found that the Chinese character "王"("Wang") is not only a very frequent character in the summaries but also used in other terms such as "霸王"(the Conqueror), "帝王"(regal); "國王"(king); "女王" (queen) and "王朝" (dynasty). Those terms also appear frequently in the result summaries. In our approach, where we are using the count of individual characters, the very high frequency of "王" breaks observation 2. Thus the translation of "Wang Dan" cannot be extracted. However, in most cases, our observations are true in small corpora as demonstrated by the high translation accuracy of our approach in query expansion from Chinese/English web search summaries.

5. Conclusion and Future Work

In this paper, we proposed a bottom-up term extraction approach to be used in small corpora. The method introduces a new measurement of a Chinese string based on frequency and RMSE, together with a Chinese MLU extraction process based on the change of the new string measurement that does not rely on any predefined thresholds. The method considers a Chinese string as a term based on the change of R's value when the size of the string increases rather than the absolute value of R. Our experiments show that this approach is effective in web mining for translation extraction of unknown query terms.

Although the proposed approach shows impressive accuracy for OOV term translation, there are still some works to be conducted in the future. Our experiments were conducted using a small scale test set which only has 69 OOV terms from NTCIR6 CLIR task queries. It might be necessary to test our approach under larger scale test set such as a test set that has over 1000 OOV terms. Although there are 140, only 50 of them have document relevance results given by NTCIR. There are only 11 OOV terms in those 50 queries. As a result, the number of OOV terms is not enough to distinguish the different approaches. While many researchers [4; 5; 7; 9; 13] had showed that better query translation quality should improve the CLIR performance, it might be necessary to test actual benefits of high translation accuracy in CLIR in the future when we have appropriate testing data.

6. REFERENCES

[1] Mutual information, Wikipedia.

[2] The List of common use Chinese Characters, Ministry of Education of the People's Republic of China.

[3] A. Chen, H. Jiang, and F. Gey, Combining multiple sources for short query translation in Chinese-English cross-language information retrieval, Proceedings of the fifth international workshop on on Information retrieval with Asian languages, ACM Press, Hong Kong, China, 2000.

[4] A. Chen, and F. Gey, Experiments on Cross-language and Patent retrieval at NTCIR3 Worksho, Proceedings of the 3rd NTCIR Workshop, Japan, 2003.

[5] P.-J. Cheng, J.-W. Teng, R.-C. Chen, J.-H. Wang, W.-H. Lu, and L.-F. Chien, Translating unknown queries with web corpora for cross-language information retrieval, Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Sheffield, United Kingdom, 2004.

[6] L.-F. Chien, PAT-tree-based keyword extraction for Chinese information retrieval, Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval ACM Press, Philadelphia, Pennsylvania, United States 1997.

[7] J. Gao, J.-Y. Nie, E. Xun, J. Zhang, M. Zhou, and C. Huang, Improving query translation for cross-language information retrieval using statistical models, Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, New Orleans, Louisiana, United States, 2001.

[8] M.-G. Jang, S.H. Myaeng, and S.Y. Park, Using mutual information to resolve query translation ambiguities and query term weighting, Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, Association for Computational Linguistics, College Park, Maryland, 1999.

[9] C. Lu, Y. Xu, and S. Geva, Translation disambiguation in web-based translation extraction for English-Chinese CLIR, Proceeding of The 22nd Annual ACM Symposium on Applied Computing, 2007.

[10] C. Lu, Y. Xu, and S. Geva, Improving translation accuracy in web-based translation extraction, Proceedings of the 6th NTCIR Workshop Meeting on Evaluation of Information Access Technologies:Information Retrieval, Question Answering and Cross-Lingual Information Access, Japan, 2007.

[11] M. Salovesh, How many words in an "average" person's vocabulary?, http://unauthorised.org/anthropology/anthro-l/august-1996/0436.html, 1996.

[12] J.F.d. Silva, and G.P. Lopes, A Local Maxima Method and a Fair Dispersion Normalization for Extracting Multiword Units, International Conference on Mathematics of Language, 1999.

[13] Y. Zhang, and P. Vines, Using the web for automated translation extraction in cross-language information retrieval, Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Sheffield, United Kingdom, 2004.

7. Appendix Sample of translated terms

OOV term	SQUT	SCP	SCPCD	SE	MI
Autumn Struggle:	秋鬥大遊	從秋鬥	秋鬥	秋鬥	秋鬥
Jonnie Walker:	約翰走路	約翰走路	黑次元	高雄演唱	高雄演唱
Charity Golf Tournament:	慈善高爾夫球賽	慈善高爾夫球賽		慈善高	慈善高
Embryonic Stem Cell:	胚胎幹細胞	胚胎幹細胞	胚胎幹細胞		
Florence Griffith Joyner:	花蝴蝶	葛瑞菲絲	葛瑞菲絲	花蝴蝶	花蝴蝶
FloJo:	佛羅倫薩格里菲斯	花蝴蝶	花蝴蝶	花蝴蝶	花蝴蝶
Michael Jordan:	麥可喬丹	麥可喬丹	喬丹	喬丹	喬丹
Hu Jin tao:	胡錦濤	胡錦濤	胡錦濤	胡錦濤	胡錦濤
Wang Dan:		天安門	王丹	王丹	王丹
Tiananmen	天安門廣場	天安門	天安門	天安門	天安門
Akira Kurosawa:	黑澤明	黑澤明	黑澤明	黑澤明	黑澤明
Keizo Obuchi:	小淵惠三	小淵惠三	小淵惠三	小淵惠三	小淵惠三
Environmental Hormone:	環境荷爾蒙	環境荷爾蒙	環境荷爾蒙	環境荷爾蒙	
Acquired Immune Deficiency Syndrome:	後天免疫缺乏症候群	愛滋病	愛滋病	愛滋病	愛滋
Social Problem:	社會問題	社會問題	社會問題		

Kia Motors:	起亞汽車	起亞汽車	起亞汽車	起亞	起亞
Self Defense Force:	自衛隊	自衛隊	自衛隊	自衛隊	自衛隊
Animal Cloning Technique:	動物克隆技術	動物克隆技術			
Political Crisis:	政治危機	政治危機	政治危機		
Public Officer:	公職人員	公職人員	公職人員	公職人員	
Research Trend:	研究趨勢	研究趨勢	研究趨勢	研究趨勢	
Foreign Worker:	外籍勞工	外籍勞工	外籍勞工	外籍勞工	
World Cup:	世界盃	世界盃	世界盃	世界盃	世界盃
Apple Computer:	蘋果公司	蘋果電腦	蘋果電腦	蘋果電腦	蘋果電腦
Weapon of Mass Destruction:	大規模毀滅性武器	大規模毀滅性武器	性武器		
Energy Consumption:	能源消費	能源消費	能源消費		
International Space Station:	國際太空站	國際太空站	國際太空站		
President Habibie:	哈比比總統	哈比比總統	哈比比總統	哈比比	
Underground Nuclear Test:	地下核試驗	地下核試驗	地下核試		
F117:	戰鬥機	隱形戰機	隱形戰	隱形戰	隱形戰
Stealth Fighter:	隱形戰機	隱形戰機	形戰鬥機	隱形戰	隱形戰
Masako:	雅子	太子妃	雅子	雅子	雅子
Copyright Protection:	版權保護	版權保護	版權保護	版權保護	版權保護
Daepodong:	大浦洞	大浦洞	大浦洞	大浦洞	大浦洞
Contactless SMART Card:	智慧卡	非接觸式智慧卡	非接觸式智慧卡	非接觸式	非接觸式
Han Dynasty:	漢朝	大漢風	漢朝	漢朝	漢朝
Promoting Academic Excellence:	學術追求卓越發展計畫	卓越計畫	卓越發展計畫	卓越發展計畫	卓越發
China Airlines:	中華航空	中華航空	中華航空	中華航空	長榮
El Nino	聖嬰	聖嬰現象	聖嬰現象	聖嬰	聖嬰
Mount Ali:	阿里山	阿里山	阿里山	阿里山	阿里山
Kazuhiro Sasaki:	佐佐木主浩	佐佐木主浩	佐佐木	佐佐木	佐佐木
Seattle Mariners:	西雅圖水手	西雅圖水手	西雅圖水手		
Takeshi Kitano:	北野武	北野武	北野武	北野武	北野武
Nissan Motor Company:	日產汽車公司	汽車公司	汽車公司	處經濟	處經濟
Renault:	雷諾	休旅車	雷諾	雷諾	雷諾
war crime:	戰爭罪	戰爭罪	戰爭罪	戰爭罪	
Kim Dae Jung:	金大中	金大中	金大中	金大中	金大中
Medecins Sans Frontieres:	無國界醫生	無國界醫生	無國界醫生		
volcanic eruptions:	火山爆發				
Clinton:	克林顿	克林顿	克林顿		
Science Camp:	科學營	科學營	科學營	科學營	
Kim Il Sung:	金日成	金日成	金日成	金日成	金日成
anticancer drug:	抗癌藥物				
consumption tax:	消費稅	消費稅	消費稅	消費稅	費稅
Uruguay Round:	烏拉圭回合	烏拉圭回合	烏拉圭回合		
Economic Collaboration:	經濟整合	經濟整合	經濟整合	經濟整合	經濟整合
Kim Jong Il:	金正日	金正日	金正日	金正日	金正日

IR Evaluation Using Multiple Assessors per Topic

Andrew Trotman
Department of Computer Science
University of Otago
Dunedin, New Zealand
andrew@cs.otago.ac.nz

Dylan Jenkinson
Department of Computer Science
University of Otago
Dunedin, New Zealand
djenkins@cs.otago.ac.nz

Abstract: *Information retrieval test sets consist of three parts: documents, topics, and assessments. Assessments are time-consuming to generate. Even using pooling it took about 7 hours per topic to assess for INEX 2006.*

Traditionally the assessment of a single topic is performed by a single human. Herein we examine the consequences of using multiple assessors per topic.

A set of 15 topics were used. The mean topic pool contained 98 documents. Between 3 and 5 separate assessors (per topic) assessed all documents in a pool. One assessor was designated baseline. All were then used to generate 10,000 synthetic multi-assessor assessment sets.

The baseline relative rank order of all runs submitted to the INEX 2006 relevant-in-context task was compared to those of the synthetics. The mean Spearman's rank correlation coefficient was 0.986 and all coefficients were above 0.95 – the correlation is very strong. Non matching rank-orders are seen when the mean average precision difference between runs is less than 0.05. In the top 10 runs no significantly different runs were ranked in a different order in more than 5% of the synthetics. Using multiple assessors per topic is very unlikely to affect the outcome of an evaluation forum.

Keywords: Information Retrieval.

1. Introduction

Information retrieval evaluation forms such as TREC [12] and INEX [2] are large international collaborations aiming to improve the performance of search engines. Each year they release a document collection and a set of information needs called topics. Participants index the documents, run the queries, and submit the results (called runs) back to the forum.

The accuracy of each run is then measured by comparing the results lists against the known correct answers for each topic. These known correct answers are called assessments (or judgments).

Producing the assessments is expensive and time-consuming. At TREC the assessors are retired information specialists who are paid to perform the task. At INEX the assessors are the participants themselves who must perform the task in addition to their ordinary duties. In both cases a human must decide which documents (and additionally at INEX which parts of those documents) are relevant to which topics and which are not. The test collections, however, are so large that it is not feasible to judge every document against every topic.

TREC introduced the pooling method of reducing the assessment load. In TREC pooling the (typically 100) top ranking documents from each run are pooled. The pool is then de-duplicated and the pool documents are judged. Those documents that do not appear in the pool are not judged and are assumed to be non-relevant. INEX uses a similar pooling method, but targets a pool size of (currently) 500 documents, but the number of results that end up in the pool varies from topic to topic.

INEX imposes an additional rule – wherever possible a topic should be judged by the topic author. Even if this is not possible it should be judged by a single judge. In 2006 these assessors took an average of 7 hours per topic, and some were asked to assess as many as 3 topics. Assessment is a burden and consequently several others have investigated methods of reducing the load. Ogilvie & Lalmas [3], for example, show that for INEX using binary relevance is as effective as graded relevance. Piwowarski *et al*. [6] show that when identifying relevant parts of a document a yellow highlighting method is more effective than identifying the relevance of each individual XML element. Anecdotes suggest that changes over the last 5 years have reduced the assessment load from about a week per topic to about a day per topic.

We are interested in further reducing the assessment load. To do this we propose relaxing the requirement that the assessment is by a single assessor (and consequently the topic author). If we are able to do so then an individual topic might be assessed by a group of 7 graduate students over a one hour period. Assessment might even be used as a teaching exercise.

A double-judging experiment was run as part of INEX 2006 in which 15 topics were judged by two assessors each without the knowledge of the other. Trotman *et al*. [11] ran a further assessment experiment on these topics resulting in between 3 and 5 assessors for a mean of about 100 documents per topic. They examined the effect of shallow pooling (100 vs.

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007.

500 documents per pool) and showed that the shallow pools were effective in giving an indication of the performance of a run but not for distinguishing the performance of the top 10 runs.

Their result does suggest that using an assessor other than the original topic author is likely to be effective. They measure the performance of each run against the official assessments and correlate it with an alternative set they produced. The correlation is very strong (Spearman's of 0.97) except for the top 10 runs which do not correlate. The reason for the non-correlation is likely to be the pool size difference of 500 documents in the official assessments and 100 in theirs.

Trotman *et al*. [11] mix the official, the alternative, and their assessment sets and produce a set of between 3 and 5 separate assessments sets for a mean of 98 documents for the 15 topics. We use that set in our experiments.

We first generate the mean average precision for each run submitted to the INEX 2006 relevant-in-context task against the official runs in the Trotman *et al*. subset. The relative rank order of the runs is generated from the mean average precisions.

From the whole subset we generate 10,000 synthetic assessment sets by randomly choosing one of the assessors decisions for each document of each topic. These sets are representative of different random ways of distributing documents to different assessors for assessment – and the combination of results that might be seen.

The mean average precision is then computed for each of the 10,000 sets and the relative rank order of runs is compared to the official rank order using Spearman's correlation.

We note a mean correlation score of 0.986, that is, the relative rank orders very strongly correlate. No substantial difference would have been seen if multiple assessors had been used per topic.

Further we examine the mean average precision difference between two runs at which we would expect to see a difference between the official assessments and those of multiple assessors.

For each possible pair of runs (A and B) we compute the difference in mean average precision. From the 10,000 synthetic runs we note the number of times the relative rank order of A and B is different to that seen with the official runs. We bucket this into mean average precision differences of 0.01 and fit a line to it. We note the point at which the probability of a switch is less than 0.05 (which we consider statistically significant) is between 0.02 and 0.03.

Since the best MAP score seen is 0.36, this is representative of an approximately 7% increase in mean average precision. We believe that anything smaller is not interesting anyway; Voorhees & Buckley [13] suggest a MAP increase of 0.05 (15% at 0.3) is needed for there to be a meaningful difference between runs.

2. The INEX Evaluation Forum

The details of how a test set is constructed differs between evaluation forum. INEX currently uses a dump of the Wikipedia subsequently converted into XML [1]. The collection is distributed to participants who are then asked to identify typical information needs they have of the collection. These needs are expressed in written paragraph form along with the reason the information is needed. This expression of the information need is referred to as the narrative.

Users do not typically type their information need into a search engine as paragraphs of text. The average length of a web query is between 2 and 3 words [8] and not a long explanation of information needed. These web queries are derivatives of the user's information need, and there are many possible such queries a user might give. Specifically INEX identifies two kinds of queries: Content Only (CO) queries, and Content and Structure (CAS) queries. The former is the typical keyword only query seen by a web search engine; the latter additionally includes structural constraints identifying which document structures (sections, paragraphs, etc.) are likely to contain relevant instances of the keywords and the preferred granularity of the search result. Along with the narrative participants are asked to submit a CO query and, if applicable, a CAS query too. Upon receiving a set of topics a participant extracts the queries and runs them through their search engines producing a set of results per topic.

INEX is investigating the evaluation of search engines that identify results smaller in size than a whole document. In 2006 four tasks were identified: *thorough*, *focused*, *relevant-in-context*, and *best-in-context*.

In the thorough task the search engine must identify relevant XML-elements from relevant documents and rank them relative to each other. The focused task, by contrast, must do the same but without identifying overlapping results.

For the relevant-in-context task the search engine must first identify and rank relevant documents, then it must identify all relevant XML-elements within those documents (unordered, non-overlapping). The best-in-context task, by contrast, must identify the "best" point, in each relevant document, from which a user should start reading in order to satisfy their information need.

Each of these tasks was examined by Trotman *et al*. [10] in an effort to identify a user base. They gave a use-case for each but concluded that the relevant-in-context task was the most viable. This task is, consequently, of interest to us.

An obvious use of the relevant-in-context paradigm is searching collections of long documents such as a library of books. Relevant books would first be identified then relevant parts (XML-elements or passages) within those books would be highlighted. In 2007 INEX initiated a book searching track using

42,049 books totalling 210GB of text – the results have not yet been published.

For the relevant-in-context task a result for a single topic is a set of 1,500 elements grouped together by document. A run is a set of results, one for each of a set of topics. These runs are then submitted to INEX for evaluation.

A perfect run would identify only relevant documents and within those documents only relevant passages. Pehcevski *et al*. [5] state that "Users want to see as much relevant information as possible with as little irrelevant information as possible". Any metric used to score a search engine is giving a measure of this, typically weighted so that the beginning of a result list is over-weighted while the end is under-weighted. But in order to evaluate a run the correct answers to the query must be known (so that a run can be compared to it) and producing the answers is expensive and time consuming.

INEX topics are assessed by their original author (where possible), that is, the participants. Assessment is in addition to their ordinary duties which can include teaching and research. In 2006 each group was asked to assess three topics. If all three were assessed by the same individual the total load was about 5 days. Methods of reducing the assessment burden are, consequently, of interest.

Several methods of reducing the assessment load are already being used at INEX. Topics are not assessed to exhaustion; instead a pooling strategy called top-n is used.

In top-n pooling the first result from each run is added to a pool. The pool is then de-duplicated and if it contains more than *n* (in 2006 *n*=500) documents (not elements) then the pool is full. If not then the second result is taken from each run, the pools is again de-duplicated, and checked for exceeding *n*. The process continues in this way until at least *n* documents are in the pool or no more documents can be selected. Just those documents in the pool are assessed by a human. The validity of pooling at TREC has been studied extensively [14] in short, it is sound (however we accept that INEX pooling is subtly different).

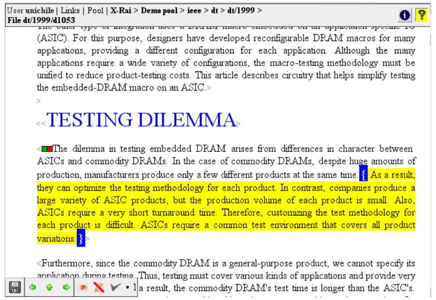
Identifying the relevance within a document is done using a yellow highlighting method as shown in Figure 1. Documents from the pool are presented to the assessor who identifies relevant passages. Relevant documents must contain a relevant passage. The crossover between the relevant passages and XML-elements in a run is used to measure performance (precision and recall).

Even using pooling and yellow highlighting, the mean time to judge a topic at INEX 2006 was 6:51:00, nearly seven hours. It took 1:02, just over a minute, per relevant document and 44 seconds per irrelevant document [6].

We believe there remains further room for improvement in the time to assess – particularly by moving away from the model that the topic should be assessed by the original topic author. Others have already analysed the agreement level between multiple assessors [4; 9] and they are inline with those of TREC and it does appear valid to assess with a different assessor than the topic author.

A further move away from using the original topic assessor is to use more than one assessor per topic. The pool might be split in half and two assessors asked to assess half the pool each. Further, a pool might be split into seven pieces and seven students each asked to spend one hour assessing. If this were possible assessing might additionally be used as a teaching exercise.

Before embracing such a change it is essential to test the validity of it. We do this by synthetically generating runs assessed by multiple assessors and comparing the result to that seen when only one assessor is used.



User **unichile** | Links | Pool | **X-Rai** > **Demo pool** > **ieee** > **dt** > **dt/1999** >
File dt/1999/d1053

(ASIC). For this purpose, designers have developed reconfigurable DRAM macros for many applications, providing a different configuration for each application. Although the many applications require a wide variety of configurations, the macro-testing methodology must be unified to reduce product-testing costs. This article describes circuitry that helps simplify testing the embedded-DRAM macro on an ASIC.>

>

<<TESTING DILEMMA>

<The dilemma in testing embedded DRAM arises from differences in character between ASICs and commodity DRAMs. In the case of commodity DRAMs, despite huge amounts of production, manufacturers produce only a few different products at the same time. { As a result, they can optimize the testing methodology for each product. In contrast, companies produce a large variety of ASIC products, but the production volume of each product is small. Also, ASICs require a very short turnaround time. Therefore, customizing the test methodology for each product is difficult. ASICs require a common test environment that covers all product variations. }>

<Furthermore, since the commodity DRAM is a general-purpose product, we cannot specify its application during testing. Thus, testing must cover various kinds of applications and provide very

a result, the commodity DRAM's test time is longer than the ASIC's.

Figure 1: Yellow Highlighting Assessment.

3. Experimental Equipment

3.1. INEX Topics and Assessments

At INEX 2006 there were 125 topics. Of those 15 were assessed by two separate assessors neither of which was aware that topics were being assessed by someone else. Clearly in each case at least one of the assessors was not the topic author.

During a session at the INEX 2006 workshop an experiment was run in which some documents from those 15 topics were assessed by further assessors [11]. A new pool was generated by applying the top-n pooling strategy with *n* set to 100. For reasons outlined elsewhere [11] this second pool was not a complete subset of the original, however the mean crossover was 98%.

Not all the assessors completed the assessment task due to the time constraint (of about an hour and a half). Trotman *et al*. [11] generated, from both experiments, a set of assessments for the 15 topics consisting of between 3 and 5 assessors for each topic. To get into their set an assessor was required to complete at least half the assessment task. Additionally all assessments on documents not assessed by all assessors in their set were discarded. The details are given in

Table 1 where it can be seen that, for example, topic 310 had four assessors (two of which were at the workshop) who all assessed the same 91 documents, they disagreed on the relevance of 17 of those. Each assessor may have assessed additional documents but those documents are not included.

Trotman *et al.* [11] report on the cross-assessor agreement levels in this data and report that as assessors are added the intersection decreases while the union continues to increase.

Runs submitted to INEX 2006 were scored against one set of assessments known as the official set. Those generated by the second assessor are known as the alternate set. Those generated at the workshop are known as the Dagstuhl set. We use the designation *baseline* to refer to the 15 topic subset of assessments that was used for scoring at INEX, but only for those documents that are included in Table 1.

Table 1: For each INEX topic the number of assessed documents, the number of assessors for that topic, and the number of documents for which there is disagreement over the relevance.

Topic	Docs	Assessors	Disagreement
304	135	3	19
310	91	4	17
314	130	4	26
319	78	4	19
321	132	3	8
327	78	5	7
329	86	5	13
355	83	3	9
364	56	5	17
385	87	4	5
403	113	4	16
404	104	4	30
405	99	4	3
406	67	5	25
407	132	3	7
Total	1,471	60	221

3.2. Metrics

INEX measures the performance of a relevant-in-context run using mean average generalized precision (MAgP)[5]. This metric was introduced for the first time in 2006. As the metric is new, it is not yet clear whether or not there is any inherent bias in the metric. We, instead, use mean un-interpolated average precision (MAP) as that metric is well understood and it makes our result generalizable to other evaluation forums. It should be noted that in the absence of element or passage results within a document MAgP reduces to MAP.

The INEX test set was used because we are unaware of any others for which there are more than 3 assessors per topic.

Mean un-interpolated average precision is defined as the mean, over a number of topics, of the average precision of each topic. The average precision for a topic is defined as the precision of each relevant document taken at each relevant document in the results list, divided by the number of known relevant documents. Precision, in turn, is defined as the number of relevant documents at and before a relevant document in the results list divided by the rank position of the document in the results list.

All un-assessed documents are assumed to be non-relevant. All assessed document containing any relevant content are considered relevant (see Section 2).

3.3. Runs

There were 64 runs submitted to the INEX 2006 relevant-in-context task. The elements were discarded, and identical consecutive documents conflated resulting in a relative rank ordering of documents for each of the 15 topics. We do not concern ourselves with how those runs were generated or whether CO or CAS queries were used.

3.4. Simulated Assessment Sets

For each topic, the number of documents for which the assessors are not unanimous is shown in Table 1. It ranges from 3 to 30. Examining topic 405 there were 4 separate assessors who each assessed the same 99 documents. Of those they agree on the relevance of 96 and disagree on the relevance of 3.

If the assessment for topic 405 had been split between the assessors then the relevance of any given document would be determined by the decision of any one of the assessors. The relevance of the documents for which all the assessors agree would not change, but for the 3 documents it would depend on who assessed these documents. There are 64 different ways these contentious documents might have been distributed between the different assessors, however as each document can be either relevant or not relevant there are only 8, 2^3, different possible outcomes.

In total there are 221 documents for which the relevance is under dispute giving 2^{221}, $3*10^{66}$, different possible sets of assessments that could be drawn from the data.

It is not practical to generate all these different combinations. Instead we choose to randomly sample the space to generate a large subset of them. We fully appreciate that for some topics every possible set of assessments is represented many times, but in combination with other topics, it is highly unlikely that our method will result in a duplicate.

The probability that document d is relevant to a given topic t is $P_t(r|d)$ and defined as

$$P_t(r \mid d) = \frac{N_r}{N} \quad (1)$$

Where N is the number of assessors that assessed the document and N_r is the number of those assessors who consider the document relevant. If all assessors agree a given document is relevant then $P_t(r|d) = 1$, if they all agree the document is not relevant then $P_t(r|d) = 0$, otherwise it is the proportion of assessors that considered the document relevant.

Including the baseline assessor (as we do) is synonymous with that assessor getting help from the others. An alternative is to exclude the baseline which is synonymous with that assessor subcontracting the others. With the small number of available assessors it is not practical to test this latter case.

A synthetic document assessment for document *d* (for a single topic, *t*) is generated by choosing a random number *p* the range [0,1) and then comparing it to $P_t(r|d)$. If $p < P_t(r|d)$ then the document is set to relevant, otherwise it is set to non-relevant. This is a random selection of an assessor (with replacement).

A synthetic topic assessment is generated by generating a synthetic assessment for each document assessed for that topic. For topic 405, that is 99 synthetic document assessments are needed. Likewise, a set of assessments is generated for all documents of all topics forming a synthetic assessment set. Note that a document can both be relevant to one topic and not relevant to another in the same assessment set, but must be either relevant or not relevant with respect to a single topic.

There are alternative methods of generating synthetic assessment sets; however we believe they are equivalent.

4. Experiments

We conducted three experiments to investigate the effect of using multiple assessors per topic. In the first we correlate the performance using the baseline assessments to that of the 10,000 synthetic assessments. In the second we identify the difference in MAP at which using multiple assessors has a significant effect on relative system performance. Finally, in the third experiment we examine the top 10 runs submitted to the INEX 2006 relevant-in-context task and look for differences that would be identified using synthetic assessments and check whether those differences are statistically significant.

4.1. Experiment 1

The aim of this experiment was to determine if a measurably real difference exists between the relative rank order produced using one assessor and that of using multiple assessors. That is, would there have been a different result if multiple assessors had been used?

First, using the baseline assessment set, the mean average precision of every run submitted to the INEX 2006 relevant-in-context task was measured. From this the relative rank order of the runs was generated and recorded.

Next, for each of the 10,000 synthetic assessment sets the mean average precision for each run was computed and recorded. The relative rank order was generated and recorded.

Then, using Spearman’s rank correlation, the correlation between the baseline rank order and each of the 10,000 synthetic rank orders was generated and recorded.

Finally, the mean of the correlation coefficients was computed and recorded.

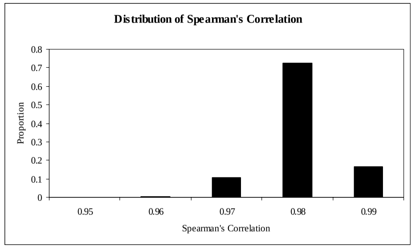


Figure 2: Distribution of the 10,000 Spearman’s correlation coefficients. Each measures the correlation between the baseline and a single synthetic assessment.

4.2. Results 1

The 10,000 Spearman’s rank correlation coefficients were bucketed into buckets of size 0.01 and are plotted in Figure 2. A correlation coefficient of 1 occurs when there is complete agreement in relative rank orders. A coefficient of -1 is seen in perfect disagreement. A coefficient of 0 occurs when there is no correlation (the ranks are independent of each other).

Of the 10,000 correlations, 7,245 (72%) lie in the range [0.98-0.99). All 10,000 fall above 0.95, and 89% fall above 0.98. The mean correlation coefficient was 0.986.

There is a very strong (near perfect) correlation between the relative rank order seen using a single assessor and that which would have been seen using multiple assessors. From this it is reasonable to conclude that if multiple assessors had been used there would not have been a measurably real difference in the result of the relevant-in-context task at INEX 2006.

4.3. Experiment 2

The aim of this experiment was to identify whether the difference between the mean average precision scores of two runs was plausibly real. The methodology is similar to that of Voorhees & Buckley [13] who generated assessment sets by splitting a single set of assessments into multiple groups for a similar purpose. Our experiment differs from theirs in so far as we use multiple synthetic assessment sets and compare to a baseline.

First the relative rank order of all the runs was generated against the baseline assessments (as outlined in Experiment 1). Then for all possible pairs of runs, the difference in mean average precision was computed and recorded.

Next, for each of the 10,000 synthetic assessment sets, the relative rank order of the two runs was computed and compared to the baseline order. The number of times a switch was seen was recorded.

4.4. Results 2

We define a significant difference as a greater than 5% chance of a relative rank order switch compared to the baseline.

Figure 3 shows, for each run pair, the probability, when multiple assessors are used, of seeing a different outcome than when the baseline assessor was used. Scores vary from 0.00 to 0.99.

Figure 4 shows the same result as Figure 3, but grouped into buckets representing differences from baseline mean average precision of 0.01.

The probability of a switch in relative rank order decreases as the difference in mean average precision increases.

This result is as expected because runs that vary very little are far more sensitive to small changes than those that vary a great deal. The figures show that when the difference in mean average precision is greater than 0.05 there is essentially no chance of a switch.

Voorhees & Buckley [13] suggest that a difference in mean average precision of less than 0.05 is insufficient to conclude that a difference is meaningful. In our experiments the probability of getting a different result when mean average precision scores differ by more than 0.05 is essentially zero.

From this it is reasonable to conclude that if multiple assessors had been used there would not have been a material difference in the result of the relevant-in-context task at INEX 2006. Any differences that would have been seen would only have occurred where runs were not significantly different anyway.

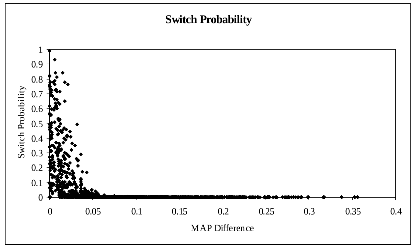


Figure 3: For all possible pairs of runs submitted to the INEX 2006 relevant-in-context task, the probability of a switch in relative rank order from the baseline rank order.

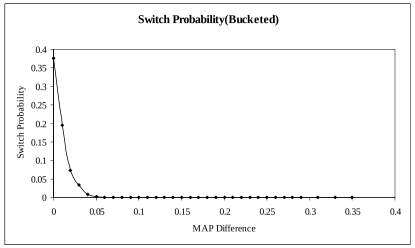


Figure 4: The results in Figure 3 bucketed into differences of 0.01.

4.5. Experiment 3

Of particular interest in collaborative information retrieval evaluation forums is the performance of the top 10 systems. Far more can be learned by examining what is happening there than by examining the worst 10 runs.

The aim of experiment 3 is to determine whether or not a different outcome in the INEX 2006 relevant-in-context task would have been seen if multiple assessors had been used instead of a single assessor.

All runs were ranked using the baseline assessment set and then Experiment 2 was repeated using only the runs that ranked in the top 10.

Additionally, all pairs of runs that showed a probability of switching of greater than 5% were identified. Then, for each of those pairs of runs the average precision of each run was measured and recorded for each topic.

Finally, a two-tailed *t*-test was performed on the average precisions (of 15 topics) to identify whether or not there was a statistically significant difference between the pairs of runs. The *t*-test was chosen because it is believed to be the most appropriate measure of significance [7].

We expect all pairs of runs showing a greater than 5% chance of switching order to have no significant difference in performance as measured by the *t*-test. If, for example, two runs differ by only one document, and one run considers that document relevant and the other not, then the better run is simply a function of whether or not that document is considered relevant in a synthetic assessment set. If exactly half the assessors consider the document relevant then the probability of a switch is expected to be 0.50, but there is no significant difference between the runs.

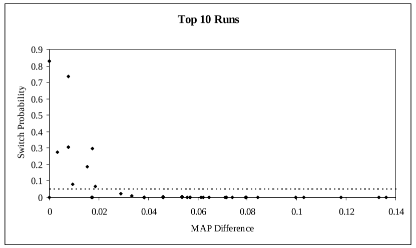


Figure 5: Observed chance of a switch in relative performance of the top 10 runs submitted to INEX 2006 relevant-in-context task.

Table 2: Two-tailed *t*-test p value for all pairs of runs showing a greater than 5% chance of switching relative rank order when multiple assessors are used. Run-codes have been used; see Table 3 for the run names.

Run A	Run B	p
41-A	40-A	0.82
40-C	41-A	0.85
40-B	41-A	0.85
19-A	16-A	0.94
19-A	16-B	0.63
16-C	40-C	1.00
16-C	40-B	1.00
16-C	41-A	0.88
16-C	40-A	0.71
16-A	16-B	0.61

Table 3: Names of INEX 2006 relevant-in-context runs. Codes are used in Table 2 for clarity.

Code	Group	INEX run name
16-A	16	zet-dirichlet-AC
16-B	16	zet-okapi-AC
16-C	16	zet-pivot-AC
19-A	19	TOPX-CO-AllInContext-exp
40-A	40	Okpi-2-7-0.75-2006-SansOvlp-ParDocSansArticleBdy
40-B	40	OkTg-Lineaire-RkSym-100pcent-Okpi-2-7-0.75-DocDoxParent-VectTagFamClass-2006etiq-50it-2006-SansOvlp-ParDocument
40-C	40	OkTg-Lineaire-RkSym-100pcent-SemiSup5-Okpi-2-7-0.75-DocDoxParent-VectTagFamClass-2006etiq-50it-2006-SansOvlp-ParDocument
41-A	41	A_CO_ARTorNAME

4.6. Results 3

Figure 5 shows, when using the 10,000 synthetic assessment sets, and only the top 10 runs, the probability of a getting a different result from that of the baseline assessment set. The 5% probability that we consider significant is drawn dotted. There are 10 pairs of runs exhibiting instability, but some are very close in the figure and are hard to see.

Table 2 lists each pair of runs for which the probability of a switch in relative performance is greater than 5% along with the *p* value from a two-tailed *t*-test. Short codes have been used for clarity. Table 3 gives the full names of the runs and the participating group's id given the short code. Of note is that the top 10 runs were produced by only 4 separate participating groups.

In all cases where there is a significant chance of getting a different relative rank order, no significant difference between runs measured with a *t*-test is seen. That is, using multiple assessors is only unstable in the cases where there is no significant difference between the runs anyway. *T*-tests on all pairs of the top 10 runs suggests that only 6 pairs differ significantly.

From this it is reasonable to conclude that if multiple assessors had been used there would not have been a material difference in the result of the relevant-in-context task at INEX 2006.

5. Discussion and Conclusions

Information retrieval test sets consist of three parts, a document collection, a set of topics and a set of assessments. At INEX the assessments are built by the participants themselves who in 2006 were expected to assess 3 topics each, totaling about a working week.

We are particularly interested in reducing this assessment load for two reasons: First, a week is a substantial amount of time to spend doing nothing other than assessing topics. Second, we are looking for methods of using our participation in the evaluation forums as a teaching tool.

One way to achieve both goals is to ask a class of students to assess topics. An individual topic might be split amongst seven students each working independently, but collectively assessing an entire topic.

Such an approach to assessment is only valid if it can be shown that the result of using multiple assessors per topic would be indifferent from that of using a single assessor. Indeed, we have shown that the result would have been the same if multiple assessors had been used in the INEX 2006 relevant-in-context task.

INEX 2006 was chosen because for 15 of those topics there were between 3 and 5 assessors each – making it possible to construct synthetic multiple-assessor assessment sets. The relevant-in-context task was chosen because it is the most plausible XML-IR retrieval task.

First the relative rank order of the runs was measured using a baseline assessment set derived from the official INEX assessments.

10,000 random multiple-assessor assessment sets were then generated.

Next the relative rank order of the runs was measured with each of the synthetic assessment sets. These were shown to very strongly correlate with the original.

The point of instability was examined and it occurs when the mean average precision difference between two runs is less than about 0.05. Differences in MAP of this small amount are not very interesting.

Focusing on the top-10 runs, 10 pairs showed a greater than 5% chance of producing a different set of results when multiple assessors were used instead of a single assessor. Investigation into the significance of difference between these runs suggested that none existed.

From this it can be said that on the data that was used in the experiments, the only place where a difference in relative rank order of runs would be seen is where there is no significant difference between runs anyway.

We believe that, in the light of no evidence to the contrary, it is reasonable to conclude that using more than one assessor per topic is valid for information retrieval evaluation forums.

This result could have a profound effect on the nature of these forums. INEX 2006 assessment was allotted 50 days, in part because the assessors (being the participants) had to assess in addition to their ordinary duties. If a large number of people could be coordinated simultaneously, as is inherent in the teaching of classrooms of students, the assessment phase might be reduced to just a few days.

Some participants find it difficult to complete the assessment task. The result we present suggests that it is valid to ask another group (who might have spare resources) to complete the assessment of these topics. Doing so would both increase the number of topics used for evaluation and protect the investment in assessing already contributed by the original assessors.

After examining the effect of using multiple assessors per topic (rather than the conventional single assessor per topic) we conclude that it would have no material effect on the evaluation of search engine performance.

6. Acknowledgements

Partly funded by a University of Otago Research Grant.

7. References

[1] Denoyer, L., & Gallinari, P. (2006). The Wikipedia XML corpus. In *Proceedings of the INEX 2006 Workshop*.

[2] Malik, S., Trotman, A., Lalmas, M., & Fuhr, N. (2006). Overview of INEX 2006. In *Proceedings of the INEX 2006 Workshop*.

[3] Ogilvie, P., & Lalmas, M. (2006). Investigating the exhaustivity dimension in content oriented XML element retrieval evaluation. In *Proceedings of the 15th ACM Conference on Information and Knowledge Management (CIKM 2006)*.

[4] Pehcevski, J., & Thom, J. A. (2005). HiXEval: Highlighting XML retrieval evaluation. In *Proceedings of the INEX 2005 Workshop*.

[5] Pehcevski, J., & Thom, J. A. (2007). Evaluating focused retrieval tasks. In *Proceedings of the SIGIR 2007 Workshop on Focused Retrieval*, 33-40.

[6] Piwowarski, B., Trotman, A., & Lalmas, M. (2007 (submitted)). Sound and complete relevance assessments for XML retrieval.

[7] Sanderson, M., & Zobel, J. (2005). Information retrieval system evaluation: Effort, sensitivity, and reliability. In *Proceedings of the 28th ACM SIGIR Conference on Information Retrieval*, 162-169.

[8] Spink, A., Wolfram, D., Jansen, B. J., & Saracevic, T. (2001). Searching the web: The public and their queries. *Journal of the American Society for Information Science and Technology,* 53(2):226-234.

[9] Trotman, A. (2005). Wanted: Element retrieval users. In *Proceedings of the INEX 2005 Workshop on Element Retrieval Methodology, Second Edition*, 63-69.

[10] Trotman, A., N.Pharo, & Lehtonen, M. (2006). XML-IR users and use cases. In *Proceedings of the INEX 2006 Workshop*.

[11] Trotman, A., Pharo, N., & Jenkinson, D. (2007). Can we at least agree on something? In *Proceedings of the SIGIR 2007 Workshop on Focused Retrieval*, 49-56.

[12] Voorhees, E. M. (2005). Overview of TREC 2005. In *Proceedings of the 14th Text REtrieval Conference (TREC-14).*

[13] Voorhees, E. M., & Buckley, C. (2002). The effect of topic set size on retrieval experiment error. In *Proceedings of the 25th ACM SIGIR Conference on Information Retrieval*, 316-323.

[14] Zobel, J. (1998). How reliable are the results of large-scale information retrieval experiments? In *Proceedings of the 21st ACM SIGIR Conference on Information Retrieval*, 307-314.

Integration of Information Filtering and Data Mining Process for Web Information Retrieval

Xujuan Zhou, Yuefeng Li, Peter Bruza, Yue Xu

Faculty of Information Technology
Queensland University of Technology
QLD 4000 Australia

{*x.zhou, y2.li, p.bruza, yue.xu*}*@qut.edu.au*

Abstract *This paper examines a new approach to Web information retrieval, and proposes a new two stage scheme. The aim of the first stage is to quickly filter irrelevant information based on the user profiles. The proposed user profiles learning algorithm are very efficient and effective within a relevance feedback framework. The aim of the second stage is to apply data mining techniques to rationalize the data relevance on the reduced data set. Our experiments on RCV1 (Reuters Corpus Volume 1) data collection which is used by TREC in 2002 for filtering track show that more effective and efficient access Web information has been achieved by combining the strength of information filtering and data mining method.*

Keywords Information filtering, User profiles, Data mining, Pattern taxonomic model

1 Introduction

Web search engines are designed based on traditional information retrieval techniques to return a set of potential relevant documents that match the user's direct query. The queries submitted to search engines by Web users are generally very short containing only two or three words [2]. Although such simple keywords approach works very well if the short query is an unambiguous term with fairly high discriminating power (e.g. using "Sweatshop" as query term) in general, these short queries can not clearly describe a user's true information search intent and they will open up the problem of vocabulary mismatch [1]. In deed, search engines provide a "one size fits all" solution to all users. This solution often leads to information overload problem.

Relevance feedback (RF) is a well known method to help users conduct searches iteratively and it has been shown that RF can significantly improve retrieval performance [7]. In this paper, we propose to construct a user profile through user's interactive feedback. In stead of requiring the user to explicitly express and specify their information needs beforehand, we alleviate the user's cognitive burden by only asking her to indicate whether a small set of documents are relevant or not. This set of user feedback is then used as training data set and a learning method will be developed to learn a user profile from training data. In this project, the user profile is constructed from the topics of a user's interest i.e., search intent. The topic in a particular document comprises the terms which represent the subjects.

The main objective of the research work presented in this paper is to develop a novel Web information retrieval system which integrates information filtering and data mining strategies to provide more precise results for the Web search. The remainder of the paper is organized as follows. Section 2 highlights previous researches in the related area. The proposed two-stage method of filtering and data mining will be illustrated in Section 3 and Section 4. The empirical testing results will be reported in Section 5. Section 6 describes the findings of the experiments and discusses the results. The concluding remarks and future researches are given in section 7.

2 Related works

In dealing with Web information overload issues, classical methodologies/techniques from information retrieval/filtering (IR/IF) and data mining have been applied separately with various success. IF systems learn user profiles from their interaction with systems and then use this information to analyze new documents. The profiles can be constructed using a variety of learning techniques including the vector space model, genetic algorithm, and the probabilistic model or clustering. Recently, a number of ontology-based user profiles models have been developed, e.g., [3, 10].

Data mining is the process of automatically extracting useful knowledge from large data sets. Web mining is concerned with data mining on the Web. Many Web data mining methods have been developed to underpin IF system. For example,Web usage mining provide an excellent way to learn about users' interest [8]. The authors of [9] have developed a pattern taxonomy model (PTM) for Web information gathering. Many up-to-

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

date Web mining techniques (e.g., sequential association rules, closed-pattern based non-redundant association rules and rough association rules) have been intergraded into this method.

The idea of integrating IF and data mining for Web information retrieval has evolved from these two well established, but largely disparate fields. This proposed method intends to exploit the advantages of IF and data mining within the one system.

3 Learning user profiles for IF

There are two phases in combined system. The first phase comprises IF and second phase relies on data mining. The most challenging issues in filtering is to develop a method to learn user profiles efficiently and effectively from very limited user intervention and to implement a way to "filter" information from a huge collection of documents. This section presents a user profile learning method based on rough association rule mining whereby only positive feedback is required [4, 5].

3.1 User profile construction

User profiles will be represented by an ontology. Syntactically we assume that the ontology consists of primitive classes and compound classes. The primitive class is constructed from terms in Θ. The primitive classes are the smallest concepts that cannot be assembled from other classes. However, they may be inherited by derived concepts or their children. Then a set of primitive objects (terms) can be selected from the set of keywords by using the existing background knowledge.

Let D be a training set, which includes a non-empty set of positive documents D^+ and a set of negative documents D^-. Let $\Theta = \{t_1, t_2, \ldots, t_k\}$ be a set of selected terms (or primitive classes).

A set of terms is referred to as a $termset$. Given a document d (or a paragraph) and a term t, $tf(d,t)$ is defined as the number of occurrences of t in d. A set of term frequency pairs,

$$P = \{(t,f) | t \in T, f = tf(t,d) > 0\}$$

is referred to as a *pattern* in this paper.

Let $termset(P) = \{t | (t,f) \in P\}$ be the termset of P, pattern P_1 equals to pattern P_2 if and only if $termset(P_1) = termset(P_2)$. A pattern is uniquely determined by its termset. Two patterns should be composed if they have the same termset (or they are in a same category). To compose two patterns which have same termset, the composition operation, $\oplus$, that defined in [4, 5] will be used to generate new patterns.

The compound classes are constructed from a set of primitive classes: $\Omega = \{p_1, p_2, \ldots, p_k\}$. There are "is-a" and "part-of"relations between these objects. A document is irrelevant if its any part-of section does not include any pattern.

Let $p =< termset(p), wd(p) >$, we can also view it as a rough association rule which has the form of

$$< termset(p), wd(p) > \rightarrow positive,$$

where $termset$ is a set of selected terms, and wd is a weight distribution of these terms in the rule.

Rough association rules can be discovered from a set of positive documents (or paragraphes) by extracting patterns of term frequency pairs, composing patterns with the same termsets, and normalizing the weight distributions (see [4] or [5]).

Let $O = \{(p_1, N_1), (p_2, N_2), \ldots, (p_n, N_n)\}$ be a set of compound objects (discorded patterns), where p_i are patterns $(1 \leq i \leq n)$ and N_i denote numbers of patterns that composed together. A support function can be attained from O, which satisfies:

$$support(p_i) = \frac{N_i}{\sum_{(p_j,N_j)\in O} N_j} \quad (1)$$

for all $(p_i, N_i) \in O$.

To describe the semantic relations between compound classes (discovered patterns), we use a common hypothesis space Θ. We then can map the discovered patterns onto the common hypothesis. The following mapping is designed for this purpose:

$$\xi : DP \rightarrow 2^{\Theta} - \{\emptyset\}, \text{ such that}$$

$$\xi(p_i) = termset(p_i) \quad (2)$$

where $DP = \{p_i | (p_i, N_i) \in O\}$.

Finally, we can obtain a probability functions pr_ξ on the set of terms to represent the discovered in the discovered patterns, which satisfies:

$$pr_\xi(t) = \sum_{p \in DP, t \in \xi(p)} \frac{support(p)}{|termset(p)|} \quad (3)$$

for all $t \in \Theta$.

3.2 Filtering

The objective of filtering phase is to filter our non-relevant incoming documents. To determine a reasonable threshold, in this paper, we discuss how to classify incoming documents into three regions: relevant, boundary, and irrelevant documents region according to the above discovery.

Let p be a pattern and d be a new incoming document. Our basic assumption is that d should be relevant if $termset(p) \subseteq d$. The set of incoming documents that satisfy $termset(p) \subseteq d$ is called the *covering set* of p and denoted as $[p]$. The **positive region** (*POS*) is the union of all covering sets for all $p \in DP$.

The set of incoming documents that satisfy $\exists p \in DP \Rightarrow termset(p) \cap d \neq \emptyset$ is called the **boundary region** (*BND*). Also, the set of incoming documents that satisfy $\forall p \in DP \Rightarrow termset(p) \cap d = \emptyset$ is called the **negative region** (*NEG*). Given an incoming document d, the decision rules can be determined naturally as follows:

$$\frac{\exists p \in DP \Rightarrow termset(p) \subseteq d \neq \emptyset}{d \in POS}$$

$$\frac{\exists p \in DP \Rightarrow termset(p) \cap d \neq \emptyset}{d \in BND}, \text{ and}$$

$$\frac{\forall p \in DP \Rightarrow termset(p) \cap d = \emptyset}{d \in NEG}.$$

The probability function pr_ξ on Θ (see Equation 3) has the following property:

$$\sum_{t \in d} pr_\xi(t) \geq \min_{p \in DP}\{\sum_{t \in \xi(p)} pr_\xi(t)\} \qquad (4)$$

for all $d \in POS$.

We use $\min_{p \in DP}\{\sum_{t \in \xi(p)} pr_\xi(t)\} + \alpha$ as the threshold. A very important conclusion we can draw from the above analysis is that the above threshold can retain all POS incoming documents and part of BND incoming documents, where α is an experimental coefficient that is used for obtaining the part of BND incoming documents.

By incorporating filtering into the web search, likely irrelevant data will be filtered out quickly at the beginning. The remaining data will be comprised of relevant, boundary or maybe a few irrelevant documents. The size of the remaining dataset is dramatically reduced.

4 Data mining

After filtering, a data mining process based on the pattern taxonomy model (PTM) [9] will be carried out on the residual data set. The following is a brief introduction to PTM.

A sequence $s =< x_1, \ldots, x_m >$ ($x_i \in T$ is a termset) is an ordered list. A sequence $\alpha =< a_1, \ldots, a_m >$ is a sub-sequence of another sequence $\beta =< b_1, \ldots, b_n >$, denoted by $\alpha \subseteq \beta$, if and only if $\exists i_1, \ldots, i_m$ such that $1 \leq i_1 < i_2 \ldots < i_m \leq n$ and $\alpha_1 \subseteq \beta_{i_1}, \alpha_2 \subseteq \beta_{i_2}, \ldots, \alpha_m \subseteq \beta_{i_m}$. A sequential pattern s is a *very closed sequential pattern* of s' if $s \subseteq s'$ and $support(s) - support(s') < \lambda \times support(s')$, where λ is a small positive decimal.

The above definitions can be used to create a pattern taxonomy as depicted in Figure 1, where a, b, c, and d are terms, the arrows are "is-a" relation, e.g., phrase $< (a)(b) >$ is a sub-sequence of $< (a)(b)(c) >$.

If the frequency is used to define the *support* function for all patterns, then $support(< (a)(b) >) \geq support(< (a)(b)(c) >)$. In general, 3 sub-sequence patterns of $< (a)(b)(c) >$ can be obtained. They are $< (a)(b) >$, $< (a)(c) >$ and $< (b)(c) >$. If patterns have supports which are very closed to their parents supports then these patterns are called non-closed patterns. The not very closed sequential patterns will be removed. e.g.,$< (a)(c) >$ in Fig. 1 has been pruned.

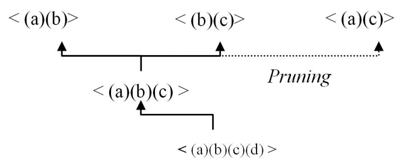


Figure 1: Pattern taxonomy.

After a pattern taxonomy has been extracted from a training set, it is utilized to calculate $pr(d)$ which is the relevance degree of each new incoming document d for a given topic. The following is the procedure of making decisions to return relevant document to the user:

1. Find all longest patterns in document d;
   e.g., ($< (a)(b)(c) >$) is a longest pattern
   if ($< (a)(b)(c)(d) >$) does not appear in d.

2. Determine $pr(d)$ according to the taxonomy.
   e.g., $pr(d) = support(< (a)(b)(c) >) + support(< (a)(b) >) + support(< (b)(c) >)$.

5 Experiments

To evaluate the effectiveness of the filtering and retrieval function used by our proposed system, several experiments have been conducted.

5.1 Dataset

The standard TREC test collections RCV1 was used to test the effectiveness of the proposed model. TREC has developed and provided 100 topics for the filtering track aiming at building a robust filtering system. Each topic is divided into two sets: training set and testing set. Our experiments use the Split of TREC-10/2000. Document relevance judgments have been supplies for each topic. The set of one hundred TREC topics is used to represent the diverse Web user's information needs. The experiments simulated user feedback by assuming that the user would recognize as relevant an officially judged relevant document. The documents have been pre-processed by removing stop-words and stemming terms before they are used in all experiments.

5.2 Baseline methods

Two baseline models are used: BM25 model and a PTM-based model. BM25 [6] is one of sate-of-the-art retrieval function used in document retrieval, such as Web search. In this paper, the following weighting function is used: given a query Q, containing keywords $q_1, q_2 \ldots, q_n$, the BM25 score of a document D is:

$$\begin{aligned} score(D,Q) &= \sum_{i=1}^{n} \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \\ &* \frac{f(q_i, D) * (k_1 + 1)}{f(q_i, D) + k_1 * (1 - b + b * \frac{|D|}{avgdl})} \end{aligned}$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i. where $f(q_i, D)$ is q_i's term frequency in the document D, $|D|$ is the length of the document D (number of words), and $avgdl$ is the average document length in the text collection from which documents are drawn. Parameter settings in the experiments were $k_1 = 1.2$ and $b = 0.75$.

The PTM-based method used the pattern-based taxonomy rather than single words to represent documents. The authors of [9] have conducted experiments on TREC collection (RVC1 corpus) and have compared the performance of their model with keyword based models such as Rocchio and traditional Probabilistic model. They concluded that their method outperforms the keyword based methods.

5.3 Results

Effectiveness was measured by three means: The F_β measure ($\beta = 1$ is used in our experiments), Mean Average Precision (MAP) and the break-even (B/E) point.

The results reported in here are the average scores of B/E point, MAP and F_1 on all 100 TREC topics for all methods.

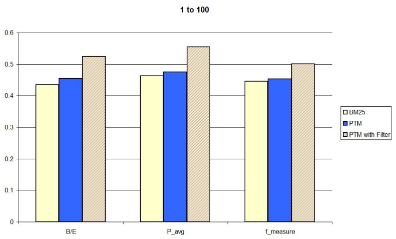


Figure 2: Results on topics 1-100 for all three methods

6 Discussion

In the filtering phase, by using the rough association rule method to build ontology-based user profiles, the only positive documents are needed. An ontology is able to provide rich semantic relationship between patterns. Therefore, the ontology-based user profiles can perhaps express user information needs and searching goals more accurately and comprehensively. Based on these profiles, most irrelevant documents are able to be filtered out and consequently the chance of generating noisy patterns is reduced significantly. In the data mining phase, a pattern taxonomy is built on a data set which has less noise hence promotes precision without negatively impacting recall.

In short, the experiment results provide evidence that the combination of filtering and data mining can improve information access significantly.

7 Conclusions

This paper illustrates a new model which integrates an ontology-based user profile filtering and pattern based data mining technology together to alleviate Web information overload and mismatch problems. The proposed method has been evaluated using standard TREC data collection with encouraging results.

Compared with the orthodox data mining method PTM the experiments based on the new method demonstrated that the performance of information retrieval can be significantly improved. The improvement of the new method is mainly due to the success of irrelevant information removal by the filtering process.

Acknowledgements This paper was partially supported by Grant DP0556455 from Australian Research Council.

References

[1] Peter Bruza, Robert McArthur and Simon Dennis. Interactive internet search: keyword, directory and query reformulation mechanisms compared. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 280–287, New York, NY, USA, 2000. ACM.

[2] Bernard J. Jansen, Amanda Spink and Tefko Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Inf. Process. Manage.*, Volume 36, Number 2, pages 207–227, 2000.

[3] Y. Li and N. Zhong. Ontology-based web mining model. In *IEEE/WIC International Conference on Web Intelligence*, 2003.

[4] Yuefeng Li and Ning Zhong. Rough association rule mining in text documents for acquiring web user information needs. In *Web Intelligence*, pages 226–232, 2006.

[5] Yuefeng Li and Ning Zhong. Mining rough association from text documents for web information gathering. *T. Rough Sets*, Volume 7, pages 103–119, 2007.

[6] Stephen E. Robertson, Steve Walker and Micheline Hancock-Beaulieu. Okapi at trec-7: Automatic ad hoc, filtering, vlc and interactive. In *TREC*, pages 199–210, 1998.

[7] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, Volume 41(4), pages 288–97, 1990.

[8] Jaideep Srivastava, Robert Cooley, Mukund Deshpande and Pang-Ning Tan. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, Volume 1, Number 2, pages 12–23, 2000.

[9] S.T.Wu, Y.Li, Y. Xu, B. Pham and P.Chen. Automatic pattern-taxonomy extraction for web mining. In *the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 242 – 248, China, 2004.

[10] X. Zhou, Y. Li, Y. Xu and R. Lau. Relevance assessment of topic ontology. In *The Fourth International Conference on Active Media Technology*, Relevance Assessment of Topic Ontology, 2006.

Does brandname influence perceived search result quality? Yahoo!, Google, and WebKumara

Peter Bailey
CSIRO ICT Centre
Canberra, Australia
peter.bailey@csiro.au

Paul Thomas
Australian National University
Canberra, Australia
paul.thomas@anu.edu.au

David Hawking
CSIRO ICT Centre
Canberra, Australia
david.hawking@csiro.au

Abstract *Improving the quality of search engine results is the goal of costly efforts by major Web search engine companies. Using* in situ *side-by-side result set comparisons and random assignment of brandnames to result sets, we investigated whether perceptions of quality were influenced by brand association. In the first experiment (15 searchers) we found no significant preference for or against results labelled "Google" relative to those labelled "Yahoo!". In the second experiment (20 searchers) result sets were again generated by Google and Yahoo! but were randomly labelled "Yahoo!" or "WebKumara" (a fictitious name). Again, we found no significant preference for one brandname label over the other. Contrary to previous findings, we found a statistically significant preference for Google-generated results over those of Yahoo! when data from three separate experiments (total 70 subjects) was combined.*

Keywords Information retrieval

1 Introduction

The quality of search results is vitally important to the success of major Web search engines. It is understood that the operators of these search engines continually monitor their result quality and that of their competitors. However, over recent years, the market share of searches carried out by users on different search engines has changed. Recent reports credit Google with an increasing market share [4] at the expense of Yahoo! and MSN in particular,[1] although there does not appear to be a strong actual difference in result quality [6]. Other factors, such as a difference in *perceived* quality, may be contributing to this difference in market share.

2 Related work

Objective Web search engine evaluation is known to be a difficult task, due to the inability to compare individual search engines against a common corpus. Hawking et al. compared twenty public search engines using TREC Web track-based methods using a set of 54 queries from real Web search engine logs [1]. They compared 20 different Web search engines across seven different standard IR measures including P@n ($n < 20$), MRR1, relative recall, and average precision.

Tang and Sun rejected precision/recall metrics as being inappropriate, and instead investigated a small number of new user effort sensitive evaluation measures, from a set of 8 topics from 4 PhD students in [5]. This study, while interesting, may have insufficient topics to be reliable. (It is widely accepted in the IR community that typically 50 or more topics are required for such investigations [3], though the exact number depends on the statistical power required, the size of the effect to be observed and the number of variables present in the experimental design.)

Jansen et al. [2] have investigated branding, to understand why some search engines have large market share despite result sets being of similar quality. To control result quality, one set of results was generated for each of four queries; these result sets were branded with logos and other design elements from Google, Yahoo!, MSN, and AI^{2}RS to give a total of sixteen "result sets". Participants were asked to judge the relevance of each result. Sets branded with Yahoo! had highest judged "average precision", despite being identical to the others, and Google, MSR, and AI^{2}RS followed. Jansen et al. used only four queries however and did not report on the statistical significance of their results.

Thomas and Hawking [6] present a new method for allowing in-context judgements by real users (and their real information needs and search queries) with side-by-side comparisons between different systems. They used this method to report on user perceptions of

[1] Note that in markets such as China and Korea local engines are reported to dominate.

Proceedings of the 12th Australasian Document Computing Symposium. Melbourne, Australia, December 10, 2007.

result quality on two anonymised whole-of-Web search engines. Users were asked to use the search interface for their normal day-to-day search needs, and to judge which of the two result lists was preferred or if no difference could be detected. Judgements, including of "no difference" were not compulsory. Unlike the method of Jansen et al., Thomas and Hawking were able to use naturally-arising queries and to compare complete result sets rather than judging individual documents; however, they did not consider the effect of branding.

3 First branding experiment

We replicated the Thomas and Hawking experiment and method: for each user-generated query, two result sets were presented side by side in a random order and users were given the opportunity to indicate which (if either) was "better". "Better" was not further defined. Instead of anonymising the Web search engines as in the Thomas and Hawking work, we identified the individual result lists with the brand names of two major search engine companies. One list was labelled "results from Google" and the other "results from Yahoo!"; however, the labels were assigned randomly so that they only reflected the true source of the results 50% of the time. The experimental interface is illustrated in Figure 1.

Note that results are branded only with a name, rather than for example colours and logos. Result colour layout among all the major search engines has consolidated around a white background, blue title hyperlinks, black snippet text, and green url and meta information.

19 users participated, of whom 15 submitted at least one usable query. (A usable query is one where a preference is expressed, even if it was "no difference".) In total the users submitted 370 queries and expressed 123 definite preferences. A definite preference is where a user judged one list to be better than another; no preference is where the "no difference" judgement is made, or no selection is made at all. Since the experiment aimed to find out whether a difference existed, only definite preferences were considered as positive evidence. Participants' demographics are summarised in Table 1. Participants had a varied range of educational disciplines and employment positions and organisations, but there is a bias towards postgraduate education levels.

Results are shown in Table 2. Note that the numbers in the results table are the number of people, not the number of judgements, since we wish to establish the number of individual users who had a discernible preference one way or the other over the totality of their own searches (not on any individual search carried out).

Sex	Male: 10, female: 9
Education	Postgraduate degree: 13, first degree: 5, other: 1
Age	26–68 (mean 34.9, std. dev 8.4 years)

Table 1: User demographics for the first experiment.

Our analysis considers both the real and the labelled provider of the preferred result sets. These may be different: for example, a participant who expressed preferences for 5 result sets from Google and 2 from Yahoo! will be recorded as preferring results from Google overall. Since labels are applied at random, these seven preferred sets may have been labelled with "results from Google" in only 1 case and "Yahoo!" in the other 6; this will be recorded as a preference for results labelled "Yahoo!".

If a participant had an equal number of result sets marked from each provider or label, no overall preference was recorded. These were treated conservatively, as evidence against both alternatives.

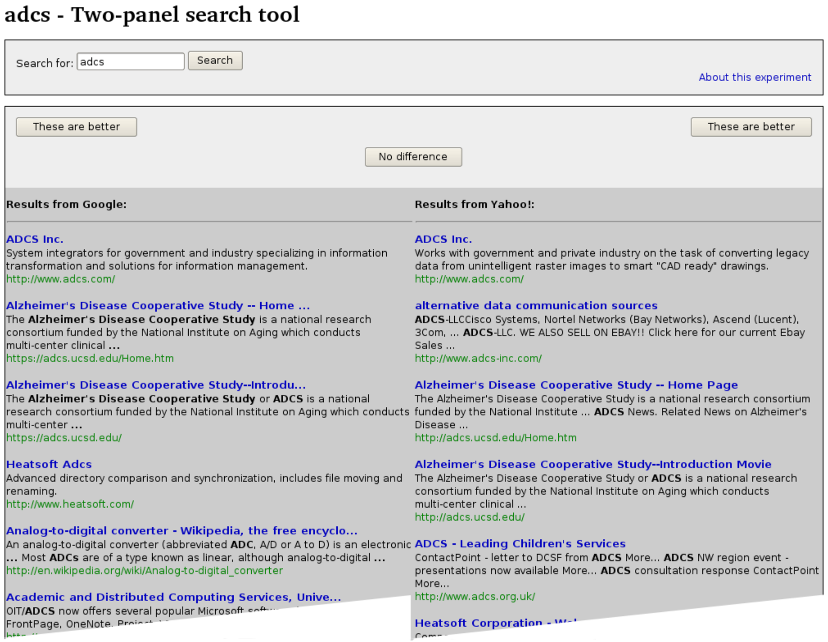
We used both binomial sign tests (less powerful) and Wilcoxon signed-rank tests (more powerful) to assess the branding results. No significant difference was found with either test. We conclude that our participants were not influenced by the brand reputation of either search engine.

4 Second branding experiment

Given this result, we decided to conduct a second experiment to see whether users would prefer the results branded as coming from a well-known search engine (Yahoo!) to the results branded as coming from an unknown search engine. We invented the name of a search engine — "WebKumara" — for this purpose. However, in all other respects the experimental method remained unchanged. Thus results were provided by both Google and Yahoo! and randomly branded as coming from either WebKumara or Yahoo!, and the quality of search results was unchanged.

20 users participated, submitted 284 queries, and expressed 115 definite preferences. Results are shown in Table 2. (User demographics were similar to those in experiment 1.)

Binomial sign tests and Wilcoxon signed-rank tests were performed on these results. Neither test revealed a significant difference between results labelled "WebKumara" and those labelled "Yahoo!". Searchers do not seem to prefer result rankings associated with a well-known search engine over those from an unknown one.



adcs - Two-panel search tool

Search for: adcs Search

About this experiment

These are better | No difference | These are better

Results from Google:

ADCS Inc.
System integrators for government and industry specializing in information transformation and solutions for information management.
http://www.adcs.com/

Alzheimer's Disease Cooperative Study -- Home ...
The **Alzheimer's Disease Cooperative Study** is a national research consortium funded by the National Institute on Aging which conducts multi-center clinical **...**
https://adcs.ucsd.edu/Home.htm

Alzheimer's Disease Cooperative Study--Introdu...
The **Alzheimer's Disease Cooperative Study** or **ADCS** is a national research consortium funded by the National Institute on Aging which conducts multi-center **...**
https://adcs.ucsd.edu/

Heatsoft Adcs
Advanced directory comparison and synchronization, includes file moving and renaming.
http://www.heatsoft.com/

Analog-to-digital converter - Wikipedia, the free encyclo...
An analog-to-digital converter (abbreviated **ADC**, A/D or A to D) is an electronic **...** Most **ADCs** are of a type known as linear, although analog-to-digital **...**
http://en.wikipedia.org/wiki/Analog-to-digital_converter

Academic and Distributed Computing Services, Unive...
OIT/**ADCS** now offers several popular Microsoft
FrontPage, OneNote,

Results from Yahoo!:

ADCS Inc.
Works with government and private industry on the task of converting legacy data from unintelligent raster images to smart "CAD ready" drawings.
http://www.adcs.com/

alternative data communication sources
ADCS-LLCCisco Systems, Nortel Networks (Bay Networks), Ascend (Lucent), 3Com, ... **ADCS**-LLC. WE ALSO SELL ON EBAY!! Click here for our current Ebay Sales ...
http://www.adcs-inc.com/

Alzheimer's Disease Cooperative Study -- Home Page
The Alzheimer's Disease Cooperative Study is a national research consortium funded by the National Institute ... **ADCS** News. Related News on Alzheimer's Disease ...
http://adcs.ucsd.edu/Home.htm

Alzheimer's Disease Cooperative Study--Introduction Movie
The Alzheimer's Disease Cooperative Study or **ADCS** is a national research consortium funded by the National Institute on Aging which conducts multi-center clinical ...
http://adcs.ucsd.edu/

ADCS - Leading Children's Services
ContactPoint - letter to DCSF from **ADCS** More... **ADCS** NW region event - presentations now available More... **ADCS** consultation response ContactPoint More...
http://www.adcs.org.uk/

Heatsoft Corporation

Figure 1: User interface for the first experiment.

First experiment (19 participants; 15 users with preferences)			
Preferred results from Google:	9 users	Preferred results labelled “Google”:	7 users
Preferred results from Yahoo!:	3 users	Preferred results labelled “Yahoo!”:	4 users
No overall preference:	3 users	No overall preference:	4 users
Second experiment (20 participants; 20 users with preferences)			
Preferred results from Google:	13 users†	Preferred results labelled “WebKumara”:	12 users
Preferred results from Yahoo!:	2 users	Preferred results labelled “Yahoo!”:	6 users
No overall preference:	5 users	No overall preference:	2 users
Aggregated Google v. Yahoo! results (70 users)			
Preferred results from Google:	45 users‡		
Preferred results from Yahoo!:	17 users		
No overall preference:	8 users		

Table 2: Overall user preferences. † significant at $\alpha = 0.05$, Wilcoxon signed-rank test. ‡ significant at $\alpha = 0.01$, same test.

These results contrast with those of Jansen et al. [2], who saw a preference for major search engine brands. We note that there are differences in methodology which may explain the difference: while Jansen et al. asked for judgements on each individual document, we asked for judgements on entire document sets. Users may consider one search engine to be better at returning relevant, but essentially duplicate, documents; this will favour the search engine on their measure but not on ours.

5 Preference between Yahoo! and Google

We aggregated data for preference between actual engines from the first and second branding experiment with those reported in the Thomas and Hawking experiment [6]. Note that if the same user participated in more than one experiment their results were combined unless they used different user-ids (which to preserve privacy would be undetectable by the experimenters).

The bottom panel of Table 2 shows that the consistent small advantage to Google across the three experiments translates to a highly significant ($p < 0.01$) preference across the aggregated users. Caution must be exercised in interpreting this result:

- 25 out of 70 users did not find Google results to be better.
- Data was collected over a period of many months during which time ranking functions and index contents may have varied considerably.
- There are significant biases in the demographics of our users — they tended to be well-educated Australian and British people.
- Although the result is highly significant, the size of the effect (of a preference for Google) may be small.

6 Conclusions

In judging result quality, users do not appear to be strongly influenced by the brandname of the search engine alleged to have generated the results, even if that brandname is totally unknown (in fact, fictitious). However, in everyday use their choice of search engine may be influenced by yet other factors besides result quality, such as other aspects of branding (e.g. colours and logos), speed of delivery of a page of results, effectiveness of advertising, or pre-loading as the default search engine in computer or browser setup.

In each of our individual experiments we found a small preference for results generated by Google over those generated by Yahoo! at the times when the data was collected. Combining all the data we found a highly significant difference in favour of those from Google, with 64% of participants exhibiting a (blind) preference for the engine with the largest market share. This result should be interpreted with caution, taking into account the caveats expressed in the previous section.

Acknowledgements We thank the many participants who carried out judgements over their queries.

References

[1] David Hawking, Nick Craswell, Peter Bailey and Kathleen Griffiths. Measuring search engine quality. *Information Retrieval*, Volume 41, Number 1, 2001.

[2] Bernard J Jansen, Mimi Zhang and Ying Zhang. Brand awareness and the evaluation of search results. In *Proc. WWW*, 2007. Poster.

[3] Mark Sanderson and Justin Zobel. Information retrieval system evaluation: Effort, sensitivity, and reliability. In *Proc. ACM SIGIR*, 2005.

[4] Danny Sullivan. comScore media metrix search engine ratings. `http://searchenginewatch.com/showPage.html?page=2156431`, August 2006.

[5] Muh-Chyun Tang and Ying Sun. Evaluation of web-based search engines using user-effort measures. *Library and Information Science Research Electronic Journal*, Volume 13, Number 2, 2003.

[6] Paul Thomas and David Hawking. Evaluation by comparing result sets in context. In *Proc. CIKM*, 2006.

Automatic Thread Classification for Linux User Forum Information Access

Timothy Baldwin, David Martinez, Richard B. Penman
CSSE
University of Melbourne
VIC 3010 Australia
`{tim,davidm,rbp}@csse.unimelb.edu.au`

Abstract *We experiment with text classification of threads from Linux web user forums, in the context of improving information access to the problems and solutions described in the threads. We specifically focus on classifying threads according to: (1) them describing a specific problem vs. containing a more general discussion; (2) the completeness of the initial post in the thread; and (3) whether problem(s) in the initial post are resolved in the thread or not. We approach these tasks in both classification and regression frameworks using a range of machine learners and evaluation metrics.*

Keywords Web Documents, Document Management

1 Introduction

Due to the sheer scale of web data, simple keyword matching is an effective means of information access for many informational web queries. There still remain significant clusters of information access needs, however, where keyword matching is less successful, due to a combination of factors including: overly-specific information needs (e.g. as specified in technical queries such as `kswapd0 hogs cpu "Slackware 10.0" "windows XP" vmware "no swapping"`); low density of relevant documents (e.g. for monolingual queries in low-density languages such as Uighur, or over domains with little web presence); the inability of keyword queries to handle data ranges (e.g. `Dell laptop $1000-1500`); and information streams spanning multiple documents, with no complete description of the contained information in any one of the documents (e.g. as occurs in logs of mailing lists). While query expansion and document normalisation can go some way towards ameliorating the first two effects [8, 2], the third question of querying and reasoning over data ranges tends to point to the need for some form of information extraction and semantic normalisation of the document content, and the final question of document segmentation requires some form of meta-document identification or threading.

In this paper, we are concerned with information access in the domain of Linux troubleshooting, based on Linux web user forum data. Consider the following scenario:

> *Kim, a Debian GNU/Linux user, notices that as a result of the latest upgrade on her laptop, she can no longer start up the GNOME desktop environment. She goes to Google to troubleshoot the problem and tries inputting the version details of various X packages and the hardware particulars of her laptop, along with different combinations of keywords such as `broken`, `not working` and `won't start`; all of the top-ranking hits are either outdated and inapplicable to the latest version packages, or irrelevant to the task at hand. She checks the archives of a selection of relevant-sounding Debian mailing lists without luck. Finally after searching the web for 2 hours she stumbles across a series of pages describing an `error` with gtk and the method for correcting the problem.*

This example (based on real-world experience) is intended to illustrate the fact that, while web search engines such as Google are remarkably successful at locating individual documents/sites typifying a well-defined information type, they have shortcomings in: (1) tracking data streams spanning multiple documents as found, e.g., in mailing list archives; (2) identifying documents associated with particular (ranges of) distribution or package versions, or identifying minor vs. major version changes in a given package; and (3) predicting acceptable lexical variance between a query and a document (e.g. between `broken` and `error`).

Our proposed alternative to conventional (web-based) information retrieval over Linux data is the ILIAD (Improved Linux Information Access by Data Mining) system which trawls the main English-based Linux web forums throughout the world, analyses each thread to arrive at a conceptual representation specified for the package, version and system information, and

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

pre-classified according to the particular problem type. Further, we aim to distill the evolution of the proposed solutions and diagnostics in a given thread into a single succinct list of factoids, and the various threads on the web pertaining to a particular problem into a single ranked list of possible solutions, with links to the original web data. Access to this information then takes the form of a web interface where the user selects the type of problem experienced (e.g. some component of a package is broken or the user wishes to configure a package in a particular way), the package or component type that is applied to (e.g. emacs or the X window system in general), and (optionally) the system and hardware configuration (e.g. Debian 3.1, or ALSA 1.1 on a ThinkPad X60), and the system returns information relevant to that query in a pre-distilled form for easy application. This paper describes the first tentative steps towards this goal, in performing thread-level estimation of the utility of a given thread for troubleshooting purposes.

2 Related work

Recently there has been growing interest in the automatic processing of discussion-based information sources, such as mailing lists or web forums. However, there is little work that focuses specifically on thread-level classification. Most related work in this area relies on speech acts to annotate utterances in the discussion, such as the approaches in [5] and [4]. Here, speech acts such as *question* and *elaboration* are detected and applied in different tasks, including identifying the roles of participants, finding unanswered threads, predicting what type of response would be appropriate in a given context, and question answering.

Another related line of research is on discussion summarisation, to provide only the most relevant information to the user. An example of this kind of work can be seen in [9], where a system is developed to summarise technical online IRC (Internet Rely Chat) discussions by clustering message segments and finding the most relevant parts using machine learning methods.

Regarding the classification of threads, [6] present an evaluation of automatic assessment of the post quality of online discussions of software. Their approach is related to our methods, and they rely on quality assessment from online users to train and test their classifiers. Using different types of features they are able to build systems with high performance for their task. Their motivation is also to deliver information to end-users in a more effective way.

3 Data description

There is a vast range of forums, newsgroups and mailing lists available on the web that cover different aspects of the Linux domain, from general discussions to very specific applications. For our purposes, we chose to target well-known, active forums, including sites that are specifically devoted to Linux (e.g. LINUXQUESTIONS and FEDORAFORUM) and general-domain resources that contain Linux-specific sections (e.g. EXPERTS EXCHANGE and GOOGLE GROUPS). In the interests of maximising the proportion of troubleshooting-related threads, we focused primarily on two main resources: Linuxquestions,[1] and a subset of the Debian mailing lists.[2] From those we further selected three specific subforums intended to include a representative range of both general-purpose and hardware/package-specific forums, namely Linuxquestions-software, Debian-amd64, and Debian-apache as our data sources.

We crawled the data from these forums using dedicated software tailored to the VBulletin[3] forum manager (for Linuxquestions) and the pipermail web log (for the Debian mailing lists). In order to extract the relevant information from the documents we analysed the table structure of the VBulletin forums and the header structure of the pipermail web logs. We required specific rules to identify quotations in the posts, which were represented with special codes as formatted text in XML. This software has also been used to crawl data from other resources (e.g. FEDORAFORUM).

Each of the forums was then stored in a MySQL relational database. We designed a common representation for the discussion lists, based on forums, threads, and posts:

Forum: the source forum, namely Linuxquestions-software, Debian-amd64 or Debian-apache.

Thread: each thread is linked to a given forum, and stored with a title, author, date, and flag (e.g. *sticky post*).

Post: each post is related to a thread and forum. We also register its title, position in the thread, date, which post it follows (for nested threads), raw text, and formatted text.

We filtered out threads containing only one post and those containing more than 14 answers to the original post. Obviously threads with no answers cannot provide problem solutions, and long threads tend to be more discussion related (e.g. discussing the relative merits of different distributions or packages) and/or contain too much information for accurate processing. For Linuxquestions we also removed those threads flagged as "sticky" because they usually contain polls and discussions.

After filtering, our data collection contains more than 380,000 posts spanning 90,000 threads. The detailed statistics of the crawled data from each of the three forums are given in Table 1. We can see that most of the data comes from the Linuxquestions forum,

[1] http://www.linuxquestions.org

[2] http://lists.debian.org/completeindex.html

[3] http://www.vbulletin.com

	Linuxquestions	Debian-apache	Debian-amd64	Overall
Years	2000-2007	2001-2006	2003-2006	
Threads	80,814	5,251	4,364	90,429
Posts	352,677	8,063	19,670	380,410
Users	42,622	3,484	2,305	48,411
Average posts per thread	4.36	1.54	4.51	4.21
Average posts per user	8.27	2.31	8.53	7.86

Table 1: Details of the forum data

which is the one with the most users. Also, we can notice a clear difference in the length of threads for the two Debian subforums.

4 Task descriptions

The first step in extracting information from the threads is to identify which threads are more relevant in the context of troubleshooting. There are different characteristics of a thread that can make it less useful, including: it may not refer to a specific problem; the thread may be unsolved; or the description of the problem may be unclear. For this work we detected 5 characteristics that are important in profiling the predicted troubleshooting utility of a given thread:

Task orientation: is the thread focused on solving a specific problem or devoted to a more general discussion on some topic?

Completeness: does the initial post include a sufficiently detailed specification of the problem for a third party to be able to realistically provide a solution?

Solvedness: is there a documented solution to the original problem described by the thread initiator in the thread (including the possibility of URLs pointing off to solutions elsewhere on that same forum or generally on the web)?

Spam: is the thread spam?

Problem type: free text keyword description of the type of problem described (e.g. *software installation*).

We hand-annotated a subset of the thread collection in order to get a better understanding of the data and train thread-level classifiers over. 250 threads were annotated independently by three annotators, in terms of the 5 thread-level features described above. The threads were chosen randomly from across the three forums. For the "Problem type", we allowed the annotators to either choose from a pull-down list of tags from previously-annotated threads, or alternatively to enter a free-text description for the current thread. This information is not used in the experiments described in this paper. Similarly, our simple filtering of threads based on the number of posts removed all spam from our 250 thread sample, such that the "Spam" task is also not a relevant task for the current dataset.

The annotators were asked to rate the "Task orientation", "Completeness" and "Solvedness" of each thread based on a five point scale, with a score of 1 indicating a high degree of fit with the given designation, and a score of 5 indicating a low degree of fit. The mean numeric value across the 3 annotators was used to derive the gold-standard value for each of the three tasks. For the majority of our experiments, we convert the numeric score into a binary value, using 2.5 as our breakpoint. The annotation process took between 5.7 and 8.7 hours depending on the annotator.

We measured the inter-tagger agreement for the 3 different tasks using the Spearman rank correlation and the Kappa statistic. Rank correlation was calculated between each annotator and the mean value, and for the Kappa statistic we converted the 5-way scores into 3-way values (positive, neuter and negative) and computed the agreement between the 3 annotators. The reason for calculating the rank correlation in addition to the Kappa statistic is that Kappa is incapable of capturing the fact that the rating scale is ordinal, and treats a 4 vs. 5 rating difference equally to a 1 vs. 5 rating difference, where there is clearly higher agreement in the first instance; rank correlation is better equipped to model this difference, by viewing the rating according to its relative position in the ranking rather than the raw numbers.

Table 2 shows the inter-tagger agreement for the three tasks.

We can see that the Kappa values are low, indicating that there is low agreement relative to the label bias of the task, and that the classifiers may have problems with the three-way classification task. However, the high rank correlation shows that, while the individual annotators may have interpreted the raw numbers on the 5-point scale differently, there is general consistency in terms of what they considered to be more or less task oriented, complete or solved. We approach the task in both classification and regression terms in order to explore both the "direct hit" discrete view of the data as performed by the annotators, and the ordinal view of the data represented in the ranking (mapping the real-valued outputs of the regression model onto a ranking).

Test	Task Orientation	Completeness	Solvedness
Rank Correlation A1/mean	0.94	0.86	0.71
Rank Correlation A2/mean	0.94	0.81	0.82
Rank Correlation A3/mean	0.90	0.85	0.79
3-way kappa	0.64	0.21	0.38

Table 2: Agreement statistics: rank correlation between each annotator and the mean, and the 3-way Kappa statistic

5 Features

In order to represent the threads for our classifiers and regression models, we defined two common sets of features to use for all three tasks. As the baseline feature set we constructed simple bag of words (BoW) features, consisting of all the words occurring in each of the threads. The second feature set (THREAD) relies on the thread structure and specific contextual features for the representation. We identified four subparts of the thread that appear to contain different types of information, and are expected to impact differently on each of three thread classification tasks:

Initial Post (INITIALPOST): the first post of the thread, which contains the original problem description; we also include the immediately proceeding posts (based on the chronological order of the posts) if they were authored by the same user, as they usually constitute clarifications or supplementary information. This part of the thread is expected to be particularly relevant to both the "Task Orientation" and "Completeness" tasks.

First Response (FIRSTRESPONSE): the first post in the thread belonging to a user other than the initiator; expected to be relevant for all three tasks.

All Responses (ALLRESPONSES): the remaining posts in the thread after removing the two sets above; expected to be relevant for the "Task Orientation" and "Solvedness" tasks.

Final Post from the Initiator (FINALPOSTINIT): the final (sequence of) post(s) from the thread initiator after the initial post, i.e. in response to a post from someone else; expected to be particularly relevant for the "Solvedness" task.

From each of these sets we extracted a group of 18 lexical and contextual features, as defined in Table 3. Below, we denote each of these 4 threads subparts with the indicated acronyms (i.e. INITIALPOST, FIRSTRESPONSE, ALLRESPONSES and FINALPOSTINIT), and the union of these groups as THREAD.

Apart from the intra-thread features, there are other forum-level features that are indicative of the rating of a given thread according to the three classification tasks, such as the particular users that initiate/respond to the threads, or the number of external links pointing to the target thread. We plan to analyse these feature types in the future.

6 Machine Learners

In our experiments, we used three different machine learning software packages to build our classifiers and regression models:

SVMLIB [1]: an integrated toolkit for support vector machine classification, regression, and distribution estimation (one-class SVM). For our experiments we used the classification and regression modules, relying on an RBF kernel in each case.

TIMBL [3]: a fast implementation of a number of variants of the k-nearest neighbour algorithm, out of which we use an extended version of IB1.

WEKA [7]: a suite of machine learning and regression algorithms for data mining tasks. From the suite of algorithms, in our experiments we applied the following classification algorithms: JRIP, J4.8 decision trees, support vector machine with a linear kernel (SVM), naive Bayes (NB), ADABOOSTM1, BAGGING and STACKING; and the following regression algorithms: linear regression (LM) and PERCEPTRON.

7 Evaluation

In this section, we describe our experiments on the three different thread classification tasks. First we present our experimental setting, and then we analyse the results of the different feature sets for two classifiers. Next, we report on the classification accuracy of the different models in binary classification terms, and also rank correlation for the regression models. Finally, we present learning curves to shed light on the relationship between the amount of training data and our relative results.

7.1 Experimental setting

For all our experiments on the 250 thread dataset, we performed stratified 10-fold cross-validation. For the main classification tasks we binarised the 1-5 values given by the three annotators into positive and negative instances relying on the mean of the scores. We decided to remove the few cases where the mean of the three annotators' ratings was exactly at the midpoint of the 5-point scale (i.e. 3). This left us with 244 threads for the

Feature Description	Feature Type
Number of posts	Integer
Number of Linux distribution mentions (e.g. *redhat*)	Integer
Number of beginner keywords mentioned (e.g. *noob*)	Integer
Number of emoticons detected (e.g. *:-)*)	Integer
Number of version numbers mentioned (e.g. *version 5.1*)	Integer
Number of URLs detected (e.g. *www.ubuntu.org*)	Integer
Proportion of words relative to full thread	Real
Proportion of sentences relative to full thread	Real
Proportion of question sentences in set	Real
Proportion of exclamation sentences in set	Real
Proportion of simple declarative sentences in set	Real
Proportion of code sentences in set	Real
Proportion of other sentences in set	Real
Average sentence length	Real
Average word length	Real
Proportion of sentences to first question	Real
Proportion of thread posts to first class post	Real
Proportion of thread posts to last class post	Real

Table 3: Lexical and contextual features extracted for each of the 4 thread subparts

"Task Orientation" task, 232 for the "Completeness" task, and 222 for the "Solvedness" task. For the rank-correlation evaluation we used all 250 threads for each task.

We measured accuracy values for the classification experiments. As our baseline we used a majority class (MC) classifier that assigns the majority class in the training data to all test instances.

7.2 Feature sets

For our first experiment we analysed the performance of binary classifiers over the different feature sets. This gives us an indication of whether the THREAD features provide an enhancement over the classic BOW model. We further contrast this with the combination of the two feature sets (All) to determine whether they are complementary. At this stage of our experiments, we focused exclusively on two learners: TIMBL and SVMLIB.

The results for TIMBL are shown in Table 4. We can see that the contribution of the feature sets differs across the three tasks, but that the performance is lower than our MC baseline in most cases. The most significant drop occurs for BOW in the "Solvedness" task, while the other feature sets are more stable across different tasks.

The results for SVMLIB are given in Table 5. Here the system performs at the same level as the MC baseline in most cases. There is a slight increase is the "Solvedness" task performance for some of the feature sets, and a small decrease for the THREAD feature set, but overall, the classifiers mirror the performance of the baseline.

Looking over the results for the three tasks across the different feature sets, we find that with TIMBL there is partial agreement with our expectations of which thread subparts would be most predictive of the three classification tasks. We also can see that particularly for the "Solvedness" task, the THREAD features provide useful information not present in the simple BOW features, or at the very least that the performance with the THREAD features is more consistent than with the BOW or All features. For the remainder of our experiments, therefore, we use the THREAD features exclusively.

7.3 Binary Classification and Rank Correlation

For our next experiment we applied the extra learners from the WEKA toolkit to the THREAD feature set to compare their performance with that of TIMBL and SVMLIB. The results of these experiments are given in Table 6. We can see that the overall results are low, and that in most cases the simple MC baseline obtains the highest result. Only for the "Solvedness" task do we achieve above-baseline results, using JRIP and STACKING.

We analysed the cause of the unexpectedly low performance of our classifiers, looking first at the binarisation of the tasks. In the annotation phase we observed low Kappa scores among the annotators, but considerably higher rank correlation. This suggests that the relative ordering of the ratings for the annotators is consistent but there is disagreement in the absolute numbers they use. Thus, the binarisation of the task could be one of the causes of the low performance of the classifiers, as the break point between classes may be hard to determine.

Another way to process the threads is to establish a ranking using regression models or classification weights. This allows us to measure the rank correlation

Features	Task Orientation	Completeness	Solvedness
MC	0.816	**0.948**	**0.765**
BoW	**0.820 (0.032)**	0.935 (0.021)	0.279 (0.171)
InitialPost	0.713 (0.072)	0.918 (0.039)	0.703 (0.077)
FirstResponse	0.746 (0.106)	0.905 (0.037)	0.685 (0.069)
AllResponses	0.734 (0.080)	0.909 (0.030)	0.734 (0.072)
FinalPostInit	0.668 (0.104)	0.879 (0.045)	0.667 (0.073)
Thread	0.762 (0.064)	0.914 (0.033)	0.653 (0.058)
All	**0.820 (0.031)**	0.935 (0.028)	0.401 (0.147)

Table 4: Accuracy for TiMBL using the different feature sets (standard deviation given in parentheses; best results are shown in **bold**)

Features	Task Orientation	Completeness	Solvedness
MC	**0.816**	**0.948**	0.765
BoW	**0.816** (0.027)	**0.948** (0.016)	0.779 (0.026)
InitialPost	**0.816** (0.018)	**0.948** (0.016)	0.779 (0.026)
FirstResponse	**0.816** (0.025)	**0.948** (0.016)	0.779 (0.036)
AllResponses	**0.816** (0.026)	**0.948** (0.016)	0.779 (0.031)
FinalPostInit	**0.816** (0.024)	**0.948** (0.016)	**0.788** (0.031)
Thread	0.807 (0.026)	**0.948** (0.016)	0.757 (0.041)
All	**0.816** (0.024)	**0.948** (0.016)	0.779 (0.031)

Table 5: Accuracy for SVMLib using the different feature sets (standard deviation given in parentheses; the best results are shown in **bold**)

between the gold standard values and the predictions. We performed these experiments by applying two regression models (linear regression and SVM) and two classifiers with weighted predictions. We tested different kernel types for SVM with small differences in performance, and report here the results for linear kernels. We measured the Spearman rank correlation between the gold standard weights and the predicted values, and present the results in Table 7.

The results are still lower than the simple MC baseline. Linear regression performs better than the other approaches, but still significantly below the baseline. Therefore the low correlation suggests that determining the boundary is not the only problem with this dataset.

7.4 Learning Curves

In this analysis we explored whether the hand-annotated data was sufficient in quantity to successfully train the classifiers. Thus, we tested the effect of using different amounts of training data, ranging from 20% to 100% of the total training set. We present the results for the SVMLib classifier and the linear regression model from Weka. The performance for SVMLib is given in Figure 1. We see that there is not a clear improvement in the results as the quantity of training data increases. This suggests that even with more data we do not expect much success from SVMLib for the different tasks.

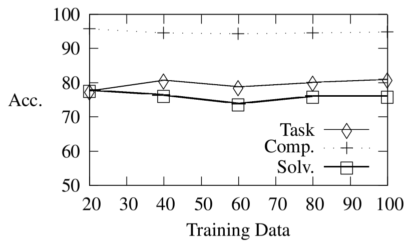


Figure 1: Learning curve for SVMLib

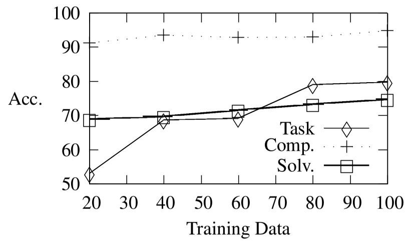


Figure 2: Learning curve for linear regression

The linear regression results are given in Figure 2. In this case the introduction of more training data increases the accuracy of the system. This indicates that the regression model could benefit from extra annotated data.

System	Task Orientation	Completeness	Solvedness
MC	**0.816**	**0.948**	0.765
TiMBL	0.762	0.914	0.653
SVMLib	0.807	**0.948**	0.757
JRip	0.807	**0.948**	0.770
J4.8	0.717	**0.948**	0.685
SVM	0.783	0.940	0.765
Perceptron	0.729	0.909	0.721
NB	0.734	0.741	0.641
LM	0.791	0.944	0.739
AdaBoostM1	0.799	0.931	0.730
Bagging	0.811	**0.948**	0.766
Stacking	**0.816**	**0.948**	**0.779**

Table 6: Accuracy for all classifiers using Thread feature-set (best result per column shown in **bold**)

System	Task Orientation	Completeness	Solvedness
MC	**0.54**	**0.53**	**0.51**
LM	0.14	0.40	0.37
SVM-regression	0.12	0.32	0.25
SVM-classification	0.18	0.10	0.18
JRip	0.08	0.22	0.04

Table 7: Spearman rank correlation between goldstandard and different models (best result per column shown in **bold**)

8 Discussion

The empirical results showed the problems in both the classification and regression frameworks. We observed that our annotation scheme produced low Kappa agreement, and this reflected on the performance of the classifiers, causing them to perform at or below the baseline. We had higher hopes in the regression experiments, because the rank correlation between annotators was higher, but the regression models did not show much improvement.

In order to test the classifiers in the same setting as the original annotation, we also explored the results of classifiers on 5-way classification. We observed that in this case the performance was clearly above the MC baseline for the "Completeness" and "Solvedness" tasks. This classification would be potentially useful to help rank threads to present to the user. This 5-way annotation task highlighted the problem of the boundary (i.e. mid-range) cases, which were difficult to detect for the classifier and contributed most of the errors.

Another problem that was not addressed in the experiments was the relationship between the different tasks. For instance, if a thread is considered discussion oriented (in terms of the "Task Orientation" task), the annotation of "Solvedness" should be different from a specific-task oriented thread. We performed an experiment to measure the results using different subsets of data from the gold standard annotation (e.g. measure "Solvedness" and "Completeness" in task-oriented threads only). The results show only slight increases over the MC baseline, suggesting that partitioning the data in a hierarchical approach would not completely solve the problem.

All in all the experiments show the complications in automatising these tasks. The results indicate that automatic methods can be developed to identify the most clear-cut cases and provide rankings of threads, but for most cases the results are below the MC baseline. It is worth mentioning that the baseline is very high (94.8%) in the case of "Completeness", and this makes it difficult to improve over. The majority class is also high for the other two tasks (77.9% and 81.6%).

9 Conclusion

In this work we present the ILIAD project for information delivery in the Linux domain, and the first exploration of utility-based classification of threads from different sources. For our main goal of providing better information-access tools, we developed software to crawl and normalise data from different mailing lists and forums. We used these tools to create a collection of more than 90,000 threads addressing different problems.

After analysis of a sample of threads, we identified some relevant characteristics that could be useful to deliver the information. We defined the "Task orientation", "Completeness", and "Solvedness" as the object of study for this paper, and annotated a subset of threads on their having these characteristics. We then developed automatic methods to classify threads according to these dimensions.

Our experiments highlighted some of the problems to carry out these tasks automatically. The low Kappa agreement between annotators lead to difficulties to train systems for binary classification. However, there was higher rank correlation, and this suggests that we can develop systems to rank threads according to their characteristics.

We are currently working on extending our feature set to include other types of features, such as the authors of the posts and the external links pointing to target threads. We are also applying and evaluating these techniques in an IR framework. Our aim in this task is to measure the contribution of devoted tools over general keyword-based search. For future work we plan to develop tools to process the content of the threads, and extract the relevant information for delivery.

Acknowledgements This research was carried out with support from Australian Research Council grant no. DP0663879.

References

[1] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[2] Hang Cui, Ji-Rong Wen, Jian-Yun Nie and Wei-Ying Ma. Probabilistic query expansion using query logs. In *Proceedings of the 11th International Conference on the World Wide Web*, pages 325–332, Honolulu, USA, 2002.

[3] Walter Daelemans, Jakub Zavrel, Ko van der Sloot and Antal van den Bosch. Timbl: Tilburg memory based learner, version 5.1, reference guide. Technical report, ILK University of Tilburg, 2004.

[4] Edward Ivanovic. Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop*, pages 79–84, Ann Arbor, USA, 2005.

[5] Jihie Kim, Grace Chern, Donghui Feng, Erin Shaw and Eduard Hovy. Mining and assessing discussions on the web through speech act analysis. In *Proceedings of the ISWC'06 Workshop on Web Content Mining with Human Language Technologies*, Athens, USA, 2006.

[6] Markus Weimer, Iryna Gurevych and Max Mühlhäuser. Automatically assessing the post quality in online discussions on software. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 125–128, Prague, Czech Republic, 2007.

[7] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, San Francisco, USA, 2005.

[8] Jinxi Xu and W. Bruce Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, Zurich, Switzerland, 1996.

[9] Liang Zhou and Eduard Hovy. Digesting virtual geek culture: The summarization of technical internet relay chat. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 298–305, Ann Arbor, USA, 2005.

Source Code Authorship Attribution using *n*-grams

Steven Burrows *S.M.M. Tahaghoghi*

School of Computer Science and Information Technology
RMIT University
GPO Box 2476V, Melbourne 3001, Australia

{*steven.burrows,seyed.tahaghoghi*}*@rmit.edu.au*

Abstract

Plagiarism and copyright infringement are major problems in academic and corporate environments. Existing solutions for detecting infringements in structured text such as source code are restricted to textual similarity comparisons of two pieces of work. In this paper, we examine authorship attribution as a means for tackling plagiarism detection. Given several samples of work from several authors, we attempt to correctly identify the author of work presented as a query. On a collection of 1 640 documents written by 100 authors, we show that we can attribute authorship in up to 67% of cases. This work can be a valuable additional indicator for the more difficult plagiarism investigations.

Keywords: Authorship Attribution, Plagiarism Detection, Co-derivative Documents

1 Introduction

Educational institutions and industry frequently encounter plagiarism and copyright infringement. Plagiarism is the presentation by one person of the ideas or work of another person as their own; this presentation may be in the form of plain text — as present in essays and reports — or structured text, such as equations and computer programming languages. Copyright provides exclusive publishing rights to authors to protect their ideas and information. Authors may licence their copyrighted works for a fee, but unauthorised reproductions by other parties are considered copyright infringement.

Searching for plagiarism or copyright infringement by human inspection is time consuming and not practical for large collections of work. Plagiarism detection tools, such as Turnitin [16] for plain text, are often used to detect infringed work. However, approaches for detecting plagiarism in plain text are not suitable for detecting plagiarism in structured text, as they ignore important aspects of structured text documents such as programming syntax in source code. For example, changing literal names or rewriting comments does not affect the function of a program.

Source code plagiarism detection tools for detecting anomalies within a set of files — such as assignment submissions by a student cohort — are available [4, 8, 11, 17, 26, 33]. However, these do not consider historical data on each author, and cannot detect whether work has been plagiarised from a source outside the set.

Authorship attribution aims to detect plagiarism by establishing a profile of an author's style from several sample documents. Work diverging significantly from the profile can be flagged for investigation. If the work fits the profile of a different known author, then the case for plagiarism is strengthened.

Authorship attribution has been studied for plain text [36], but has not been explored for structured text such as source code. We investigate several techniques for attributing authorship of source code, building on techniques used in text information retrieval. On a collection of student assignments containing 1 640 documents from 100 authors, the best of our techniques correctly attributes authorship 67% of the time.

The remainder of this paper is organised as follows. In the next section we report on the magnitude of unauthorised code reuse. In Section 3, we review existing approaches to source code similarity detection and plain-text authorship attribution. In Section 4 we describe our approach to the problem, and introduce the data we use to evaluate our solutions. We present results of our experiments and discuss our findings in Section 5, and give conclusions in Section 6.

2 Motivation

Unauthorised reuse of code is a major problem in both academia and industry. Marsden [25] describe the results of a comprehensive study of 954 students from four Australian universities where 81% admitted to having engaged in some form of plagiarism. Moreover, Alemozafar [1] describes the increasing trend of the problem at Stanford University, where the Office of Judicial Affairs witnessed violations of the honour code increase by 126% between 1998 and 2001.

In interviews with academic staff, Bull et al. [5] found that 50% of the 321 respondents agreed that there "has been an increase of plagiarism in recent years";

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

only 15% disagreed and 35% didn't know. Culwin et al. [7] describe another study in which representatives from 55 higher education computing schools completed a questionnaire. They found that 89% of institutions surveyed felt that source code plagiarism was either a "minor nuisance", a "routine headache" or "bad and getting worse". Only 9% felt that it was either "not a problem" or "under control".

Several cases of plagiarism have attracted media attention or prompted special actions by academic staff. Ketchell [21] describes a case at RMIT University where staff were alerted to suspicious pin-up advertisements offering tutoring and "extra help". The investigation found many examples of work produced by the advertiser, with some solutions concurrently sold to multiple students. Zobel [38] notes that if fresh solutions were provided for each client, profiling student work is likely to be key to automated detection.

Some web sites act as software development marketplaces by allowing users to post work specifications and invite competitive bids from independent contractors. D'Souza et al. [9] report two cases where students had finalised deals on the RentACoder web site[1] for assignment solutions valued at US$200 and AU$35; these prices are affordable to many students. To demonstrate the severity of this problem, D'Souza et al. presented a list of over twenty such web sites that function as software development marketplaces.

There is a need for whole organisations to protect themselves against plagiarism and copyright infringement. For example, Edward Waters College in Jacksonville, Florida had its accreditation revoked in 2004 after plagiarised content was found in documentation sent to its accreditation agency [3]. Accreditation was regained six months later after legal proceedings [23]. This incident resulted in reduced enrolments and threatened current students with loss of funding.

Unauthorised code reuse is also a concern in the corporate sector. For example, SCO Group sued IBM for more than one billion dollars for allegedly incorporating Unix code in its Unix-like AIX operating system in March 2003 [27]. Similarly, the Software Freedom Law Center, which investigates possible violations of software licences such as the GNU General Public License, has identified the need for automated approaches to determine whether one project is the derivative of another [22].

3 Background

Tools to detect authorship issues in source code identify matches using metric-based or structure-based approaches [32]. Jones [17] describes an unnamed metric-based system with a hybrid of physical metrics (line, word, and character counts) and Halstead metrics [13] (token occurrences, unique tokens, and Halstead volume) to characterise code. Jones uses the Euclidean distance measure on normalised program profiles to score program closeness. Other metric-based systems reported in the literature employ between four and twenty-four metrics [8, 10, 12]. These systems are among the first that implemented electronic plagiarism detection and many measure archaic features of source code no longer present in modern-day programming languages. Limitations exist in the work by Faidhi and Robinson [10] and Grier [12] who measure features in Pascal code as some of the features chosen are among the easiest to modify by plagiarists such as comments and indentation. Donaldson et al. [8] present a plagiarism detection system for Fortran code, however the summation metrics used to compare programs are sensitive to program length.

Prechelt et al. [26] describe the JPlag structure-based system that works in two phases. First, program source code files are parsed and converted into token streams. Second, the token streams are compared in an exhaustive pairwise fashion using the Greedy String Tiling algorithm as used in the YAP plagiarism detection system [33]. Collections of maximally overlapping token streams above a threshold length are stored and given a similarity score. Program pairs with a similarity score above a threshold percentage are made available to the user for manual side-by-side inspection. Systems such as this are unsuited to verify authorship as they aim to identify functionally equivalent — rather than stylistically consistent — code.

In previous work [6] we describe a scalable code similarity solution that uses the Zettair search engine[2] to index n-grams [34] of tokens extracted from the parsed program source code. At search time, the index is queried using the n-gram representations of each program to identify candidate results. These are then post-filtered using the local alignment approximate string matching technique [30].

Plain text infringement detection tools [5, 28] have been used on academic and corporate text-based work such as essays and reports. Hoad and Zobel [14] describe two methods for detecting co-derivative text documents. *Ranking* involves presenting the user with a list of candidate answers sorted by similarity to a query; this approach is commonly used by search engines to retrieve multiple answers to a query where there is not necessarily a single correct answer. *Fingerprinting* computes compact descriptions of documents using a hash function. It is critical that the generated integers are as uniformly distributed as possible to reduce the number of duplicates — this is a property of any good hash function. Other issues for consideration are the substring length used for the hash function (fingerprint granularity) and the number of fingerprints used to represent a document (fingerprint resolution). Manber [24] describe the *sif* plagiarism detection

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

[2] http://www.seg.rmit.edu.au/zettair

system that uses fingerprinting. Fingerprinting is a lossy approach, so we use ranking in our work to retain all stylistic markers.

Authorship attribution techniques for plain text apply text retrieval, statistical, or machine-learning approaches on a set of document features to characterise an author's *style* — Zhao et al. [36, 37] compare many of these. Zhao and Zobel [36] describe three classes of authorship attribution. In one-class authorship attribution, a pool of documents is gathered where some are known to belong to a single author and others are of unknown authorship; the task is to determine whether a new document belongs to the known author. In binary authorship attribution, a collection contains documents belonging to one of two authors, and we must determine to which author a new document belongs. Binary authorship attribution is a special case of multi-class authorship attribution, where more than two authors are involved. We focus on the multi-class attribution problem for source code as it best similates a real-life scenario involving authorship verification of work samples from large classes.

Turnitin[3] and the related iThenticate[4] system are well-known text plagiarism detection tools. These compare submitted work to documents from the Web and other repositories [15, 16]. However, details of the system implementation of Turnitin are difficult to obtain due to commercial considerations.

4 Methodology

We use an information retrieval model for attributing authorship; to determine the author of a document, we use it as a search engine query — the *query document*, written by the *query author* — on a corpus with known authors; our approach produces a list of all documents in the corpus ranked by estimated similarity to the query document. Ideally, every other corpus document by the query author would appear at the top of the ranked list, followed by all other documents. We now detail each component of this process.

4.1 Document collection construction

Our collection contains 1 640 assignment submissions from the 100 most prolific authors of C programming assignment submissions submitted to our school over the period 1999 to 2006. We removed byte-identical submissions from the same authors, and removed all personal details from the source code in adherence to ethics guidelines.

We expect our collection to have some bias towards weaker students. For example, our collection has many submissions from our first semester C programming course — Programming Techniques. We found that approximately 12% of submissions over nine consecutive offerings of Programming Techniques were from repeat students who submitted to more than one offering. However we expect some of the strongest students who pursue further coursework programs to also submit more assignments than average.

Each author in our collection is represented by at least fourteen documents; the number of documents per author is shown in Figure 1a. The collection contains source code of varying complexity and size. The distribution of submission lengths is shown in Figure 1b, with the exception of twenty-nine large outliers (less than 2% of the total number of files in the collection) that we omit from the graph for brevity. Twenty-six of these submissions were between 5 000 and 9 000 tokens, with the remaining cases having lengths of 10 951, 14 998 and 22 598 tokens respectively.

While some documents in this corpus may be plagiarised from other sources, we consider our results to be valid; much research in information retrieval and authorship attribution relies on imperfect ground truth. For example, Amitay et al. [2] describe collections obtained from online sources where manual verification of data was infeasible, but where some authors may have created work under multiple aliases.

Recent plain-text authorship attribution work has demonstrated that "function words" — such as "and", "or", and "but" — are strong markers of individual style [37]. In keeping with this approach, we discard all comments and literals from the program source code in our collection, and retain only operators and keywords; these are listed in Table 1 [20]. Note that the reduction was done at a lexical level, rather than with reference to the C language grammar, and so overloaded tokens are not treated differently in this work. For example, all ampersands are treated as bitwise-and operators, even if they were used to generate a pointer from a variable.

4.2 Evaluation

To preserve some local information on token sequences, we extract windows of n tokens overlapping by one token; a program of t tokens is represented by $t - n + 1$ n-grams. We ignore submissions smaller than n. This approach is similar to that of our previous work [6]. To determine the best value of n, we tested average attribution effectiveness over 100 runs, with each run using random samples of work from our collection.

To determine the best choice of document similarity measure, we evaluate four common text similarity measures built into the Zettair search engine that we use: Okapi BM25 [18, 19], Cosine [34], Pivoted Cosine with a pivot value of 0.2 [29], and language modeling with Dirichlet smoothing with μ set to 2000 [35]. We also propose a fifth metric that we call Author1 designed specifically for source code authorship attribution. This measure is based upon the idea that the term frequency in a query and a document should be similar when the query is a document. We define this measure as:

[3] `http://www.turnitin.com`

[4] `http://www.ithenticate.com`

Operators			
(	parenthesis	!=	not equals
[	bracket	&	bitwise and
->	i. member access	^	bitwise xor
.	member access	|	bitwise or
++	increment	&&	boolean and
--	decrement	||	boolean or
!	not	?	conditional
~	complement	=	equals
*	multiply	+=	plus equals
/	divide	-=	minus equals
%	modulo	*=	multiply equals
+	plus	/=	divide equals
-	minus	%=	modulo equals
<<	left shift	<<=	left shift equals
>>	rightshift	>>=	right shift equals
<	less than	&=	bitwise and equals
>	greater than	^=	bitwise xor equals
<=	less than equals	|=	bitwise or equals
>=	great. than equals	,	comma
==	equality		

Keywords				
auto	do	goto	signed	unsigned
break	double	if	sizeof	void
case	else	int	static	volatile
char	enum	long	struct	while
const	extern	register	switch	
continue	float	return	typedef	
default	for	short	union	

Table 1: Operators and keywords preserved in documents in our collection [20].

$$\text{Author1}(Q, D_d) = \sum_{t \in Q \cup D_d} \frac{1}{min(|f_{q,t} - f_{d,t}|, 0.5)}$$

where t is a term, Q is a query, D_d is a document in collection D, $f_{q,t}$ is the query-term frequency, and $f_{d,t}$ is the document-term frequency.

Our choices for a baseline comparison system were limited — there is much previous work on source code plagiarism detection, but none on source code authorship attribution. We chose the most recent published metric-based plagiarism detection approach — that of Jones [17] — as our baseline. We believe that structure-based plagiarism detection systems such as JPlag [26] are less suitable for this task as they match contiguous plagiarised chunks of source code best, but this remains to be proven in future work.

We measure authorship attribution effectiveness using the reciprocal rank of the first correct author match for each query. We also measure authorship attribution effectiveness of whole result sets using average precision. Approaches are compared using the mean reciprocal rank percentage (MRR%) and mean average precision percentage (MAP%) over random samples of documents in our collection, where each document is treated in turn as the query document.

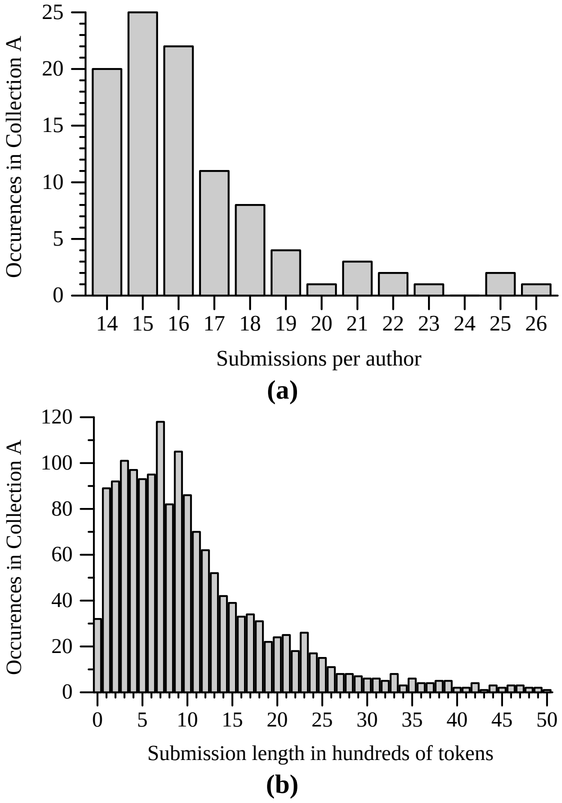


Figure 1: **(a)** Number of submissions for each author in COLLECTION A. **(b)** Distribution of submission lengths in COLLECTION A in hundreds of tokens, rounded down to the nearest 100 tokens.

5 Results

In this section, we first discuss results describing the most appropriate n-gram size for our problem domain. Second, we explore the degradation of reciprocal rank and average precision scores as the number of authors is increased and the number of submissions per author is decreased. Finally, we present our results in comparison to a baseline system before offering discussion.

5.1 *n*-gram results

Table 2 illustrates the performance of the five similarity measures using sixteen different values of n. For each similarity measure and n-gram size combination, we use all work samples from ten authors selected at random from COLLECTION A and calculate the mean reciprocal rank and mean average precision percentages of 100 runs. With ten authors, we would expect random behaviour to produce a mean reciprocal rank and mean average precision of around 10%.

In the same table we highlight the most effective gram size for each similarity measure. Unigrams are clearly inferior, and the best choice of n-gram size is six or eight for mean reciprocal rank and twenty for mean average precision. Effectiveness degrades for the largest n-gram sizes, as the number of large n-grams shared between program representations falls quickly. Such large n-gram sizes are suitable only for

Gram	Similarity Measure MRR%				
Size	Au1	Cos	Dir	Oka	P.Co
1	54.28	61.75	23.22	44.56	28.72
2	67.15	70.24	29.10	68.60	54.51
4	73.17	75.06	56.17	76.30	74.11
6	75.65	**76.51**	61.50	**77.77**	**76.81**
8	**76.63**	75.90	**62.59**	77.43	76.80
10	75.63	74.97	61.64	76.49	76.25
12	75.91	75.00	63.96	76.64	76.31
14	73.76	73.07	63.92	74.46	74.23
16	73.16	72.50	64.80	73.50	73.56
18	72.05	71.97	66.34	72.51	72.44
20	70.03	70.03	66.16	70.36	70.39
30	62.21	62.17	61.88	62.29	62.22
40	56.64	56.53	56.51	56.59	56.62
50	52.96	53.08	52.93	53.07	53.10
70	48.22	48.26	48.13	48.28	48.29
90	44.19	44.21	44.13	44.23	44.22
Gram	Similarity Measure MAP%				
Size	Au1	Cos	Dir	Oka	P.Co
1	18.71	21.83	12.92	19.39	14.27
2	22.94	24.95	13.95	25.53	19.91
4	26.69	27.73	21.35	28.84	26.28
6	29.25	29.52	23.41	30.60	29.33
8	32.46	32.00	25.85	33.34	32.56
10	37.56	37.25	30.36	38.55	38.07
12	44.00	43.56	36.91	44.77	44.36
14	53.34	53.11	46.19	54.12	53.77
16	58.82	58.58	52.26	59.43	59.32
18	63.25	63.28	58.43	63.90	63.74
20	**64.33**	**64.48**	**60.91**	**64.95**	**64.86**
30	60.89	60.91	60.47	61.08	60.96
40	56.08	56.11	56.16	56.16	56.15
50	52.75	52.90	52.76	52.90	52.90
70	48.10	48.18	48.09	48.19	48.17
90	44.17	44.19	44.12	44.21	44.20

Table 2: Effect of increasing the n-gram size using collections of ten authors and sixteen samples of work per author on average.

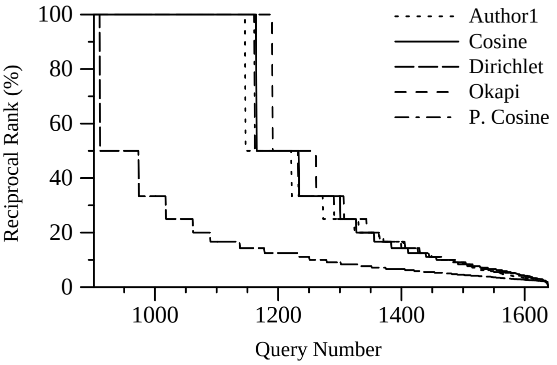


Figure 2: Reciprocal rank of five similarity measures tested using using 6-gram program representations. The combined results of ten runs are shown using collections of ten authors and sixteen samples of work per author on average. The first 900 results are omitted as these all achieved a perfect reciprocal rank.

near-duplicate cases of plagiarism which are rare — for example iParadigms [16] detail that 29% of papers received exhibit “significant plagiarism” whereas only 1% are entire copies. The remaining 70% have no plagiarism.

We conducted Tukey HSD (Honestly Significant Difference) tests on a subset of results in Table 2. Testing the 6-gram, 8-gram and 10-gram rows, we found that Okapi was statistically most effective when comparing mean reciprocal rank. Okapi with 6-grams is our best result, but this is not statistically significant from six out of fourteen other combinations of similarity measure and n-gram size tested. Concerning mean average precision, Okapi was only statistically more significant than Dirichlet and the Author1 metric when testing the 18-gram, 20-gram and 30-gram rows. Okapi with 20-grams is our most effective combination; this is statistically better than 18-grams and 30-grams, but is only statistically better than Dirichlet when considering other combinations in the same row.

Three of the five similarity measures shared the highest mean reciprocal rank score for the 6-gram program representations; this is illustrated in Figure 2, with results sorted by decreasing reciprocal rank. The best 900 query results for each measure had a perfect reciprocal rank score, and we omit these for brevity. With the exception of the Dirichlet measure, the similarity measures performed with similar effectiveness, with perfect reciprocal rank scores in at least 1 146 queries out of 1 640. The poor performance of the Dirichlet measure may possibly be addressed through empirical tuning of the μ parameter for our problem domain.

We select 6-grams as our choice of gram size for experiments described in Sections 5.2 and 5.3.

5.2 *n*-gram scalability

In Table 3 we show the degradation of mean reciprocal rank and mean average precision scores over five collection sizes for all five similarity measures evaluated. Unsurprisingly, the scores drop as the collection size increases, but we still achieve a mean reciprocal rank of 67% using the Okapi measure on our full collection. We anticipate that the scores will degrade to an unacceptable level when the collection size reaches thousands of authors — this is a real-life collection size as evidenced by our school assignment archives dating back to 1999. We intend to verify authorship in future work by comparing query documents to other documents by the same author and a subset of other authors instead of exhaustive comparison. The intention is to be satisfied that submitted works belong to the author, and not necessarily find the source of plagiarism offences.

To verify statistical significance of these results, we considered rows with ten and twenty-five submissions per author and we found that results were more conclu-

Num Auth	Similarity Measure MRR% Au1	Cos	Dir	Oka	P.Co
10	**75.65**	**76.51**	**61.50**	**77.77**	**76.81**
25	69.68	70.88	58.47	72.52	71.32
50	66.21	67.32	57.44	69.61	68.30
75	64.43	65.41	56.63	68.06	66.58
100	63.26	64.14	56.31	67.01	65.48
Num Auth	Similarity Measure MAP% Au1	Cos	Dir	Oka	P.Co
10	**29.25**	**29.52**	**23.41**	**30.60**	**29.33**
25	20.63	20.74	16.52	22.06	20.89
50	16.75	16.84	13.86	18.23	17.17
75	15.05	15.12	12.73	16.54	15.53
100	14.09	14.16	12.16	15.55	14.58

Table 3: Effect of increasing the number of authors with 6-gram program representations and collections of sixteen submissions per author on average.

Subs Auth	Similarity Measure MRR% Au1	Cos	Dir	Oka	P.Co
2	31.90	32.82	25.74	33.49	32.22
4	50.70	50.22	39.17	51.27	49.99
8	64.22	63.84	51.27	65.20	63.70
12	70.62	70.63	56.55	71.95	70.90
16	**75.65**	**76.51**	**61.50**	**77.77**	**76.81**
Subs Auth	Similarity Measure MAP% Au1	Cos	Dir	Oka	P.Co
2	**31.90**	**32.82**	**25.74**	**33.49**	**32.21**
4	31.46	31.57	24.67	32.34	31.19
8	30.04	30.32	23.89	31.38	30.10
12	29.49	29.87	23.31	30.94	29.62
16	29.25	29.52	23.41	30.60	29.33

Table 4: Effect of increasing the number of submissions per author with 6-gram program representations and collections of ten authors.

sive for this experiment. Okapi was statistically most effective for both mean reciprocal rank and mean average precision. Okapi with 10-grams was always the statistically most effective combination except for Pivoted Cosine with 10-grams when measuring mean reciprocal rank.

In Table 4 we show the effect of changing the number of submissions per author. We note that the final row of results in these tables are for sixteen submissions per author *on average*; the actual number of varies from fourteen to twenty-six. We observe a steady decline in mean reciprocal rank scores as the number of submissions per author is decreased. Given that reciprocal rank uses the single best result, having more submissions per author gives a higher chance of encountering a document by the same author with a strong style match. We note that our collection is biased towards the most prolific authors in our school, so having a sharp decline like this is undesirable. However, we observe a different trend for mean average precision — these scores are close to constant with a gradual increase as the number of submissions per author is reduced. Given that mean average precision is a measure showing non-degradation of effectiveness using a combination of evidence, we plan to design future experiments with this in mind by implementing similar ideas with voting models.

For this experiment, we tested statistical significance of the rows marked bold in Table 4 and the two nearest rows for each. Okapi is once again the statistically most effective similarity measure. When measuring mean reciprocal rank, Okapi with sixteen submissions per author is statistically better than any other combination except Pivoted Cosine with sixteen submissions per author. However when measuring mean average precision, Okapi with any number of submissions per author is only statistically better than Dirichlet when considering results in the same row.

Number of Authors	Similarity Measure MRR% Okapi BM25	Euclidean
10	**77.77**	**40.06**
25	72.52	28.41
50	69.61	22.62
75	68.06	20.35
100	67.01	18.89
Number of Authors	Similarity Measure MAP% Okapi BM25	Euclidean
10	**30.60**	**15.15**
25	22.06	7.37
50	18.23	4.52
75	16.54	3.52
100	15.55	2.97

Table 5: Comparison of our work to a baseline system [17] using the same collection properties described in Table 3.

5.3 Baseline comparison

In Table 5 we compare the effectiveness of the best of our approaches with the metric-based system described by Jones [17]. Our approach is statistically more effective than the baseline system, regardless of the similarity function used.

5.4 Discussion

Authorship attribution can be a valuable tool in verifying the integrity of documents. We have shown that the combination of n-grams with text similarity measures can lead to effective authorship attribution for C program source code.

Our results indicate that 6-grams and 8-grams perform well when high mean reciprocal rank is required, while 20-grams are the best choice when aiming for high mean average precision. This differs from results on plain text, where gram lengths of three, four, or five have been shown to be most effective [14, 31]. We believe that this is due to the limitations imposed on program code by the programming language, requiring more evidence to distinguish between authors.

Our collection contains program code from students of varying competence, and our experiments do not explicitly allow for this. We expect that results of experiments using structured text collections from more mature authors may remedy this problem. Possible resources for such experiments include program code from open source projects and supplementary examples from programming text books.

Other published work relates to plain text authorship attribution. This is a mature field, and Zhao and Zobel [36] compare the effectiveness of several strategies. However, these results cannot be assumed in the structured text domain as our n-gram experimental results have highlighted that larger n-gram sizes are needed.

6 Conclusion

In this paper, we have presented the case for detecting authorship attribution in program source code, and shown how existing automated tools are not suited to handle this problem.

We have reported experimental results that use a search engine to index and retrieve tokenised representations of C program source codes, with the aim of ranking documents authored by the query author above all other documents in the collection.

Our results show that larger n-gram sizes are needed than that of the plain text author attribution domain. We have shown that our work is clearly more effective than that of a reference baseline solution. We have also investigated scalability implications that are critical considerations before a structured text authorship attribution system is put into production.

Surprisingly, Okapi BM25 was modestly more effective than our Author1 metric custom designed for our problem domain. In future work we plan to explore the authorship attribution metric further as our results confirm that a small difference in term frequency is important.

Program source code is one example of structured text; other examples include mathematical equations, XML, LaTeX, word processing markup, and we intend to investigate authorship attribution on such data.

We intend to use collections outside the academic domain. Our university assignment collection exhibits high variance in coding quality, so we plan to secure new collections from industry for further experiments.

We also plan to more vigorously investigate our choice of features. Our work uses more features than that of metric-based plagiarism detection systems and we use these features with n-grams to look for short patterns to evaluate of author style. Other combinations of features will be investigated. We note that this work examines features at the lexical level and more work is needed to extend tokenisation to the grammatical level so that overloaded tokens are treated differently.

The experiments described in this paper are limited to 100 authors; we expect that attribution effectiveness will continue to degrade as the number of authors is increased. We conjecture that a practical system might not perform exhaustive comparisons to all other authors; but instead compare the query document to other samples of work by the same author and samples from other authors obtained randomly or determined by some form of initial filtering. We plan to explore voting schemes that use multiple indicators to determine authorship of query documents.

Acknowledgements

We thank Andrew Turpin for his valuable ideas and feedback.

References

[1] A. Alemozafar. Online software battles plagiarism at Stanford. *The Stanford Daily*, February 2003.

[2] E. Amitay, S. Yogev and E. Yom-Tov. Serial sharers: Detecting split identities of web authors. In B. Stein, M. Koppel and E. Stamatotos (editors), *Proceedings of the SIGIR'07 International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection*, pages 11–18, Amsterdam, Netherlands, July 2007. CEUR Workshop Proceedings.

[3] B. Bollag. Edward waters college loses accreditation following plagiarism scandal. *The Chronicle of Higher Education*, December 2004. URL: `http://chronicle.com/prm/daily/2004/12/2004120904n.htm` [Accessed 8 October 2007].

[4] K. Bowyer and L. Hall. Experience using MOSS to detect cheating on programming assignments. In *Proceedings of the Twenty-Ninth ASEE/IEEE Frontiers in Education Conference*, pages 18–22, San Juan, Puerto Rico, November 1999. IEEE Computer Society Press.

[5] J. Bull, C. Collins, E. Coughlin and D. Sharp. Technical review of plagiarism detection software report. Technical Report LU1 3JU, University of Luton Computer Assisted Assessment Centre, Bedfordshire, England, July 2002.

[6] S. Burrows, S. M. M. Tahaghoghi and J. Zobel. Efficient plagiarism detection for large code repositories. *Software: Practice and Experience*, Volume 37, Number 2, pages 151–175, February 2007.

[7] F. Culwin, A. MacLeod and T. Lancaster. Source code plagiarism in UK HE computing schools: Issues, attitudes and tools. Technical Report SBU-CISM-01-01, South Bank University School of Computing, Information Systems and Mathematics, September 2001.

[8] J. Donaldson, A. Lancaster and P. Sposato. A plagiarism detection system. In *Proceedings of the Twelfth SIGCSE Technical Symposium on Computer Science Education*, pages 21–25. ACM Press, 1981.

[9] D. D'Souza, M. Hamilton and M. Harris. Software development marketplaces—implications for plagiarism. In S. Mann and Simon (editors), *Proceedings of the Ninth Australasian Computing Education Conference*, pages 27–33, Ballarat, Australia, January 2007. Australian Computer Society.

[10] J. A. W. Faidhi and S. K. Robinson. An empirical approach for detecting program similarity and plagiarism within a university programming environment. *Computers and Education*, Volume 11, Number 1, pages 11–19, 1987.

[11] D. Gitchell and N. Tran. Sim: A utility for detecting similarity in computer programs. In J. Prey and R. E. Noonan (editors), *Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education*, pages 266–270, March 1999.

[12] S. Grier. A tool that detects plagiarism in Pascal programs. In *Proceedings of the Twelfth SIGCSE Technical Symposium on Computer Science Education*, pages 15–20. ACM Press, 1981.

[13] M. H. Halstead. Natural laws controlling algorithm structure? *ACM SIGPLAN Notices*, Volume 7, Number 2, pages 19–26, February 1972.

[14] T. Hoad and J. Zobel. Methods for identifying versioned and plagiarised documents. *Journal of the American Society for Information Science and Technology*, Volume 54, Number 3, pages 203–215, February 2002.

[15] iParadigms. iThenticate questions and answers. *iThenticate Web Site*, October 2007. URL: `http://www.ithenticate.com/static/training.html` [Accessed 4 October 2007].

[16] iParadigms. Turnitin plagiarism prevention. *Turnitin Web Site*, October 2007. URL: `http://www.turnitin.com/static/plagiarism.html` [Accessed 11 November 2007].

[17] E. Jones. Metrics based plagiarism monitoring. In *Proceedings of the Sixth Annual CCSC Northeastern Conference on The Journal of Computing in Small Colleges*, pages 253–261, Middlebury, Vermont, April 2001. Consortium for Computing Sciences in Colleges.

[18] K. Sparck Jones, S. Walker and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiments part 1. *Information Processing and Management*, Volume 36, Number 6, pages 779–808, November 2000.

[19] K. Sparck Jones, S. Walker and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiments part 2. *Information Processing and Management*, Volume 36, Number 6, pages 809–840, November 2000.

[20] A. Kelly and I. Pohl. *A Book on C*. Addison Wesley Longman, Reading, Massachusetts, fourth edition, December 1997. pp. 77, 705.

[21] M. Ketchell. The third degree. *The Age*, May 2003. URL: `http://www.theage.com.au/articles/2003/05/26/1053801319696.html` [Accessed 4 October 2007].

[22] B. M. Kuhn. Personal communication, 2007.

[23] D. Lederman. Edward waters college regains accreditation. *Inside Higher Ed*, June 2005. URL: `http://www.insidehighered.com/news/2005/06/24/waters` [Accessed 8 October 2007].

[24] U. Manber. Finding similar files in a large file system. In *Proceedings of the USENIX Winter 1994 Technical Conference*, pages 1–10, San Fransisco, California, January 1994.

[25] H. Marsden. Who cheats at university? A self-report study of dishonest academic behaviours in a sample of Australian university students. *Australian Journal of Psychology*, Volume 57, Number 1, pages 1–10, 2005.

[26] L. Prechelt, G. Malpohl and M. Philippsen. Finding plagiarisms among a set of programs with JPlag. *Journal of Universal Computer Science*, Volume 8, Number 11, pages 1016–1038, November 2002.

[27] S. Shankland. SCO sues Big Blue over Unix, Linux. *CNET News.com*, March 2003. URL: `http://news.com.com/2100-1016-991464.html` [Accessed 4 October 2007].

[28] A. Si, H. V. Leong and R. W. H. Lau. CHECK: A document plagiarism detection system. In *Proceedings of the 1997 ACM Symposium on Applied Computing*, pages 70–77, San Jose, California, February 1997. ACM Press.

[29] A. Singhal, C. Buckley and M. Mitra. Pivoted document length normalization. In *Proceedings of the Nineteenth annual international ACM SIGIR conference on Research and development in information retrieval*, pages 21–29, Zurich, Switzerland, August 1996. ACM Press.

[30] T. Smith and M. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, Volume 147, Number 1, pages 195–197, March 1981.

[31] B. Stein, S. Meyer zu Eissen and M. Potthast. Strategies for retrieving plagiarized documents. In *Proceedings of the Thirtieth annual international ACM SIGIR conference on Research and development in information retrieval*, pages 825–826, Amsterdam, Netherlands, July 2007. ACM Press.

[32] K. Verco and M. Wise. Software for detecting suspected plagiarism: Comparing structure and attribute-counting systems. In J. Rosenberg (editor), *Proceedings of the First Australian Conference on Computer Science Education*, pages 81–88, Sydney, Australia, July 1996.

[33] M. Wise. YAP3: Improved detection of similarities in computer program and other texts. In J. Impagliazzo, E. S. Adams and K. J. Klee (editors), *Proceedings of the Twenty-Seventh SIGCSE Technical Symposium on Computer Science Education*, pages 130–134, Philadelphia, Pennsylvania, February 1996. SIGCSE: ACM Special Interest Group on Computer Science Education, ACM Press.

[34] I. Witten, A. Moffat and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, second edition, 1999.

[35] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, Volume 22, Number 2, pages 179–214, April 2004.

[36] Y. Zhao and J. Zobel. Effective and scalable authorship attribution using function words. In G. G. Lee, A. Yamada, H. Meng and S. H. Myaeng (editors), *Proceedings of the Second AIRS Asian Information Retrieval Symposium*, pages 174–189, Jeju Island, Korea, October 2005. Springer.

[37] Y. Zhao, J. Zobel and P. Vines. Using relative entropy for authorship attribution. In H. T. Ng, M.-K. Leong, M.-Y. Kan and D. Ji (editors), *Proceedings of the Third AIRS Asian Information Retrieval Symposium*, Volume 4182 of *Lecture Notes in Computer Science*, pages 92–105, Singapore, Singapore, October 2006. Springer.

[38] J. Zobel. "Uni cheats racket": A case study in plagiarism investigation. In R. Lister and A. Young (editors), *Proceedings of the Sixth Australasian Computing Education Conference*, Volume 30 of *Conferences in Research and Practice in Information Technology (CRPIT)*, pages 357–365, Dunedin, New Zealand, January 2004. Australian Computer Society.

A Framework for Measuring the Impact of Web Spam

Timothy Jones
Department of Computer Science
The Australian National University
Canberra, Australia
tim.jones@anu.edu.au

David Hawking
CSIRO ICT Centre
Canberra, Australia
David.Hawking@acm.org

Ramesh Sankaranarayana
Department of Computer Science
The Australian National University
Canberra, Australia
ramesh@cs.anu.edu.au

Abstract *Web spam potentially causes three deleterious effects: unnecessary work for crawlers and search engines; diversion of traffic away from legitimate businesses; and annoyance to search engine users through poorer results.*

Past research on web spam has focused on spamming techniques, spam suppression techniques, and methods for classifying web content as spam or non-spam.

Here we focus on the deterioration of search result quality caused by the presence of spam in a country-scale web. We present a framework for measuring the degradation in quality of search results caused by the presence of web spam. We index the 80 million page UK2006 web spam collection on one machine. We trial the proposed framework in an experiment with the UK2006 collection and demonstrate that simple removal of spam pages from result sets can increase result quality. We conclude that the framework is a reasonable vehicle for research in this area and outline changes necessary for planned future experiments.

Keywords Web Information Retrieval, Web Spam, Adversarial Information Retrieval

1 The web spam problem

Web search engines are the first port of call for many users of the World Wide Web. This creates a strong commercial pressure to achieve a high web search rank. Many site operators strive to improve the layout and content of their sites using Search Engine Optimisation (SEO). However, some operators attempt to fool the ranking algorithms used by web search engines, using techniques commonly referred to as *black hat SEO* or *Web spam*. Web spam is a problem because it negatively affects the quality of the result list.

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007.

Web spam is usually defined as "any deliberate action that is meant to trigger an unjustifiably favorable relevance or importance for some web page, considering the page's true value" [5]. It is effective because few users browse past the first page of results [7]. It would only take a handful of spammed target pages to completely change the result list seen by users. Furthermore, the likelihood of a person clicking on a result is reduced if the result is moved down by even one rank, irrespective of result quality [7]. Hence, moving a result up by one rank can bring many more visitors.

Literature on the web spam problem has primarily focused on the classification of spam web pages. By combining these classification techniques, spam pages can be detected with a high degree of precision, usually around 80% [1, 9]. However, it is unclear how much effect this web spam has on the quality of results. There are a few important questions currently unanswered:

1. How does webspam affect result quality?
2. Are particular types of spam more damaging than others?
3. What is the best thing do with this spam once it is detected?

2 Simulating a UK search engine using the UK2006 collection

The UK2006 web spam collection [3] is a web snapshot crawled without any spam rejection. The collection is a snapshot of normal content and actual web spam, and provides a basis for comparison of anti-spam techniques. The collection contains around 80 million pages, with roughly 4.16 billon links to and from 11,000 hosts. 2,725 of these hosts have labels provided by human judges. Two automatic judges also contributed to the labelling. One marks controlled domains (such as gov.uk) as good, and the other marks pages in the open directory project[1] as good. These combine to give a total of 10,662 judgements, covering most of the hosts in the collection. Each label is one of {"normal", "borderline", "spam", "can not classify"}, using the guidelines[2] provided to the judges. Following the approach in [3], for our experiments we consider only hosts marked by two human judges or from controlled domains. Of these labels, 5,549 of them are "normal" and 1,924 of them are "spam".

2.1 Indexing

We indexed the 2 terabytes of the UK2006 collection on a low cost machine with 2 Intel P4 3.0GHz CPUs, 3GB of RAM, and just over 1TB of disk space. As the collection is compressed, this disk space is sufficient. The total system cost was approximately $2,000 AUD.

PADRE [6] was used for indexing and query processing. It supports searching of dynamically defined meta-collections, each comprising indexes of up to 16 primary collections. The experiments reported here used a meta-collection of four primary collections, comprising the whole UK2006 colleciton. This reduces the risk of exhausting disk space during indexing. Meta-collections accurately simulate the effect of indexing all the data as a single index, except that extra effort is required to correctly index links which cross from one primary collection to another. The total time to decompress and index the entire collection was 166.3 hours, resulting in a 129.5 GB index.

2.2 Query processing

To emulate commercial search, we used Document-At-A-Time (DAAT) [2, 8] query procesing. DAAT allows early termination of postings scans because document numbers are assigned in order of a descending query-independent *static score*. Unfortunately, due to time and hardware constraints, our document numbers were only assigned in collection order. Query response time was very reasonable. The time to generate and present 100 results for 100 queries to another machine on the network via the web interface was only 25 seconds, provided the search engine was in a "warm" state.

The quality of the search results produced is subjectively poor. For our initial experiment, we wanted to determine whether a baseline ranking can be improved simply by removing known spam items. For this it is not necessary that the baseline be of the highest quality.

The interested reader is invited to examine our baseline retrieval engine[3]. Further information about and access to the experiment can be found at `uk.wirrapoi.com`.

3 Does web spam affect result quality?

An easy treatment of web spam is to simply remove it from the result list. This is attractive because it enables the easy combination of spam detection techniques and because indexes do not have to be rebuilt. However, it does little to combat anchortext or link spam whose target is *not* a spam page. We test whether simply removing spam from result pages improves quality.

3.1 The presence of spam

Clearly, if spam pages never appear in our results, there will be nothing to remove. Consequently, we checked to see how much spam is present in typical result pages. For this. we obtained queries from the dogpile search spy[4], a tool for viewing live searches on the dogpile.com search engine. Since spam is denser around popular queries [4], we selected every query that appeared twice or more in a 72 hour period. After filtering these queries to remove searches that included domain names not present in our collection, we had 328 unique queries. The top ten results were produced for each query, and the number of spam labelled hosts was counted (Figure 1). Clearly, our ranking has been influenced by this spam, as an average 32% of these results are labelled as spam, compared with 17% in the overall collection. This demonstrates that web spam is over represented in typical result pages.

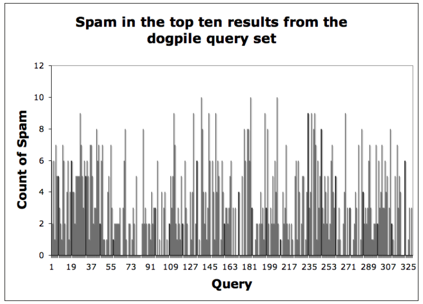


Figure 1: The amount of spam present in the top ten results for 328 queries from the dogpile search spy. Queries are sorted by their popularity, with the most popular being the far left.

3.2 Experimental setup

Volunteer subjects submitted queries of their own using our two-panel evaluation interface (see [10], shown in Figure 5). Two result pages are presented side by side, and users are invited to judge one list as being better than the other, or "no difference" between the lists. We informed users they were accessing a UK search service and suggested that, if they had difficulty thinking of queries, to imagine that they were about to travel to the UK. They were presented with two sets of unlabeled

[1] http://www.dmoz.org/

[2] `http://www.yr-bcn.es/webspam/datasets/uk2006-info/`

[3] `http://uk.wirrapoi.com/padre-sw.cgi?collection=uk&query=spam`

[4] `http://www.dogpile.com/info.dogpl/searchspy/`

search results, *standard* and *filtered*. Both derive from a single search processed as described above, which returns up to 100 results. *Standard* comprises the first 10 of these results and *filtered* comprises the first ten after pages from previously labelled[5] spam sites were removed. The left-right order of presentation of *standard* and *filtered* was randomized to avoid bias.

3.3 Results

245 preference judgements were collected from 31 users over a period of two days. These judgements covered 239 unique queries. Of these judgements, 78 were votes for the *filtered* result set, 36 were votes for the *standard* result set, and 131 were explicit votes for neither set. In a few cases, the result sets were identical because there was no labelled spam present in the top ten *standard* results. Discarding judgements on identical sets, we get 75 votes for *filtered*, 83 votes of no difference, and 35 votes for *standard*. Overall preferences for each user were also computed. This was done by scoring a vote for *filtered* as +1, no difference as 0, and *standard* as −1, then summing these scores. A user's preference will be *standard*, no difference, or *filtered* if their sum is less than, equal to, or greater than zero respectively. Under this scheme, 19 users preferred *filtered*, 7 had no preference, and 5 preferred *standard*. Results are presented graphically in Figure 2, while the total number of judgements and judgement sum for each user can be seen in Figure 3.

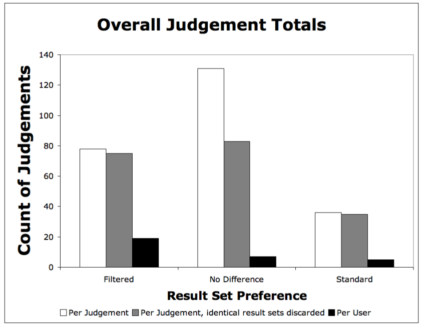


Figure 2: Overall totals of judgements. The white bars show total judgements overall, and the black bars show one judgement average preference for a user

We also counted the number of spam results present in each submitted query. 187 (88%) of all queries had some spam present, with a total of 613 labelled spam pages being presented to users in the *standard* result set (26% of all result pages presented in that panel). We visualise the distribution of user judgements with respect to the amount of spam present in the result set in Figure 4. There appears to be no correlation between number of judgements made by a user, and judgement preference (Figure 3).

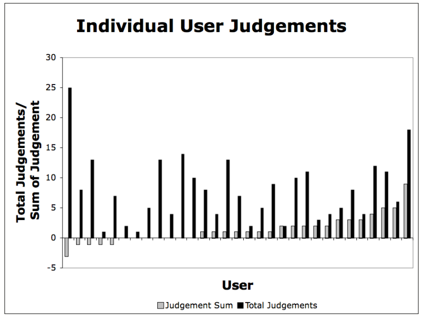


Figure 3: Judgement totals for individual users. The black lines show the total number of judgements, while the grey lines show the sum of that users judgements (plus one for each *filtered* vote, minus one for each *standard* vote).

3.4 Discussion

Ignoring the no difference votes, there is a strongly significant difference between the total votes for *filtered* and *standard* (Pearson's chi-square test, $p < 0.0001$). However, *filtered* cannot be said to be strictly better than *standard* as the total *filtered* votes are not greater than the total *standard* votes plus the no difference votes.

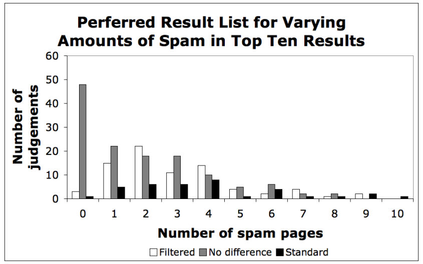
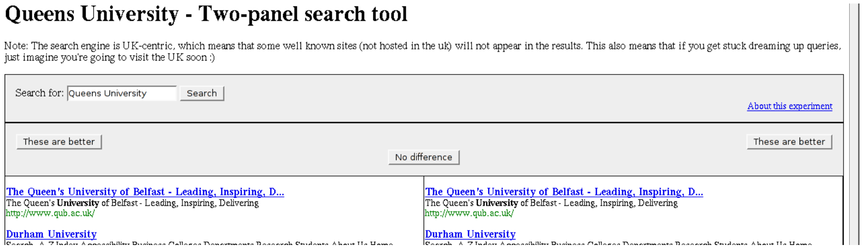


Figure 4: The distribution of judgements for varying amounts of spam in the top ten results. It is interesting that a stronger preference for the *filtered* set does not develop as more spam appears in the results.

Examining Figure 4 there appears to be no correlation between the amount of spam removed in the *filtered* set and the judgement that users make (other than a preference for no difference when no spam is present). It is not yet clear why this is, as intuitively more spam removed would equate to higher quality results.

Anecdotally, users observed that many search results which are not labelled as spam nonetheless did not deserve to be ranked as highly as they were. This may be due to incompleteness of the labelling or deficiencies

[5] using the data supplied with the UK2006 collection



Queens University - Two-panel search tool

Note: The search engine is UK-centric, which means that some well known sites (not hosted in the uk) will not appear in the results. This also means that if you get stuck dreaming up queries, just imagine you're going to visit the UK soon :)

Search for: Queens University

The Queen's University of Belfast - Leading, Inspiring, D...
The Queen's **University** of Belfast - Leading, Inspiring, Delivering
http://www.qub.ac.uk/

Durham University

The Queen's University of Belfast - Leading, Inspiring, D...
The Queen's **University** of Belfast - Leading, Inspiring, Delivering
http://www.qub.ac.uk/

Durham University

Figure 5: A screen shot of our two panel judgement interface. Two result pages are presented, and the user is invited to judge the left as better, the right as better, or both the same. The left and right order of the panels are randomised each query.

in our ranking algorithm. It also may be due to non-spam pages benefiting from artificially inflated quantities of links and anchortext. Future work is planned to investigate techniques for nullifying this sort of "optimisation".

In future work using this framework, we need to think carefully about what constitutes the ideal baseline. We want a ranking which is as high quality as possible, without employing any techniques for countering optimisation and spam. Because score components such as link counts, PageRank scores, and anchortext scores may change dramatically when counter-optimisation methods are applied, it is clear that we will need separate indexes to support the baseline and the spam-reduced version.

The queries made up by our volunteers are unlikely to be representative of the real work load of a UK search engine. For greater realism, we will recruit volunteers in the UK, or obtain lists of actual UK queries.

4 Conclusion and future work

With basic hardware, we successfully indexed the 2 terabyte, 80 million page UK2006 collection and implemented a UK search engine with sufficiently good result quality and response time to support an initial experiment in spam rejection.

Our evaluation method was sufficiently sensitive to detect differences between baseline and filtered rankings. We showed that spam does affect the quality of results for a large number of queries.

In future work, we plan to implement a better baseline and to compare it with a range of approaches to spam nullification.

Acknowledgements Many thanks to Paul Thomas for providing his two panel interface code and helpful advice. Also thanks to the many volunteers who supplied searches and judgements.

References

[1] L. Becchetti, C. Castillo, D. Donato, S. Leonardi and R. Baeza-Yates. Link-based characterization and detection of web spam. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2006.

[2] Andrei Z. Broder, David Carmel, Michael Herscovici, Aya Soffer and Jason Zien. Efficient query evaluation using a two-level retrieval process. In *Proceedings of CIKM '03*, pages 426–434, New York, NY, USA, 2003. ACM Press.

[3] Carlos Castillo, Debora Donato, Luca Becchetti, Paolo Boldi, Stefano Leonardi, Massimo Santini and Seb astiano Vigna. A reference collection for web spam. *SIGIR Forum*, Volume 40, Number 2, pages 11–24, December 2006.

[4] Kumar Chellapilla and David M. Chickering. Improving cloaking detection using search query popularity and monetizability. In *Proceedings of the Second International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, pages 17–24, Seattle, WA, August 2006.

[5] Z. Gyöngyi and Garcia-Molina H. Web spam taxonomy. In *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2005.

[6] David Hawking, Peter Bailey and Nick Craswell. Efficient and flexible search using text and metadata. Technical report, CSIRO Mathematical and Information Sciences, 2000. `http://es.csiro.au/pubs/hawking_tr00b.pdf`.

[7] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference*, pages 154–161, 2005.

[8] Xiaohui Long and Torsten Suel. Optimized query execution in large search engines with global page ordering. In *Proceedings of VLDB 2003*, pages 129–140, 2003.

[9] Alexandros Ntoulas, Marc Najork, Mark Manasse and Dennis Fetterly. Detecting spam web pages through content analysis. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 83–92, New York, NY, USA, 2006. ACM Press.

[10] Paul Thomas and David Hawking. Evaluation by comparing result sets in context. In *Proc. CIKM*, pages 94–101, Arlington, Virginia, USA, November 2006. ACM Press.

Search and Navigation in Structured Document Retrieval: Comparison of User Behaviour in Search on Document Passages and XML Elements

Gabriella Kazai
Microsoft Research
Cambridge, UK
gabkaz@microsoft.com

Abstract *This paper investigates search and browsing behaviour of users presented with two types of structured document retrieval approaches: passage retrieval and XML element retrieval. Our findings, based on the system logs gathered from 82 participants of the INEX 2006 interactive track experiment (iTrack), indicate that XML element retrieval leads to increased task performance. In addition, qualitative analysis of our video study, where we recorded the interactions of four participants, highlights potential issues with the experimental design employed at iTrack 2006.*

Keywords XML element retrieval, passage retrieval, INEX interactive track, video user study.

1 Introduction

Long before the eXtensible Markup Language (XML) [2] became widely adopted as a standard document format, work in the field of Structured Document Retrieval (SDR) was already underway to address shortcomings of traditional IR. Recognising that users are often only interested in the parts of a document that is relevant to their information need, researchers turned their attention to developing new retrieval approaches to deliver more focused results to users.

The underlying premise of SDR is that the logical structure of a document serves as a source of valuable information that should be exploited in indexing, retrieval and presentation of the document. The ultimate goal is to improve retrieval effectiveness through a more focused retrieval approach, which returns document components to the user, instead of complete documents, thereby reducing users' effort in locating relevant information. The need to consider smaller units within documents as retrievable entities has been considered particularly viable in the case of long documents or documents that cover a variety of topics.

Early examples of SDR include approaches to passage retrieval [3, 7, 9] and hypertext [5]. The more recent revival of SDR is a result of the widespread use of XML on the Web and in digital libraries, which has led to a drastic increase in the number of XML IR systems being developed [1]. Another important catalyst of recent advances in XML IR is that of the INitiative for the Evaluation of XML retrieval (INEX)[1] [6]. Since 2002, INEX has been promoting research in XML IR by providing a forum for researchers to evaluate their XML retrieval approaches and compare their results.

Since 2004, the interactive track (iTrack) at INEX [13, 10] has been investigating the behaviour of users when interacting with components of XML documents. In 2006, iTrack aimed to answer questions regarding the differences and similarities between XML element retrieval and passage retrieval approaches [11].

We took part in the iTrack 2006 experiments with 8 participants. The standard participation involved each participant completing four search tasks using either a passage retrieval or an XML element retrieval system. Data was collected through system logging as well as via questionnaires. In addition, we video recorded four of our participants and administered additional questionnaires. This paper reports our analysis of the log data combined with the qualitative data we collected from the videos. Analysis of the questionnaire data is reported separately in [8].

The paper is structured as follows. Section 2 details the experimental setup. Section 3 presents the results of our analysis of the system logs for 82 users. Section 4 summarises the findings of our video study for our 4 users. Conclusions and future work are reviewed in Section 5.

2 Experimental Setup

The interactive track (iTrack) at INEX 2006 [11] examined users' interaction with XML documents in an experimental laboratory environment. The task focused on comparing XML element retrieval with passage retrieval to explore potential benefits and trade-offs.

2.1 Document Collection and Topics

The experiment used the INEX Wikipedia collection [4] consisting of 659,388 documents totalling about 4.6GB of data and twelve topics (simulated work tasks).

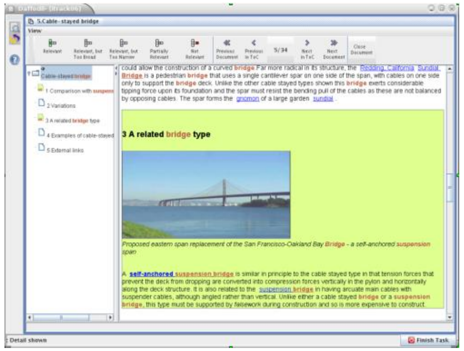
2.2 Search Systems

Participants were asked to perform search using two different search engines: the Panop-

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007.

[1] http://inex.is.informatik.uni-duisburg.de/

ticTM/FunnelbackTM passage retrieval system provided by CSIRO (passage system) and the TopX [12] XML element retrieval system (element system).



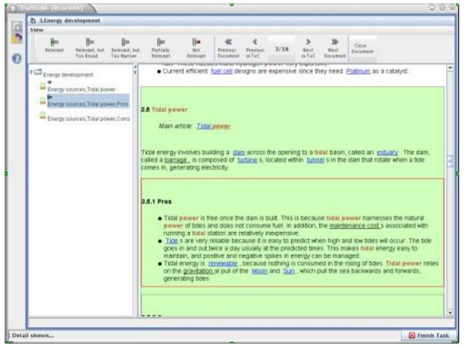
could allow the construction of a curved bridge.Far more radical in its structure, the Redding, California Sundial Bridge is a pedestrian bridge that uses a single cantilever spar on one side of the span, with cables on one side only to support the bridge deck. Unlike the other cable stayed types shown this bridge exerts considerable tipping force upon its foundation and the spar must resist the bending pull of the cables as these are not balanced by opposing cables. The spar forms the gnomon of a large garden sundial .

3 A related bridge type

Proposed eastern span replacement of the San Francisco-Oakland Bay Bridge - a self-anchored suspension span

A self-anchored suspension bridge is similar in principle to the cable stayed type in that tension forces that prevent the deck from dropping are converted into compression forces vertically in the pylon and horizontally along the deck structure. It is also related to the suspension bridge in having arcuate main cables with suspender cables, although angled rather than vertical. Unlike either a cable stayed bridge or a suspension bridge, this type must be supported by falsework during construction and so is more expensive to construct.

Figure 1: TopX XML element retrieval system.



- Current efficient fuel cell designs are expensive since they need Platinum as a catalyst.

3.5 Tidal power

Main article: Tidal power

Tidal energy involves building a dam across the opening to a tidal basin, called an estuary . The dam, called a barrage , is composed of turbine s, located within tunnel s in the dam that rotate when a tide comes in, generating electricity.

3.5.1 Pros

- Tidal power is free once the dam is built. This is because tidal power harnesses the natural power of tides and does not consume fuel. In addition, the maintenance cost s associated with running a tidal station are relatively inexpensive.
- Tide s are very reliable because it is easy to predict when high and low tides will occur. The tide goes in and out twice a day usually at the predicted times. This makes tidal energy easy to maintain, and positive and negative spikes in energy can be managed.
- Tidal energy is renewable , because nothing is consumed in the rising of tides. Tidal power relies on the gravitation al pull of the Moon and Sun , which pull the sea backwards and forwards, generating tides.

Figure 2: Panoptic passage retrieval system.

In order to remove any experimental bias due to user interface differences, both search engines were interfaced with a consistent look and feel. The similarity of the two systems can be seen in Figures 1 and 2. Figure 1 shows the front end to the TopX search engine, while Figure 2 shows the front end to Panoptic.

The difference between the two systems was subtle. In both, as a response to a user query, the search engine returned an ordered list of documents and, within each document, an ordered list of non-overlapping document parts. The main difference is in the returned retrieval entities: The passage retrieval backend returned non-overlapping passages derived by linearly splitting the documents. The element retrieval system returned XML elements of varying granularity based on the hierarchical document structure. In both versions, the passages and the elements were grouped by the document they belong to. Another important difference is that the table of contents in the document view of the element system included all sections down to a certain level, whereas the table of contents of the passage retrieval system only contained passages that were estimated relevant by the system.

Both systems indicated the parts of the documents that were estimated useful for the searcher in several ways: 1) Up to three high ranking passages/elements were shown per document in the result list. For each, a relevance bar icon indicated the degree of potential usefulness. 2) Relevant document parts were also indicated within the table of contents pane of the document view (see Figures 1 and 2) using the same relevance bar icons. Finally, 3) the relevant document parts were also highlighted in the text of the documents using shades of green and yellow to indicate varying degree of relevance.

2.3 Participants

Seven research teams took part in iTrack 2006, engaging a total of 82 participating users. Our own contribution to this pool was 8 users. Four of our users also participated in a video study, where we recorded their interactions with the search systems to supplement the log data.

We refer to our four participants (three male and one female) as Mark, John, Ben and Eva, respectively. Their average age was 32.25, the youngest being 24, the oldest 45 years old. On average, our users had 11.5 years of search experience on the Web. Summary information can be found in Table 1.

Table 1: Participants of our video study. Search experience is given in years. Acronyms: English (En), Other (Othr), Undergraduate Degree (U.Deg), Usability Consultant (UC), Researcher (Res), Student (Stu).

	Mark	John	Ben	Eva
Gender	M	M	M	F
Age	45	29	24	31
Native lang.	En	En	Othr	Othr
Education	PhD	PhD	U.Deg.	PhD
Occupation	UC	Res	Stu	Res
Web search exp.	10	12	12	12

2.4 Methodology

Initially, participants were presented with a sample topic and given as long as necessary to familiarize themselves with each search system. Each participant then had to complete four search tasks, choosing from a pool of three topics per task (see Table 2). The order in which each category topic was presented to a participant, and the system on which the search task was assigned changed between users as shown in Table 3. Fifteen minutes were allocated to complete a task using one of the search engines.

In addition to the standard track participation, we video recorded the search sessions of four of our participants in order to obtain detailed qualitative information regarding their interaction with the search systems. This allowed us to capture the context of users' interactions and interpret the actions logged by the system. In

Table 2: Topic categorisation.

Category	Topics	Category	Topics
A1	1,2,3	B1	2,3,4
A2	5,6,7	B2	6,7,8
A3	9,10,11	B3	10,11,12
A4	4,8,12	B4	1,5,9

Table 3: Permutations of the topic categories and retrieval systems across participants.

System:		Element		Passage	
Participant		**Topic categories**			
P1	–	A1	A2	A3	A4
P2	–	A2	A1	A4	A3
P3	–	A3	A4	A1	A2
P4	–	A4	A3	A2	A1
System:		**Passage**		**Element**	
Participant		**Topic categories**			
P5	Mark	B4	B3	B2	B1
P6	John	B3	B4	B1	B2
P7	Ben	B2	B1	B4	B3
P8	Eva	B1	B2	B3	B4

the following section, we describe the findings from the log data analysis for all 82 participants and then discuss in detail the findings from our video recordings.

3 Log Analysis

The collected log data from search tasks of 82 participants comprises 378 logged search sessions, out of which 301 were completed or had valid XML output (some were aborted or restarted). The statistics presented here are based on these 301 session logs.

3.1 Search Sessions

A search session was defined as one experiment involving one participant and one search task. The maximum session duration was hence 15 minutes, the time limit allocated to a task.

Table 4 summarises our findings. From the total of 301 sessions, 173 were search tasks conducted on the passage system and 128 on the element retrieval system. The average session duration was 10 minutes, 2/3-rd of the allocated time. On average, users spent less time using the passage retrieval system: 9.8 minutes per session, compared with 10.4 minutes using the element retrieval system. This alone, however, does not reveal whether users were successful in completing their tasks within that time. Participants may have given up their search or may have run out of time before finishing a task.

Users averaged 0.49 and 0.74 queries, 0.52 and 0.54 result clicks, and 0.98 and 1.3 visited (browsed) components per minute on the passage and element retrieval systems, respectively. Thus, they issued more queries, clicked on more results in the retrieved list and browsed more inside documents with the element retrieval system than with the passage system. This could be attributed to the differences in search performance, suggesting that the passage system was better at answering users' queries. However, it could also mean that users were better supported in their browsing using the element system. Note that the average query rate per second is only 0.01 across all iTrack experiments, compared with 0.028 reported in [14] for the Web.

3.2 Search Trails

To study users' post-query browsing behaviour, we extracted search trails from the system logs. We defined a search trail as a series of visited document components, initiated with a click on a result in the ranked list, and terminated by a new query or by the end of the session. We extracted 812 trails in total. From these, we calculated the features described in Table 5.

Among 812 trails, 460 were obtained from the 173 logged sessions on the passage system (2.65 trails per session) and 352 from the 128 sessions on the element system (2.75 trails per session). This means that users created roughly the same number of trails in both systems. The average trail length indicates that users visited, on average, 1.4 more document components per query using the element retrieval system (6.9) than with the passage system (5.5). However, they spent on average less time per trail step using the element system (9 seconds vs. 10.3 seconds for the passage system). This suggests that users accessed information in smaller chunks using the element retrieval system, which took less time to skim over but lead to extended trails. A look at the medians on the other hand reveals that the element retrieval system's average trail duration is heavily influenced by outliers.

The logs contained a total of 1580 result clicks (856 in the passage and 724 in the element system). The average rank position of clicks reveals that users looked, on average, further down the ranking with the element system (rank 6 vs. 4.9) than with the passage system.

3.3 Search Success

During the experiments, participants provided relevance feedback by assigning one of five relevance grades (not relevant, partly relevant, too big, too narrow, fully relevant) to some of the visited document components. Table 6 presents statistics on the collected relevance assessments. A total of 2308 judgements were collected: 1169 on the passage (6.8 per session) and 1139 on the element system (8.9 per session). Out of these, 1833 document components were judged relevant: 943 on the passage (5.45 per session) and 890 on the element system (6.95 per session). This means that overall, users found and judged more relevant document components using the element system, thus were likely to have been more successful in completing their task. Comparing this with our reported session durations, we can conclude that although users spent less time on a task using the passage system, they were also less successful. Thus, the reduced session duration

Table 4: Analysis of search sessions for passage and XML element systems.

	Total	Passage	Element
Total sessions (analysed)	301	173	128
Average session duration (mins)	10	9.8	10.4
Average number of queries issued per session	6	4.7	7.7
Average number of queries per minute	0.6	0.49	0.74
Average number of result clicks per session	5.2	4.9	5.7
Average number of result clicks per minute	0.52	0.51	0.54
Average number of browsed components per session	11.2	9.5	13.5
Average number of browsed components per minute	1.12	0.98	1.3

Table 5: Analysis of search trail features for passage and XML element systems. Standard deviation is shown in brackets.

	Total	Passage	Element
Total search trails	812	460	352
Average trail duration (sec)	59 (131.4)	56.9 (140.9)	62.5 (117.8)
Median trail duration (sec)	14	15.3	13
Average trail length	6.1 (7.2)	5.5 (7.1)	6.9 (7.2)
Median trail length	4	4	5
Average step duration (sec)	9.7	10.3	9
Average result click position	5.4 (7.2)	4.9 (6.6)	6 (7.8)
Median result click position	3	2	3

may in fact reflect users' tendency to give up their task using the passage system.

Looking at the average rank of relevant components in the result ranking, however, shows that users of the element system had to look further down the ranking to find relevant information: on average 5.6 ranks vs. only 3.6 ranks for the passage system. This may suggest that while the passage system was better at ranking the results in the ranked list, it was then easier for users to discover more relevant information through browsing using the element system.

4 Video Study

In this sections, we provide a qualitative analysis of our users' search and browsing behaviour from our video recordings with the aim to gain detailed insight into the context of their logged actions.

4.1 Search and Navigation

Our four participants displayed varying strategies to search and navigation in the experiments. We provide a detailed review of their actions based on collected video footage that we cross-referenced with the system logs.

Mark. Mark chose topic 9 (C4) for his first task. Unbeknown to him, he carried out the task using the passage retrieval system. He only issued a single query during the whole session and worked his way down the ranked list in a linear fashion. Even though the task had to be restarted twice due to system problems, he simply re-entered the same query and continued where he left off in the ranking. Initially, his interactions were limited to selecting the whole document as entry point and then scrolling inside the document. His speed of scrolling was influenced by the text-highlights: he would stop or slow down once he reached highlighted parts. Six minutes into the task, he realized that he can navigate inside the document using the table of contents (ToC). From then on, he relied on this form of navigation almost exclusively. He would still click the whole document from the ranked list, but then he would jump straight to a highlighted section using the ToC. He reduced his scrolling activities to the occasional scroll around a highlighted text fragment either for context or in search of further relevant information. In total, Mark viewed 18 document components. The first 7 involved document level entry and scrolling. From the last 11, in 9 instances he entered at the document level, but immediately navigated to a passage using the ToC; and in 2 cases he entered directly by choosing a passage from the ranked list.

Mark's second task was topic 10 (C3), using the passage system. Again, he entered only one query during the whole session, which he repeated when the task was restarted due to a system error. As a result of the system completely crashing then, he only managed to visit 3 results before the task was aborted. In all 3 of these cases, he selected a passage directly from the ranked list and employed no additional scrolling or navigation inside the document.

For his third task, he chose topic 8 (C2) and used the element retrieval system. He issued 9 queries, but only examined results returned for 2 of these. In total, he viewed 6 results: 5 element level and 1 document level entry. He did not do any scrolling inside the document

Table 6: Analysis of search success for passage and XML element systems. Standard deviation is shown in brackets.

	Total	**Passage**	**Element**
Total relevance judgements	2308	1169	1139
Total number of relevant components	1833	943	890
Average rank of judgement	4.9 (7)	4.2 (6.8)	5.7 (8.9)
Median rank of judgement	3	2	3
Average rank of component judged relevant	4.6 (6.1)	3.6 (5.5)	5.6 (8)
Median rank of component judged relevant	2	2	3

in 4 out of the 6 instances and only once clicked on an entry in the ToC.

In his final task, he worked on topic 3 (C1) using the element system. This was the only task that he successfully completed (as stated by himself in the post-task questionnaire). He issued 2 queries and viewed 3 results in total: 2 element-level and 1 document level. All 3 cases involved extensive scrolling inside the document and no interactions with the ToC.

In summary, Mark's interactions show a learning curve and point towards preference for the SDR approach, i.e. having direct access to relevant text fragments. His initial strategy for navigating inside a document using the ToC was later replaced by extensive scrolling. Once he became confident with selecting passage level entries, he completely stopped using the ToC for navigation. When scrolling, he would typically stop at the first highlighted fragment. He would also usually scroll around a relevant fragment to obtain more context.

John. In his first task (topic 11 in C3 on the passage system), John issued 2 queries and viewed 3 results. In all three cases, he chose the document as his entry point but then he made extensive use of the ToC to navigate (clicked a total of 9 ToC links). He also scrolled frequently in both directions (up and down) from an entry point. Due to UI design issues, whereby scrolling in the right document view pane was independent from the item selection in the ToC pane, he got disoriented on multiple occasions. To reorient himself, on one occasion he browsed through the ToC (adding an additional 10 ToC clicks to the log, which reported a total of 19 ToC clicks).

He initiated 5 queries in his second task (topic 9 in C4 on the passage system) and viewed 7 results. For one query, he scrolled all the way till the end of the ranking but did not click any results. For all 7 clicked results he chose document level entry points and scrolled inside the document.

For his third task (topic 3 in C1 on the element system), he ran 7 queries and viewed 6 results, all via document level entry. Whilst viewing the fourth document, the system developed a fault with the scrollbar. In an effort to try to get around this, he clocked up an additional 9 clicks on various ToC links.

For his final task (topic 6 in C2 on the element system), he ran 3 queries and viewed only one returned result for each. The only time he clicked a passage level entry was for the final result, but then he scrolled to the top of the document and back down again. For 2 of the 3 viewed documents, he relied exclusively on scrolling to navigate inside the document. This was the only task he failed to complete.

In summary, John followed his own established search and navigation strategy. He was familiar with standard retrieval systems: he chose document level entry points and scrolling for navigation. Unlike Mark, John's search and navigation strategy was constant throughout. He used many queries but would, on average, view one result per query. Unless a relevant document was presented within the top 2 ranks, he would quite often scroll through the whole ranking before making his selection. He would always choose the whole document as entry point. Inside the document, he would always first browse by scrolling and then combine the use of ToC with further scrolling. The speed of his scrolling was influenced by the text highlights: he would slow down when scrolling over highlighted document parts. He would also scroll around highlighted areas for context.

Ben In his first task (topic 7 in C2 on the passage system), he ran 5 queries and viewed 1 document for each. In each case, he chose passage level entry points into the documents. He kept scrolling to a minimum and only once used the ToC for navigation.

For his second task (topic 2 in C1 on the passage system), he issued 4 queries, 1 of which did not result in any viewed documents. From the total of 6 viewed documents, 3 were reached through document level entry points. For navigating inside the documents, he employed the use of both the ToC (he clicked 6 ToC entries in total) and scrolling. He regularly scrolled up in a document to revisit already skim-read sections.

During his third task (topic 1 in C4 on the element system), he entered 7 queries, 2 of which were unsuccessful in providing any documents of interest to Ben. He viewed 7 documents, 3 of which he entered at the document level. For this task, he only once clicked on a link in the ToC. The rest of the time he simply scrolled. The text highlighting did not influence his speed of scroll.

His final task (topic 10 in C3 on the element system) showed a different behaviour. For 9 issued queries he viewed 8 documents, 7 of which by document level entry. He regularly clicked around in the ToC or scroll around in a document, but did not combine both forms of navigation inside a single document.

In summary, Ben's search and navigation behaviour was a mixed bag. He combined all methods and was as successful in completing his tasks as John (only failed to finish task 4). His strategy was very flexible, easily adapting to the task at hand. He confidently used the various forms of navigational methods inside a document. He varyingly chose between document level entry points and directly accessing relevant text fragments from the ranking.

Eva For her first task (topic 3 in C1 on the passage system), Eva ran 11 queries, but viewed only 4 results. After inputting a query, she would inspect the ranked list scrolling down till rank 9, on average. Once she found a relevant document, she spent on average 3.6 minutes on browsing and reading through it. She always chose document level entry points. Inside a document she would typically scroll. Although she also generated 16 ToC hits, in her own words, these were only clicked in order to allow her to make relevance judgments.

For her second task (topic 8 in C2 on the passage system), she used 5 queries, but inspected only 3 results due to system errors. For all 3 results she chose passage level entry points. She only used the ToC on one occasion in this task, which was again related to system error: She clicked a passage level entry point, but found that the document part shown in the right pane did not match the ToC entry. In order to realign the screens she toggled between entries in the ToC.

In her third task (topic 10 in C3 on the element system), she ran 4 queries and viewed 5 documents, all through passage level entries. Again she only used the ToC in order to provide relevance feedback. She completed this task just under 10 minutes.

For her final task (topic 1 in C4 on the element system), she issued 15 queries, but failed to complete the task. After carefully looking through the result lists, she only chose to view 7 documents, accessing them directly at the passage level. Inside the document, she scrolled relatively little and again only clicked in the ToC before making a relevance judgment.

In summary, Eva tended to either only choose document level or passage level entries, but not both types within an individual task. Although she stated that she only selected items in the ToC to make judgments, in her first task, she actually did make quite a lot of use of the ToC. She has even adjusted the UI to show more of the ToC and she spent long periods of time looking through and reading the ToC. Eva managed to complete half of her tasks (tasks 2 and 3).

4.2 User Opinions on Passage vs. Element Retrieval

In a post-task questionnaire, users rated their search experiences after each task using a 5 degree scale (1=frustrating, 3=neutral, 5=pleasing). Table 7 shows the averaged ratings for the two systems. Based on these numbers it appears that the XML element retrieval system is preferred by our participants as it leads to higher user satisfaction. In order to find out what criteria these overall ratings were formed, we asked participants to explain their reasoning for the assigned score.

Mark told us that he was frustrated with the passage retrieval system as it crashed for his first two tasks. He was hence unable to find any relevant information to complete the tasks. He was happier with the element retrieval system as it did actually work and he also managed to solve his fourth task. His rating in this case was based on the fact that the system "performed better" and that he "was able to find all the necessary information" for his task.

John rated both systems equal on average. His explanation was that "compared to Google or Wikipedia, the task took [him] about the same time [to complete using these systems]. Usually [he] has more windows open but the current user interface was not very flexible." He also commented that a lot of the new user interface features, such as the ToC, he did not use as he was more familiar with Web search. So, his rating was based on a combination of system performance (effectiveness and speed) and user interface features. His rating was not influenced by his success or failure in completing a task (e.g. he gave a rating of 2 after his successful completion of task 3, and a rating of 4 after he failed task 4).

Ben again rated the two systems equal on average. His rating was a reflection on a mixture of criteria: whether he was successful in his task, the retrieval effectiveness of the system (whether it was able to return relevant hits: "nothing directly mentioned peaceful revolution"), system features and their usefulness. For example he commented "the related terms was useful here", "the table of contents was quite hard to use here", and "the extra features didn't help here".

Eva's ratings were mostly dependent on her completion of the task. Additional factors for her task 1 rating included some user interface aspects: "I am used to how I search on the Web. [As] this task has sub-tasks, I would start several browsers for each sub-task. I could not do this here."

From the range of replies, we can see that a system's ability to locate relevant information is one of the main factors of user satisfaction. This is then supplemented by factors such as efficiency and general user experience supported by system features and user interface.

In a post-experiment questionnaire, users were asked to identify the differences between the two systems. Our participants' answer to this was unanimous: None of them were able to identify any difference between the element and passage retrieval systems. This would suggest that the use of structure (as exposed by the two systems) did not play a role in our users satisfaction with the systems. However, once the differences were explained to them, they did comment that the ToC for the element retrieval system was more useful as it allowed navigation to any structural part of a document. The passage retrieval system at best had three passages listed in its ToC. This has, in fact, presented an issue to John, who was thus unable to provide judgement for a passage he thought relevant, but which was not listed in the ToC. Since he had no means of adding the passage to the ToC, he was unable to provide relevance judgement for it.

Table 7: Averaged user rating of search experience for passage retrieval system (P) and XML element retrieval system (E).

	Passage	**Element**
Mark	1	3.5
John	3	3
Ben	3.5	3.5
Eva	3	3.5
Overall	2.6	3.4

4.3 System Logs vs. Video Evidence

During the combined analysis of the system log and the video study data, we came across a number of anomalies. This has lead us to uncover some issues with the iTrack experiment where the log data may lead to incorrect conclusions. For example, the logs reported a very high ratio of within-document navigation using the ToC (over 22% of all logged actions over all 82 participants; with average trail length of 6.1 steps). This would suggest to system designers that the table of contents was a very useful navigation tool for users. However, our video recordings indicate that this is likely to be a highly over-exaggerated figure. We found that users may click repeatedly on entries in the ToC in response to system hickups and errors, or as workarounds for system design faults. For example, both Ben and John used the ToC links (and clicked back and forth several times) simply to re-orient themselves within a document after the display crashed. Both Mark and Eva had to close documents and then re-open them due to the system's failure to display the document's content. They would then often click several links in the ToC just to check if the system was still responding. In addition, John clicked on ToC entries just to re-align the two panes of the display after he has been scrolling up and down in a document.

Another reason why participants clicked links in the ToC was to make relevance judgments. This was a system-imposed limitation whereby only document parts selected in the ToC could be judged and only when users selected the document part using the ToC. This has lead to a large increase in ToC navigation clicks. Based on our video records that we cross-referenced with the system logs we estimate that for our 4 participants less than half of the logged ToC clicks were valid navigational user actions. Over 50% is attributed to consequences of system design constraints or system errors and crashes.

Additional issues with the systems included that some ToC links were incorrect, their title text was wrong or they took the user to the incorrect location. This has again led to increased user navigation, but also resulted in users more frequently abandoning a particular search inside a document.

We have also witnessed users mistakenly assigning a relevance score to the wrong document part. John, for example, has on numerous occasions lost the synchronization between the left and right panes of the document view as he often scrolled inside the document. As a consequence, he has on two occasions marked the wrong section of the document relevant (i.e., the selected part in the ToC was different to what was actually shown on the screen). The same problem showed up in all our participants' videos.

5 Conclusions

In this paper we reported on our experience of participating in the INEX 2006 iTrack experiments. We provided an analysis of the system logs collected for all 82 participants at iTrack.

In addition, we gave a detailed account of four of our participants' search and navigation behaviour based on combined evidence extracted from the logs and from our video study. We found that users have their own personal styles of searching and navigating. Some users adopt new strategies easily while others prefer to stick with tried and tested methods. An obvious implication for the design of SDR systems is that any new forms of interaction and navigation has to be supported by simple and self-explanatory user interface features. Furthermore, the interaction model needs to be useful enough to promote its use.

When comparing passage and XML element retrieval methods, we found evidence to suggest that element retrieval led to increased task performance with more document components found and judged relevant. This was achieved by users at a cost of spending on average more time on a task and issuing more queries per session. On average, users would also browse more with an element retrieval system, leading to an average search trails of 6.9 visited document components (vs. 5.5 using the passage system). Both are, however, above those reported in [14] for Web search.

Furthermore, our users rated on overall the element retrieval system above the passage system (without knowing that they were rating two different systems). We also found evidence to suggest that users' navigation behaviour differs across the two systems: Participants were more likely to select full documents as entry points using the passage system and then make more extensive use of the ToC. In element retrieval, participants more often chose document parts as entry points but were then less likely to use the ToC for navigation. This can be motivated by the argument that once users gained direct access to relevant parts, they did not need to navigate as much inside the document.

Finally, our investigation has highlighted a possible issue with the experimental design adopted at iTrack. We found evidence that users' behaviour was inadvertently affected by a system imposed constraint: In order to provide relevance judgements on a document component, users had to ensure that the component was selected in the ToC. This meant that users were often forced to click an entry in the ToC before being able to make relevance judgements, and thus increasing ToC click statistics in the logs. One consequence of this is that the average search trail length reported here could well be an overestimate.

A limitation of this study lies in the question of the generality of its findings. While the set of 82 users of the overall iTrack experiments represent a sufficiently large user population, experienced system errors and crashes raise questions on the fidelity of the collected data. For example, out of the total 378 sessions 59 were generated as a result of 41 restarted search tasks (some of which were restarted multiple times) affecting 33 users in total. On the other hand, our video study only included a small group of 4 participants.

Another issue concerns the Wikipedia collection used in the experiments, where most articles are short or are divided into small chunks. Such a collection may not be best served by passage retrieval techniques as highlighted in [7]. Further studies are thus needed to determine if users prefer passage or element retrieval systems.

Our future work will extend the analysis presented here to compare systems on a per topic basis. We will also look to run user studies on a larger scale incorporating system logging and video capture. Our aim is to build on our findings in order to design appropriate user interfaces for SDR. In addition, we hope that our conclusions regarding the analysis of the system logs will help other iTrack 2006 participants in their work.

Acknowledgements Many thanks to Natasa Milic-Frayling, Birger Larsen and the anonymous reviewers for their many helpful comments.

References

[1] Henk M. Blanken, Torsten Grabs, Hans-Jörg Schek, Ralf Schenkel and Gerhard Weikum (editors). *Intelligent Search on XML Data, Applications, Languages, Models, Implementations, and Benchmarks*, Volume 2818 of *LNCS*. Springer, 2003.

[2] T Bray, J Paoli and C.M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0. http://www.w3.org/TR/1998/REC-xml-19980210, W3C Recommendation. Technical report, W3C (World Wide Web Consortium), February 1998.

[3] James P. Callan. Passage-level evidence in document retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 302–310, New York, USA, 1994. Springer-Verlag New York, Inc.

[4] Ludovic Denoyer and Patrick Gallinari. The wikipedia xml corpus. In Fuhr et al. [6], pages 12–19.

[5] Mark Edwin Frisse. Searching for information in a hypertext medical handbook. In *HYPERTEXT '87: Proceeding of the ACM conference on Hypertext*, pages 57–66, New York, NY, USA, 1987. ACM Press.

[6] Norbert Fuhr, Mounia Lalmas and Andrew Trotman (editors). *Comparative Evaluation of XML Information Retrieval Systems, 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006, Dagstuhl Castle, Germany, December 17-20, 2006, Revised and Selected Papers*, Volume 4518 of *Lecture Notes in Computer Science*. Springer, 2007.

[7] Marcin Kaszkiel and Justin Zobel. Effective ranking with arbitrary passages. *JASIST*, Volume 52, Number 4, pages 344–364, 2001.

[8] Gabriella Kazai and Andrew Trotman. Users' perspectives on the usefulness of structure for XML information retrieval. In *Proceedings of the 1st International Conference on the Theory of Inofrmation Retrieval*, pages 247–260, 2007.

[9] Xiaoyong Liu and W. Bruce Croft. Passage retrieval based on language models. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 375–382, New York, NY, USA, 2002. ACM Press.

[10] Saadia Malik, Birger Larsen and Anastasios Tombros. Report on the INEX 2005 interactive track. *SIGIR Forum*, Volume 41, Number 1, pages 67–74, 2007.

[11] Saadia Malik, Anastasios Tombros and Birger Larsen. The interactive track at INEX 2006. In Fuhr et al. [6], pages 387–399.

[12] Martin Theobald, Ralf Schenkel and Gerhard Weikum. An efficient and versatile query engine for TopX search. In Klemens Böhm, Christian S. Jensen, Laura M. Haas, Martin L. Kersten, Per-Åke Larson and Beng Chin Ooi (editors), *VLDB*, pages 625–636. ACM, 2005.

[13] Anastasios Tombros, Saadia Malik and Birger Larsen. Report on the INEX 2004 interactive track. *SIGIR Forum*, Volume 39, Number 1, pages 43–49, 2005.

[14] Ryen W. White and Dan Morris. Investigating the querying and browsing behavior of advanced search engine users. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 255–262, New York, NY, USA, 2007. ACM Press.

Document Composition and Content Selection Evaluation

Shijian Lu
CSIRO ICT Centre
Locked Bag 17,
North Ryde NSW 1670 Australia

Shijian.Lu@csiro.au

Abstract *Our work is concerned with the design of adaptive hypertext systems that produce documents tailored to their intended reader. In our approach, a system composes document on-the-fly, assembling existing text fragments. One of our challenges in this approach is to support the technical writer who configures the system. The task of the technical writer is to specify the structure of the documents to be generated, together with their applicability conditions. To perform their task, authors need to know what information is available. In this paper, we examine the impact of different strategies for presenting the existing text fragments on the task of document composition. We focus in particular on the impact on the quality of the resulting documents. We found that people compose better documents when existing text fragments are presented in a structured way.*

Keywords document composition, information reuse, document quality, evaluation, method.

1. Introduction

Communication is important to a successful organisation. Effective communication is often enabled by coherent documents. Many people are familiar with or have the experience of writing documents for some specific purpose. For example, a communicator in a research organisation may be asked to create project flyers for potential clients or the general public. Producing documents this way is a manual and laborious process. Now, the ever increasing availability of information content and advancement of natural language generation technology have made it possible to augment the task of composing documents from existing content segments. Myriad [8] delivery platform represents one such research initiative towards enabling rapid document composition.

At the core of Myriad is the VDP (virtual document planner) which embodies a plan-based approach to discourse generation based on [5]. It is through discourse operators that one defines the types of text to be produced. However, authoring discourse operators is not an easy task as expertise and knowledge from many areas are required. In order to reduce the requirement threshold, we have introduced the concept of content structure [1] which is inspired by the RST (rhetorical structure theory) [4]. A content structure is composed of content nodes organised vertically as a hierarchy and horizontally as RST units. A design supporting tool Constructor [3] has been developed. Using Constructor, content structures can be visually defined from which discourse operators can be generated automatically.

We sought to evaluate our first prototype of Constructor, to investigate both the feasibility of the approach and the usability of the prototype. We started our study by an expert user evaluation in recreating Scifly [7]. One of the main difficulties uncovered was in finding what data was available to include in a document (i.e., what data could a document designer exploit to compose a document). While Constructor provides a list of retrieval services, these are currently displayed as a flat list. The expert user found it hard to locate relevant retrieval services. We conjectured that providing structure to present existing content fragments would make it easier to locate appropriate content fragments. As a result, (1) document would get composed more quickly and (2) the result would be of better quality. To test our hypotheses, we did a user experiment. It confirmed our first hypothesis regarding the time [2]. Here, we report our findings on hypothesis (2) regarding the quality of the resulting documents.

In the next section, we describe our experiment set-up before presenting our analysis on the composed documents. In Section 4, we discuss flyer content quality evaluation. The paper concludes in Section 5.

2. The experiment

Our objective was to understand how different presentations of the existing content fragments (to be used to compose a new document) would affect the task of document composition. We focus here on the quality of the resulting documents. Our hypothesis is that providing structure to present existing content fragments would result in documents of higher quality than the documents composed when the text fragments were presented to authors without structure. Twelve CSIRO employees were randomly selected to participate in the experiment and were randomly divided into two equal sized groups. One subject could not finish the experiment because of time constraints. In the remaining subjects, there were five scientists, four research engineers and two administrative staff. Among them, there were eight men and three women.

All subjects were asked to perform the same two tasks: (1) compose a flyer about a single project ("Web Service Integration") and (2) compose a flyer two projects (specifically, "Under Water Vehicle" and "Virtual Critical Care Unit"). One group of participants

(Group A) were given the existing text fragments in an unstructured list while the other group (Group B) were given a structured list. In both tasks, subjects were presented with information regarding ten projects, from which they had to pick the appropriate information for the project(s) of the flyer they were writing. The information was made available through the two interfaces.

The domain data for the experiment was a subset of the data used in the Scifly application [7]. The input material was presented in HTML format in a browser interface. Subjects used a web browser to access the data. The interface consisted of two parts: the list part and content part. The actual content in the content part was changed according to what was selected from the list. Initially, the content part was empty. There were 205 items in the list. The list was represented in two different ways: an unstructured list and a semantically structured list.

Essentially, participants copied selected content from the web browser (from the input material), pasted it into an MS Word document, structured and ordered the content appropriately. All experiment sessions were video recorded. Pre-experiment and post-experiment questionnaires were also used. In addition, all sessions were observed, and subjects' actions were noted on paper. The result on time completion has been analysed elsewhere [2]. In the next section, we will analyse the project flyers composed by the subjects.

3. Analysis of composed flyers

At the end of the experiment, two sets of flyers have been composed by subjects: single project flyers and two-project combined flyers. After close examination of the resulting flyers, two interesting characteristics are discovered. One is about the presence of different types of content fragments. The other is about flyers length which shows close correlation with input material organisation.

3.1.Classifying content fragments

By studying the generated flyers, it is found that content fragments included in these flyers fall into four different types: namely, important content fragments, repetitive content fragments, irrelevant content fragments and marginal content fragments. Important content fragments are those which are important to flyers quality which based subjects' response in the post-experiment questionnaire. It is found that on average the number of important content fragments for the two groups are very close: 6.8 for Group A and 7.2 for Group B. Repetitive information refers to content fragments which are repeated multiple times in a flyer. We found that the total number of Repetitive fragments for Group A is 7 which are much larger than Group B's 2. Irrelevant information refers to content fragments which are not related to the interested project, research laboratory, or ICT Centre. Marginal content fragments are relevant to the topics of flyers. But, they are neither important, nor repetitive. There were 13.3 marginal fragments for Group A and 6.2 for Group B.

3.2. Flyer length

We look at the flyer length in terms of the number of pages and the number of content fragments which a flyer contains. Although no explicit length limit is given before the experiment, there is a big difference between the two groups. In terms of number of pages in the resulting flyers, Group A ranges from 1 to 4 pages while the spread for Group B is between 1 and 2. The average number of pages is 2.4 for Group A and 1.4 for Group B. Evidently, the average page number for Group A of 2.4 is much bigger than 1.3 for Group B. Indeed, the t-test shows that the difference is statistically significant.

In terms of number of content fragments, Group A ranges from 7 to 21, while the spread for Group B is between 7 and 12. The average number of content fragments is about 15 for Group A and about 11 for Group B. The difference, however, is not statistically significant.

4. Flyer quality evaluation

In order to assess the effect of organisation of input material on flyer composition quality, we need a reasonable method. As mentioned earlier, flyer quality is used to refer to content selection rather than flyer structure or flyer layout. To our knowledge, content selection evaluation has not been studied in the context of document composition. However, it has been studied in the context of summarisation [6] [9]. The Pyramid method [6] is an empirically motivated method for evaluating the quality of summarisation.

Our analysis of flyer content described in Section 3 concludes that there are four different types of content fragments. These different types of content fragments play different roles in flyer quality. Specifically, repetitive and irrelevant content fragments will contribute negatively towards flyer quality while important and marginal content fragments will contribute positively and neutrally towards flyer quality. Furthermore, irrelevant content fragments will do more damage to flyer quality than repetitive ones. Based on the observation that, in document composition, there are not only positive contribution content but also negative contribution content, the X-method is developed which is an extension of the Pyramid method for dealing with the impact pollutant content fragments. The X-method is also a X of content fragments which consists of two opposing pyramids (e.g., *Figure 1*): one positive pyramid and one negative pyramid. Like the Pyramid method [6], based on a pool of human input, a pyramid of important content fragments (ICC) will be computed. Each ICC has a weight corresponding to the number of nomination it gets as described in Section 3.1. For example, the content fragment "*contact*" is nominated by 8 subjects while "*background*" is nominated by 6 subjects. This forms the positive pyramid of IICC arranged by weight in descending order. Unlike the Pyramid method, a pyramid of pollutant content fragments is also formed. The default weights for irrelevant and repetitive content fragments are -*n* and -$\frac{n}{2}$, here, n is the number of tiers in the positive pyramid.

With X-method, the score of a flyer D_{pyr} is the ratio of the sum of the weights of its content fragments D to the sum of weights of the optimal flyer with the same number of content fragments $D_{\max}$.

$$D_{pyr} = \frac{D}{D_{\max}} \qquad (1)$$

Where, D *-- the sum of the weights of its content fragments;*

$D_{\max}$ *-- the sum of weights of the optimal flyer with the same number of content fragments.*

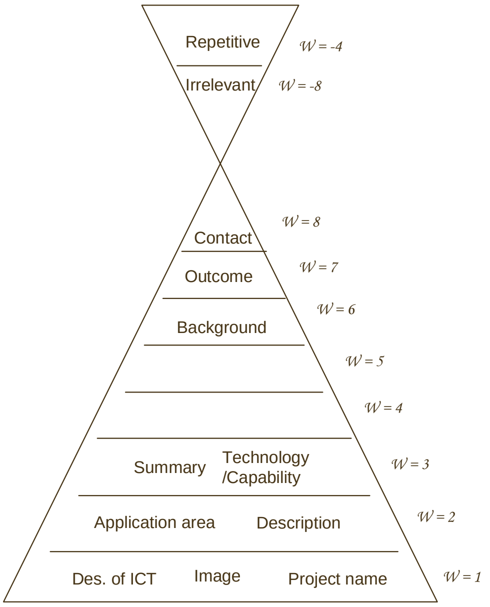


Figure 1. Content fragments weights for the X-method.

Suppose the positive pyramid has *n* tiers, T_i , with tier T_n on top and T_1 on the bottom. The weight of content fragments in tier T_i will be i . Let $|T_i|$ denote the number of content fragments in tier T_i . Let D_i be the number of content fragments in the flyer that appears in T_i . Content fragments in a flyer that do not appear in the pyramid are assigned weight zero. The total content fragment weight *D* is:

$$D = \sum_{i=1}^{n} iD_i - \tfrac{n}{2}\sum_{j=1}^{2} jR_j \qquad (2)$$

Where,

R_1 *-- the number of repetitive content fragments in a flyer;*

R_2 *-- the number of irrelevant content fragments in a flyer;*

The maximum content score for a flyer with x content fragments is:

$$D_{\max} = \sum_{i=j+1}^{n} iT_i + j\left(x - \sum_{i=j+1}^{n} T_i \right) \qquad (3)$$

Applying equations (1), (2), and (3) to the experiment data for Task 1, the X-method scores can be calculated for different flyers produced by the two groups of subjects (Figure 2). As you can see, the average score for the Group A is 0.77 compared to 0.91 for Group B. That means that the average flyer quality for the structured group is better than those for the unstructured group. However, the t-test (Table 1) shows that the difference is not statistically significant.

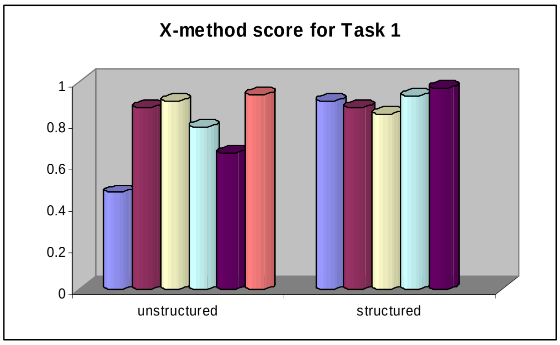


Figure 2. Quality score for Task 1

Table 1. T-test results in terms of the X-method scores for task 1

Task	Mean_A-Mean_B	t	df	P one-tailed
1	-0.1662	-1.67	9	0.064629

Similarly, the X-method scores for the different flyers composed by the two groups of subjects are calculated which is listed (Figure 3). Interestingly, this time, Group A on average scored 0.78 which is considerably higher than 0.61 scored by Group B. That means that the structured group composed better two-project combined flyers than the structured group. However, the difference is not statistically significant as demonstrated by t-test (Table 2).

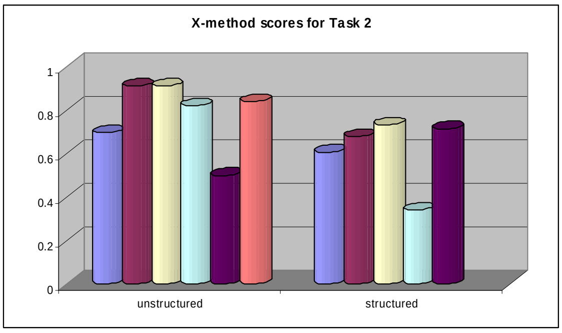


Figure 3. The quality score for Task 2

Table 2. T-test results in terms of the quality scores for task 2

Task	Mean$_A$- Mean$_B$	t	df	P one-tailed
2	0.1652	+1.71	9	0.06072

5. Discussion and conclusion

In this paper, we have presented an empirical study on document composition. It is found that the unstructured group produced significantly longer flyers than the structured group for both tasks. We surmise that this length discrepancy is caused by the difference in the organisation of content fragments. Since it is not easy to find information a subject needs from the unstructured list, subjects were inclined to get hold on all information they may come across related to the target project. In contrast, finding needed content fragments is not an issue for the structured group. Consequently, they were able to focus on strategic issues and more conscientious about the proper length of produced flyer. It is also found that there were considerably more pollutant content fragments in the resulting flyers produced by the unstructured group than those by the structured group.

The study of the effect on document quality is limited to content selection and no consideration is paid towards document structure and layout. It is found that, on average, subjects who used the structured input composed considerably better single project flyers than those who used unstructured input. But, when it comes to two-project combined flyers, subjects who used unstructured input produced considerably better flyers that those who used structured input. However, the difference in both cases is not statistically significant.

In conclusion, organisation of input information has a strong impact on document quality in document composition. Providing the topic of a document is clearly defined, semantically structured input would have positive impact on document composition quality. Another contribution of the paper lies in the development of the X-method for evaluating content selection in document composition. In practical terms, we will incorporate structure in future Constructor development for presenting retrieval services.

Acknowledgements: We are grateful to Cécile Paris and John Colton for their comments to early drafts of the paper. Our appreciation also goes to all the people who participated in our experiment.

References

[1] Lu, S., Paris, C. and Wu, M. (2005) ‘Document modelling for customised information delivery’, Proceeding of The Tenth Australasian Document Computing Symposium (ADCS 2005), Sydney, pp.11-18.

[2] Lu, S. and Paris, C. (2007): Specifying adaptive documents: an authoring tool prototype and user studies. To appear in the Special issue on Authoring of Adaptive and Adaptable Hypermedia of the International Journal of Learning Technology, edited by Alexandra Cristea and Rosa Carro.

[3] Lu, S. and Paris, C. (2006): Authoring Content Structure for Adaptive Documents. In the Proceedings of the International Workshop on Authoring of Adaptive and Adaptable Hypermedia at the 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Dublin, Ireland, June 21-23, 2006.

[4] Mann, W.C. and Thompson, S.A. (1988) ‘Rhetorical Structure Theory: Toward a functional theory of text organisation’, Text, vol.8 (3), pp.243-281.

[5] Moore, J. and Paris, C. (1993) ‘Planning Text for Advisory Dialogues: Capturing Intentional and Rhetorical Information’, Journal of Computational Linguistics, vol.19 (4), pp.651 – 694.

[6] Nenkova, Ani and Passonneau, Rebecca J. (2004). Evaluating content selection in summarization: The pyramid method. In Proceedings of the Joint Annual Meeting of Human Language Technology (HLT) and the North American chapter of the Association for Computational Linguistics (NACL), Boston, MA.

[7] Paris, C. and Colineau, N. (2006) ‘Scifly: tailored corporate brochures on demand’, CSIRO Tech Report, No. 06/268, www.csiro.au/scifly.

[8] Paris, C., Wu, M., Vander Linden, K., Post, M. and Lu, S. (2004). Myriad: An Architecture for Contextualized Information Retrieval and Delivery, in AH2004: International Conference on Adaptive Hypermedia and Adaptive Web-based Systems. August 23-26, The Netherlands. pp. 205-214.

[9] Radev, D.R., Teufel, S., Saggion, H., Lam, W., Blitzer, J., Qi, H., elebi, A., Liu, D., Drabek, E.: Evaluation challenges in large-scale document summarization. In Proc. of the 41st Annual Meeting of the Association for Computational Linguistics. (2003) pp. 375—382.

Hybrid Bitvector Index Compression

Alistair Moffat
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria, Australia 3010
alistair@csse.unimelb.edu.au

J. Shane Culpepper
NICTA Victoria Research Laboratory,
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria, Australia 3010
shanec@csse.unimelb.edu.au

Abstract *Bitvector index representations provide fast resolution of conjunctive Boolean queries, but require a great deal of storage space. On the other hand, compressed index representations are space-efficient, but query evaluation tends to be slower than bitvector evaluation, because of the need for sequential or pseudo-random access into the compressed index lists. Here we investigate a simple hybrid mechanism that stores only a small fraction of the inverted lists as bitvectors and has no or negligible effect on compressed index size compared to the use of byte codes, but improves query processing throughput compared to both byte coded representations and entirely-bitvector arrangements.*

Keywords Index compression, bitvector, byte code, intersection algorithm.

1 Text search

Document retrieval systems typically make use of an inverted index in order to provide fast keyword-based search, see, for example, Witten et al. [1999] and Zobel and Moffat [2006]. In such an index, an inverted list is stored for each term that appears in the collection, containing the ordinal identifiers of the documents in which that term appears. To process a conjunctive Boolean query, the inverted lists corresponding to the query terms are fetched from disk, and their set intersection computed. Some forms of "ranked" query, where the ranking component consists entirely of static precomputed score elements such as PageRank, can also be handled the same way – the document collection is permuted so that document identifiers are assigned in decreasing static score order, and then "top-k ranked queries" are resolved by identifying and presenting the first k matching conjunctive Boolean answers, without any further ranking step being applied.

A wide variety of representations have been developed to store the inverted lists, each of which is a set of *document pointers* representing a subset of the integers $1 \ldots n$, where n is the number of documents in the collection. For example, a set of f elements in the range $1 \ldots n$ can be stored in $f \lceil \log_2 n \rceil$ bits using a simple binary code; and can be stored in approximately $f(1.5 + \log_2(n/f))$ bits if a Golomb or Rice code is applied to the set of d-gap differences between consecutive items in the ordered set. Witten et al. [1999] provide details of these representations. For typical document collections, in which the majority of terms appear in just a few documents, but the majority of the pointers stored are for terms that appear in many documents, compressed representations (of which Golomb and Rice codes are examples) typically save as much as 70–80% of the space that would be required by a simple binary code.

Just as there are different ways in which the inverted lists can be stored, there are also different ways in which they can be manipulated in order to compute intersections. For example, if the inverted lists are stored compressed using a Golomb code, and no auxiliary information of any kind is maintained, then the intersection of two lists of $f_1 \leq f_2$ elements requires $O(f_1 + f_2) = O(f_2)$ time, since there is no alternative to sequential decompression of both lists. On the other hand, if the lists are stored using the more space-costly fixed-width binary codes, and individual elements can be accessed and inspected in $O(1)$ time, then set intersection can be computed in $O(f_1 \log(f_2/f_1))$ time via a search-dual of the Golomb code, as described by Hwang and Lin [1972]. Other combinations of representation and intersection method are discussed by Culpepper and Moffat [2007].

When multiple lists are to be intersected, rather than just two, another dimension of choice is introduced. One possibility is to compute the intersection by per-

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007.

forming a sequence of pairwise merges in which the shortest initial list is taken as a "pivot", and repeatedly intersected against another of the initial sets, in a *set versus set*, or svs, approach. Any method for the pairwise merging of sets can be employed, including the simple linear-time sequential method. The alternative is to holistically open all of the sets simultaneously, and intersect them in an interleaved manner, with a multi-way operation being used to generate the list of document identifiers that appears in every one of the input lists. In this case, different intersection methods are possible, including *adaptive* variants that are sensitive to the interactions between the lists being joined [Demaine et al., 2000, Barbay et al., 2006].

This paper describes a hybrid bitvector representation for inverted indexes, and shows experimentally that it provides fast and compact execution of typical conjunctive Boolean queries compared to previous approaches.

2 Experimenting with intersection

In order to explore different inverted list structures and intersection algorithms, an experimental testbed was created in which a stream of conjunctive queries was processed against the index of the 426 GB gov2 collection (see `trec.nist.gov`). Words that appeared only one or twice in the $25{,}205{,}181$-document collection were assumed to not have their own index lists, and the result was an index for $19{,}783{,}975$ distinct words, with each of those lists containing on average 307.6 ordinal document numbers.

The query stream contained $27{,}004$ queries of average length 2.73 terms extracted from an operational "live" web query stream as being ones that were applicable to the experimental collection, by virtue of their having a whole-of-web top-3 answer (at the time they were issued) within the `.gov` domain. In total, $15{,}208$ distinct terms appeared in the query set, a very small fraction of the terms in the collection. Culpepper and Moffat [2007] give more details of the experimental arrangements and of the query set.

To measure CPU times, the set of index lists required by the first query in the sequence was read in to memory while the execution clock was halted; then the clock was started and the intersection of those lists computed five times, to generate an "answer" list of ordinal document numbers; then the clock was halted again while the data for the second query was fetched; and so on. At the same time as these "average over five" times were being recorded, the amount of index data (in megabytes) transferred from disk to memory during query evaluation was noted, and used later to obtain a per-query average data volume. All of the experiments were carried out on a dual 2.8 Ghz Intel Xeon with 2 GB of RAM, twelve 146 GB SCSI disks in a RAID-5 configuration, and running Debian GNU/Linux.

Figure 1, adapted from Culpepper and Moffat [2007], summarizes some of the experiments undertaken using the test harness. In that work we were interested in the extent to which a small auxiliary index in each inverted list (the annotation aux on three of the data points) allowed improved trade-offs between time and space. The auxiliary index in this method was stored uncompressed, and provided a set of access points in to the byte coded compressed inverted lists, so that the forwards searching operation required by the svs approach could be supported via pseudo-random access. A parameter k was used to balance the additional cost of the auxiliary index against the desire to keep the access points close together. Compared to a svs method implemented using a binary-coded index representation; and compared to the svs method when the data was stored compressed using standard byte codes (annotation bc, and described in more detail below), the auxiliary index approach did indeed offer an interesting compromise between speed (average query time, on the vertical axis) and space (average data volume processed, on the horizontal axis).

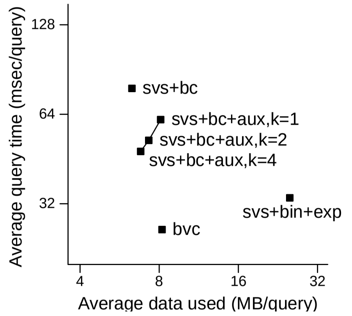


Figure 1: *Space versus CPU cost of different ways of computing conjunctive Boolean queries using a 2.8 Ghz Intel Xeon with 2 GB of RAM, adapted from Culpepper and Moffat [2007]. The horizontal axis shows the average per-query data volume required to process a set of $27{,}004$ queries; the vertical axis shows the average time taken over the same query set. Each of the marked points represents a combination of compression technique and intersection algorithm. The point marked* bvc *makes use of a bitvector representation and computes intersections using word-at-a-time* "AND" *operations.*

3 An interesting observation

Another interesting point in Figure 1 – and the basis for the further exploration that is described in this paper – is the one marked bvc. A bitvector is a very simple way of storing a set of f elements drawn from a universe $1 \ldots n$, with an n-bit array constructed in which the i th bit is set if and only if i is a member of the set. Bitvectors are a very expensive way of storing sparse sets, in which $f \ll n$. On the other hand, they have the redeeming virtue of providing $O(1)$-time lookup, meaning that a set of f_1 candidate answers can be checked against

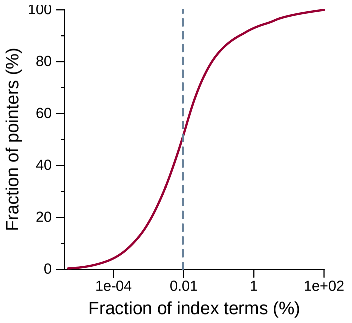


Figure 2: *Terms versus pointers: a small fraction of the terms in the collection are responsible for the great majority of the index pointers. For example, the vertical dotted line indicates that approximately* 50% *of the pointers in the index appear in the set of* 188 *inverted lists that each contain more than* $25{,}205{,}181/8 = 3{,}150{,}647$ *document identifiers.*

a second set of f_2 items in $O(f_1)$ time, once the n bits of the vector have been read. In addition, if two bitvectors are to be intersected, 32 or 64 bits at a time can be processed using whole-of-word AND operations.

The latter of these two processing modes was used to generate the bvc data point for Figure 1, and it was unsurprising that bitvectors provided fast intersection operations. What had not been expected was that the volume of data required to process the queries using a bitvector representation was also comparable with techniques that involved compression. Storing a whole index using a bitvector per index term is exorbitantly expensive, approximately 3 MB per index list in the gov2 test collection described above, making a total of nearly 60 TB for the 19 million terms indexed. But the actual queries processed tend not to be against sparse lists. Even the two-word queries in our test set typically had one word that appeared in 5% or more of the documents in the collection, and once the query reached four or more terms, on average one (or more) of the supplied query terms appeared in more than half of the documents in the collection [Culpepper and Moffat, 2007].

The bvc point in Figure 1 thus leads to an obvious question: is there some compromise arrangement that avoids the very high disk storage cost of the full bitvector index, but retains its efficiency in terms of query evaluation speed, and in terms of data volume transferred in order to process queries.

Figure 2 shows why such a hybrid approach is attractive. To construct this graph, the terms of the gov2 collection were ordered by decreasing document frequency, and then a cumulative count of pointers calculated, based on that ordering. For example, the most frequent term contributes $20{,}461{,}040$ pointers, or 0.33% of the $6{,}086{,}023{,}363$ pointers making up the index; the second most frequent term adds another $18{,}964{,}349$ pointers, and so on. Both axes of the graph are expressed as percentages, but the horizontal axis is plotted logarithmically.

What is clear from Figure 2 is that a very small fraction of the collection's terms account for a very large fraction of the index pointers. Just 0.01% of the terms account for more than 50% of the pointers, and if 1% of the terms are handled via bitvectors, more than 90% of the pointers in the index are covered.

4 A hybrid approach

The use of Golomb and Rice codes provide good compression for typical index data, in no small part because they are sensitive to the ratio f/n (see Witten et al. [1999] for a description of the Golomb code). For example, when more than around 38% of the documents in a collection contain some term, the Golomb parameter b that determines the codewords will be 1, and the effect is that of a Unary code. In such a case, the result of the coding exercise is a bitvector.

Another standard coding method for index lists (or rather, the differences between successive values in them) is via *byte codes*. In a byte code, each codeword is a multiple of eight bits long, so that all of the codewords consist of an integral number of bytes. The simplest byte code, denoted bc here, uses a single bit in each byte of the codeword to indicate whether or not this is the final byte in the value, and uses 7, or 14, or 21, and so on, bits to store the integer value in question. Using the bc approach, document pointer differences between 1 and $2^7 = 128$ are stored in a single byte; differences between 129 and $2^7 + 2^{14} = 16{,}512$ are stored in two-byte codes; and differences from $16{,}513$ to $2^7 + 2^{14} + 2^{21} = 2{,}113{,}664$ are stored as three-byte codewords. Scholer et al. [2002] examine the use of byte codes in inverted file indexing, and Brisaboa et al. [2003] and Culpepper and Moffat [2005] describe alternative byte codes with additional properties.

A byte code compression scheme was used in the bc-annotated systems shown in Figure 1. Given that the minimum codeword length in any byte code is 8 bits, and thus that a set of f elements in the range $1 \ldots n$ must consume at least $8f$ bits when coded, even after differences are taken, it is clear that a bitvector representation (in which n bits are consumed) is more compact than a byte code for any terms for which $f > n/8$. Figure 2 shows that point for the gov2 collection – the dashed vertical line separates the index lists for which a bitvector representation must be more economical (in terms of storage) from those for which a byte code is most likely to be the more economical option. As was already noted, that implies that fully half of the pointers in the index are more economically coded via a bitvector than via byte coded differences.

Storing the index list for a term t as a bitvector whenever the document frequency f_t of term t satisfies $f_t > n/8$ will thus reduce the size of a byte coded inverted index. It is also possible to allow more than the

Method	Bitvector terms	Size (GB)
Byte coded	0	7.41
Hybrid, $f_t > n/8$	188	6.97
Hybrid, $f_t > n/10$	277	7.00
Hybrid, $f_t > n/12$	367	7.07
Hybrid, $f_t > n/16$	552	7.31
Hybrid, $f_t > n/20$	775	7.67
Hybrid, $f_t > n/24$	982	8.05
Hybrid, $f_t > n/32$	1,382	8.88

Table 1: *Cost of storing a* `gov2` *index when terms appearing in more than a specified fraction of the documents are stored as bitvectors rather than as Byte coded inverted lists.*

minimum number of the inverted lists the flexibility of using a bitvector. Table 1 shows the cost, in gigabytes, of storing the `gov2` index ($19{,}783{,}975$ terms, and $6{,}086{,}023{,}363$ pointers) using byte codes alone, and then using a bitvector hybrid approach with different cutoff fractions. As can be seen, cutoffs as small as $f_t > n/32$ still result in a `gov2` index that is only a little larger than the byte coded one, and when the cutoff is $n/16$ or less, the index is smaller.

Many of these frequently-occurring terms are ones that are, at face value, unhelpful during querying. Indeed, in the past, they may well have been *stopped*, to reduce the space consumption of the index, rendering the hybrid scheme proposed here somewhat moot. However, modern retrieval and web search systems index all words and numbers, with queries such as "to be or not to be", "Dr Who", and "11 September 2001" being examples that show why complete coverage is necessary. The statistics quoted earlier in connection with the test query stream show that common words do indeed occur in typical web search queries.

Processing queries: Method One

Given the hybrid index, the obvious question now is how best to use it to resolve conjunctive Boolean queries. We experimented with two query processing approaches. In the first, queries are executed as follows:

1. All query terms are located in the vocabulary.
2. The set of terms with byte coded inverted lists, if any, are intersected using the svs approach, to yield a set of candidate answers C. If C becomes empty at any stage, then there are no answers to the query, and processing terminates.
3. The set of terms with bitvectors, if any, are intersected to get a bitvector B that represents their conjunction.
4. If there were no bitvector terms, then C can be output as the answer to the query.
5. If there were no byte coded terms, then B represents the answer, and is converted into a set of document numbers by locating all of the subscripts d for which $B[d] = 1$.
6. Otherwise, for each $d \in C$, if $B[d] = 1$, then d is output as an answer to the query.

That is, all of the byte coded terms are intersected first; then, if necessary, all of the bitvector-represented terms are intersected to get a combined bitvector; and then, if necessary, the set of candidates indicated by the byte coded terms are checked against the outcome of the bitvector merge.

Processing queries: Method Two

In the second approach to query processing, the bitvector terms are incorporated incrementally via a sequence of lookups once a set of candidate answers has been established:

1. All query terms are located in the vocabulary.
2. If there are no byte coded terms, then the bitvector terms are intersected as in Method One, and returned as the answer.
3. Otherwise, the set of terms with byte coded inverted lists are intersected using the svs approach to yield a set of candidate answers C. If C becomes empty at any stage, then there are no answers to the query, and processing terminates.
4. For each bitvector term, every element still left in C is checked against that bitvector, and retained in C only if it appears in that bitvector. If C becomes empty at any stage, then there are no answers to the query, and processing terminates.
5. When all bitvector terms have been processed, C is the list of answers.

This method avoids intersecting bitvectors once the number of viable candidate answers is small. Instead, it builds on the fact that bitvectors support random-access membership queries, and the fact that it is relatively cheap to check a small set of candidates against a bitvector by direct probing of the relevant bit positions. That is, when the size of the set of candidates C is small, which it almost certainly must be at the conclusion of the byte coded phase, it should be faster to check individual candidates against each bitvector than to AND all of the bitvectors terms together.

5 Experiments

We used the same test harness as for our previous experiments [Culpepper and Moffat, 2007], so as to be able to compare results. As was the case in that work, a set of $27{,}004$ "real" queries derived from a search log were executed, and CPU time and data transfer volume measured on a 2.8 Ghz Intel Xeon with 2 GB of RAM.

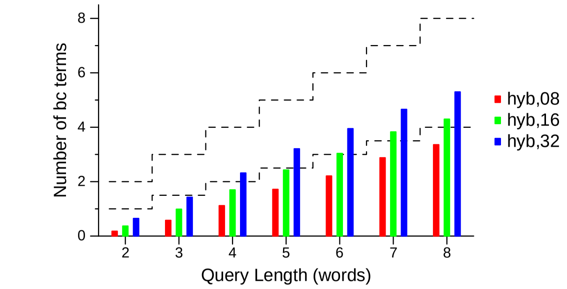


Figure 3: *Fraction of the terms in each query processed using bitvectors, for different index construction thresholds. The upper dashed line shows the total number of terms in the query; and the lower dashed line shows half of the terms. When $f_t > n/32$, on average half or more of the terms in queries of length three and greater are processed as bitvectors, either through whole-of-word* AND *operations (Method One), or via direct bit lookups (Method Two).*

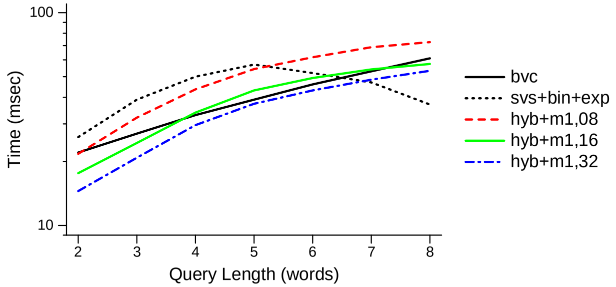


Figure 4: *Time to calculate answers to conjunctive queries using a hybrid bitvector representation, Method One processing, and three different bitvector cutoffs. The solid black line represents the pure bitvector approach, and the dotted black line shows a pure* svs *approach using a binary-coded index and exponential search. When $f_t > n/32$ is the threshold point, the hybrid approach is faster than the pure bitvector approach for all tested query lengths.*

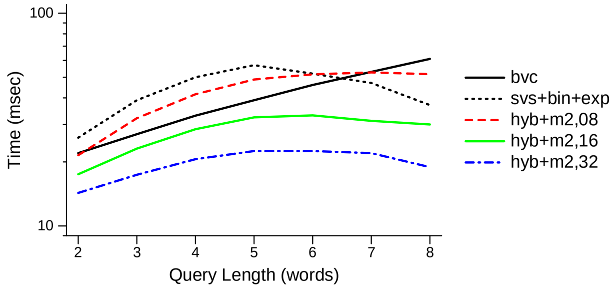


Figure 5: *Time to calculate answers to conjunctive queries using a hybrid bitvector representation, Method Two processing, and three different bitvector cutoffs. The solid black line represents the pure bitvector approach, and the dotted black line shows a pure* svs *approach using a binary-coded index and exponential search. The $f_t > n/32$ hybrid index provides significantly faster query processing than all other methods tested, across all query lengths.*

Figure 3 provides evidence in support of our hypothesis that the hybrid approach is preferable to a purely byte coded index. When the queries are analyzed based on their length in terms, the prevalence of common terms in queries becomes apparent. For example, even if as few as 552 common terms are stored as bitvectors rather than as byte coded lists (denoted in the graph as hyb,16), on average half of the query terms in queries of five or more words are stored as bitvectors, and handled more efficiently than would be the case using a pure svs+bc approach. Obviously, the larger the number of common terms stored as bitvectors, the greater the fraction of any particular query that is likely to be able to be handled via bitvector manipulations.

Figures 4 and 5 show measured querying cost across the range of query lengths in the test sequence, using Method One and Method Two processing respectively. In both Figure 4 and Figure 5 the solid black reference line indicates the cost of using a purely bitvector index to process the queries; and the dotted black line shows the speed attained by a binary coded index and svs processing using exponential search to skip forwards through the lists.

Looking at Figure 4 if the index uses bitvector form for terms that satisfy $f_t > n/32$, and a byte coded representation for all other lists, then the Method One hybrid approach is always better than the pure bitvector approach by a slender margin. Note, however, that the query cost does tend to increase as terms are added to the query, and that for long queries an uncompressed svs mechanism can be faster.

Figure 5 then shows the additional usefulness of Method Two processing. Short queries are handled equally quickly as with Method One, and long queries are handled nearly three times faster, within the same amount of index space. In addition, the hybrid bitvector approach, together with Method Two processing, improves completely on the svs+bc+aux,k=4 method described in our previous paper. We also explored combining the auxiliary-indexed byte coded lists with hybrid bitvectors, but got no additional gain. That is, the auxiliary index of Culpepper and Moffat [2007] provides faster (than strictly sequential) processing of the long inverted lists, but storing them as bitvectors is another – apparently even more efficient – way of tapping the same underlying opportunity.

Finally, Figure 6 (using the same 2.8 Ghz Intel Xeon with 2 GB of RAM) revisits the speed/data tradeoff graph that was shown in Figure 1, and illustrates the improvement attained by the hybrid bitvector storage and processing strategies. Method Two, shown in the graph, is faster than Method One, and quite comprehensively outperforms the previous approaches, including the svs+bc+aux auxiliary index method [Culpepper and Moffat, 2007].

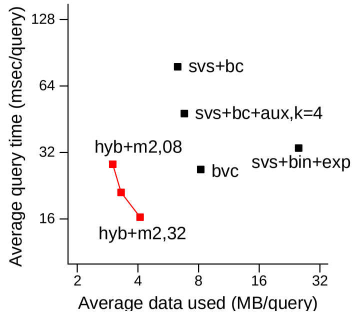


Figure 6: *Space versus CPU cost of different ways of computing conjunctive Boolean queries, showing the gain in speed attained by the hybrid bitvector approach (Method Two). The horizontal axis shows the average per-query data volume required to process a set of 27,004 queries; the vertical axis shows the average time taken over the same query set. Note the shift in both axes compared to Figure 1.*

6 Related work

Previous approaches to providing fast Boolean conjunction in inverted indexes have focused on the provision of internal structures to facilitate pseudo-random access via skipping or similar arrangements [Moffat and Zobel, 1996, Strohman and Croft, 2007, Culpepper and Moffat, 2007]; or on providing compression regimes that allow pointers to be skipped with only partial decompression being necessary [Anh and Moffat, 1998, Scholer et al., 2002, Anh and Moffat, 2006]. Other methods for fast intersection – including adaptive multi-way approaches – have been described in terms of uncompressed binary representations, so that random access operations can be performed [Demaine et al., 2000, 2001, Barbay and Kenyon, 2002, Gupta et al., 2006, Barbay et al., 2006, Sanders and Transier, 2007]. Our work here – which, as noted, builds on an observation made in a previous paper [Culpepper and Moffat, 2007] – is, we believe, the first to fully balance random-access processing of candidate answer documents against frequently occurring terms, with suitably compressed representations for all terms.

7 Conclusion

We have shown that a relatively simple combination of techniques allows fast calculation of Boolean conjunctions within a surprisingly small amount of data transferred. This approach exploits the observation that queries tend to contain common words, and that representing common words via a bitvector allows random access testing of candidates, and, if necessary, fast intersection operations prior to the list of candidates being developed. By using bitvectors for a very small number of terms that (in both documents and in queries) occur frequently, and byte coded inverted lists for the balance, we have reduced both querying time and also query-time data-transfer volumes.

The techniques described here are not, of course, applicable to other more powerful forms of querying. For example, index structures that support phrase and proximity queries have a much more complex structure, and are not amenable to storage (in their full form) using bitvectors. Nevertheless, there may be scope for evaluation regimes that make use of preliminary conjunctive filtering before a more detailed index is consulted, in which case the structures described here would still be relevant. We plan to explore this option as we continue our investigation into hybrid bitvector structures.

Acknowledgment The query log was supplied by Microsoft Search.

References

V. N. Anh and A. Moffat. Improved word-aligned binary compression for text indexing. *IEEE Transactions on Knowledge and Data Engineering*, 18(6):857–861, June 2006.

V. N. Anh and A. Moffat. Compressed inverted files with reduced decoding overheads. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proceedingsof the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*, pages 290–297, Melbourne, Australia, August 1998. ACM Press, New York.

J. Barbay and C. Kenyon. Adaptive intersection and t-threshold problems. In D. Eppstein, editor, *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2002)*, pages 390–399, January 2002.

J. Barbay, A. López-Ortiz, and T. Lu. Faster adaptive set intersections for text searching. In C. Àlvarez and M. J. Serna, editors, *Experimental Algorithms, 5th International Workshop (WEA 2006)*, volume 4007 of *LNCS*, pages 146–157. Springer, May 2006.

N. R. Brisaboa, A. Fariña, G. Navarro, and M. F. Esteller. (S, C)-dense coding: An optimized compression code for natural language text databases. In M. A. Nascimento, editor, *Proceedings of the 10th International Symposium on String Processing and Information Retrieval (SPIRE 2003)*, volume 2857 of *LNCS*, pages 122–136, Manaus, Brazil, October 2003. Springer.

J. S. Culpepper and A. Moffat. Enhanced byte codes with restricted prefix properties. In M. P. Consens and G. Navarro, editors, *Proceedings of the 12th International Symposium on String Processing and Information Retrieval (SPIRE 2005)*, volume 3772 of *LNCS*, pages 1–12, Buenos Aires, Argentina, November 2005. Springer. URL http://dx.doi.org/10.1007/11575832_1.

J. S. Culpepper and A. Moffat. Compact set representation for information retrieval. In N. Ziviani and R. Baeza-Yates, editors, *Proceedings of the 14th International Symposium on String Processing and Information Retrieval (SPIRE 2007)*, volume 4726 of *LNCS*, pages 137–148, Santiago, Chile, October 2007. Springer. URL http://dx.doi.org/10.1007/978-3-540-75530-2_13.

E. D. Demaine, A. López-Ortiz, and J. I. Munro. Adaptive set intersections, unions, and differences. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*, pages 743–752, January 2000.

E. D. Demaine, A. López-Ortiz, and J. I. Munro. Experiments on adaptive set intersections for text retrieval systems. In *Proceedings of the 3rd Workshop on Algorithm Engineering and Experiments (ALENEX 2001)*, volume 2153 of *LNCS*, pages 91–104. Springer, January 2001.

A. Gupta, W.-K. Hon, R. Shah, and J. S. Vitter. Compressed dictionaries: Space measures, data sets, and experiments. In C. Àlvarez and M. J. Serna, editors, *Proceedings of the 5th International Workshop on Experimental Algorithms (WEA 2006)*, volume 4007 of *LNCS*, pages 158–169. Springer, May 2006.

F. K. Hwang and S. Lin. A simple algorithm for merging two disjoint linearly ordered list. *SIAM Journal on Computing*, 1:31–39, 1972.

A. Moffat and J. Zobel. Self-indexing inverted files for fast text retrieval. *ACM Transactions on Information Systems*, 14(4):349–379, 1996.

P. Sanders and F. Transier. Intersection in integer inverted indices. In *Proceedings of the 9th Workshop on Algorithm Engineering and Experiments (ALENEX 2007)*, pages 71–83. SIAM, January 2007.

F. Scholer, H. E. Williams, J. Yiannis, and J. Zobel. Compression of inverted indexes for fast query evaluation. In M. Beaulieu, R. Baeza-Yates, S. H. Myaeng, and K. Järvelin, editors, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 222–229, Tampere, Finland, August 2002. ACM Press, New York.

T. Strohman and W. B. Croft. Efficient document retrieval in main memory. In C. L. A. Clarke, N. Fuhr, N. Kando, W Kraaij, and A. P. de Vries, editors, *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 175–182, Amsterdam, The Netherlands, July 2007. ACM Press, New York.

I. H. Witten, A. Moffat, and T. A. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, San Francisco, second edition, 1999.

J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Computing Surveys*, 38(2):1–56, 2006.

On the distribution of user persistence for rank-biased precision

Laurence A. F. Park

Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010 Australia

lapark@csse.unimelb.edu.au

Yuye Zhang

NICTA Victoria Research Laboratory,
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010 Australia

zhangy@csse.unimelb.edu.au

Abstract *Rank-biased precision (RBP) is a new method of information retrieval system evaluation that takes into account any uncertainty due to incomplete relevance judgements for a given document and query set. To do so, RBP uses a model of user persistence. In this article, we will present a statistical analysis of the RBP user persistence model to observe how the user persistence value affects the user persistence distribution. We also provide a method of fitting data from existing users to the persistence model, in order to compute their persistence value. Using the Microsoft MSN query log, we were able to demonstrate a typical distribution of the user persistence value and show that it closely resembles a reverse lognormal distribution, with a mean of* $p = 0.78$.

Keywords Evaluation, rank-biased precision, persistence distribution

1 Introduction

To evaluate an information retrieval system, the documents retrieved by the system are compared to a list of relevance judgements for each (document, query) pair using some evaluation metric. Therefore, the evaluation requires manually judging each pair.

As information storage and retrieval algorithms improve, and computer processing power and storage grows, so too does the expected number of documents indexed by a retrieval system. A problem that is encountered by researchers in the information retrieval field is the manual judgement of each (document, query) pair when faced with large document collections. A simple method of dealing with these large sets is to manually judge only a subset of the documents for each query, where the subset hopefully contains most of the documents relevant to that query. When evaluating a system, if a document is retrieved that was not in the judged set, it can be assumed irrelevant to the query. Although this method still provides reliable results, overall accuracy has been compromised due to uncertainty stemming from the unjudged documents.

A new method of information retrieval evaluation called rank-biased precision (RBP) [3] deals with unjudged documents by offering uncertainty in the evaluation. Therefore, the evaluation score provides a range covering the evaluation scores that would have been obtained if the unjudged documents were relevant or irrelevant.

RBP evaluation is based on a user persistence model that requires the choice of a user persistence value before the evaluation can take place. In this article, we examine the statistical properties of the user persistence model and the effect of choosing a certain user persistence value. We also examine how we can deduce the persistence value to suit a desired audience, and hence model that audience. This article makes the following contributions:

- a statistical analysis of the properties of the user persistence model and effect of the user persistence value (p).
- a method of computing the persistence value (p) from a query log in order to model a set of users
- an analysis showing that the user persistence (p) is a reverse lognormal distribution when modelled over a large audience.

The article will proceed as follows: in section 2 we will discuss the method of rank-biased precision and its associated user persistence model. In section 3 we will analyse the user persistence distribution and examine the effect of changing the user persistence value (p). Section 4 describes a method of modelling the user persistence value when given an appropriately detailed query log. Finally, in section 5 we will examine the

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007.

distribution of the user persistence value (p) computed from a query log provided by Microsoft.

2 Evaluation of large documents collections

In this section, we examine a new method of evaluating retrieval systems called rank-biased precision (RBP) that allows us to easily account for uncertainty in relevance judgements. We begin the section by outlining the method in which relevance judgements are obtained for large document collections and examine the problems associated to existing popular retrieval system metrics that are induced by the judgement process. We then introduce RBP and show how it is more suited to dealing with the uncertainty in modern document collection relevance judgements.

2.1 Obtaining relevance judgements

The ideal method of evaluating a retrieval system is to obtain a document set, a set of queries and a relevance judgements of every document for each query. Relevance judgements are assigned to each (document, query pair) manually, implying that human judges must examine every document for each query and assign a relevance score (usually 1 for relevant and 0 for irrelevant). Once these scores are obtained, the system in question is used to rank each document for each query and the results are compared to the relevance judgements using some pre-defined metric.

Modern retrieval experiments are performed on document collections containing millions of documents, so unfortunately, human relevance judgements are not possible for every (document, query pair). To obtain an estimate of the most relevant documents, TREC[1] have implemented a pooling method of document evaluation in an attempt to obtain the best estimate of relevant documents per query [4]. For each of the retrieval systems taking part in TREC, the set of top ranked documents for a given query are placed into a pool. Therefore, the pool for a query will contain the set of documents that have been highly ranked by each of the retrieval systems. The pool is then considered as a set of candidate documents that should contain most of the documents relevant to each query. Each of these documents are then manually judged and the documents that do not appear in the pool are considered irrelevant to the query. Unfortunately, the pooling method places great importance on the initial set of retrieval systems that are used, since any documents that are not added to the pool (that is, not highly ranked by any retrieval system) are considered irrelevant.

2.2 Uncertainty in relevance judgements

Rather than assuming that unjudged documents are irrelevant, we should simply take into account this uncertainty during the system evaluation process. For example, if a query retrieves documents that all have associated relevance judgements, we should be able to precisely evaluate the system, but if one or more documents do not have relevance judgements, then the evaluation should contain an associated error margin depicting the uncertainty.

Unfortunately, many of the popular information retrieval metrics do not allow for this uncertainty. It has been shown that scores produced by retrieval metrics such as mean average precision (MAP) and bpref [1] can provide drastically different scores when uncertainty is introduced due to their dependence on the number of documents relevant to each query. Consequently, such metrics are unsuited for handling uncertainty within the retrieved results in these instances.

2.3 Rank-biased precision

A new metric called rank-biased precision (RBP) [2, 3] has been designed to take into account uncertainty in relevance judgements. RBP has shown to provide error margins that converge as the uncertainty reduces.

RBP is designed around the model that a user examining a list of retrieved documents will start from the top ranked document and when examining each document, will proceed to the next document with a probability p, or finish the search with probability $1 - p$. The score provided to the system increases if a user examines a relevant document, therefore the RBP is computed as the sum of the probability of examining each relevant document:

$$RBP(p) = (1 - p) \sum_{i=1}^{\infty} r_i p^{(i-1)} \qquad (1)$$

where $r_i \in [0, 1]$ is the relevance judgement of the ith ranked document, and the $(1 - p)$ factor is used to scale the RBP within the range $[0, 1]$. The probability of a user examining the next document is also the *persistence* of the user. We can see that a user with low persistence (p close to zero) is not likely to examine past the first document, while a user with a high persistence value (p close to 1) is likely to examine many documents.

We will now provide an example of how RBP is used to obtain an intuition of how uncertainty is dealt with. Given a user with persistence of $p = 0.5$, if a given system returns a ranked document list such that the ranked document have the associated relevance judgements $(1, 1, 0, 1, ?, 0, 0, 1)$, where 1 denotes relevance, 0 denotes irrelevance and ? denotes not judged, then the RBP is computed as: $(1 - 0.5) \times (0.5^0 + 0.5^1 + 0.5^3 + 0.5^7) = 0.816$. By taking into account the uncertain relevance judgements, we can compute the uncertainty in the RBP as: $(1 - 0.5) \times (0.5^4 + \sum_{i=9}^{\infty} 0.5^{i-1}) = (1 - 0.5) \times 0.5^4 + 0.5^8 = 0.0352$. Implying that the RBP lies within the bounds $[0.816, 0.852]$, due to the uncertainty produced by the pooling process. We can see from these steps that as the number of relevance

[1] http://trec.nist.gov

judgements increase, the number of uncertain relevance judgements decrease and hence the uncertainty in the RBP decreases.

The RBP metric relies on the user persistence model and hence on the choice of the user persistence (p). In the remainder of this article we will examine properties of this persistence distribution and the effect of p. We will also examine how to choose p, when given a set of users' statistics.

3 The distribution of user persistence

Once a user has received a ranked list of search results, the typical behaviour is to scan from the top of the list to the bottom of the list in the hope that a document relevant to his or her information need is found. If a document appears to be relevant, based on its title or snippet, the user will open the document and examine it further. When the user believes that there will be no more relevant documents further down the ranked list, the user will finish examining the list. We can treat the depth at which the user stops examining the ranked list as the user's persistence. For example, a user that only examines the first two documents will have a low persistence value, while a user that examines the first twenty documents will have a higher persistence value.

The rank-biased precision evaluation metric uses the assumption that a user has a specific persistence p, which is the probability of examining the next document in the ranked list. Therefore, if we begin from the top of the list, the probability of examining the ith document is:

$$P(E = i|p) = p^{i-1} \tag{2}$$

Given that the probability of examining the next document is p, we can deduce that the probability of not examining further documents to be $1-p$. Therefore, the probability that a user examines the first i documents in the list but no more is:

$$P(L = i|p) = p^{i-1}(1-p) \tag{3}$$

We will now refer to this probabilistic distribution as the *persistence distribution*. Figure 1 provides an example of the persistence distribution for $p = 0.5$, 0.8 and 0.9. Throughout this article, we investigate the properties of the persistence distribution and examine how to select the persistence value p for a given user population. To obtain further understanding of the persistence distribution properties, we will derive its mean and variance, and examine the shape of the distribution.

The mean value of the persistence distribution (or the expected number of documents examined), derived in appendix A.1, is:

$$\mu_L(p) = \frac{1}{1-p} \tag{4}$$

where $\mu_L(p)$ is the expected number of documents examined before leaving the search list ($\mathbb{E}[L]$), given a user's persistence (p).

Persistence (p)	$\mu_L(p)$	$\sigma_L(p)$
0	1	0
0.5	2	1.414
0.666	3	2.449
0.75	4	3.464
0.8	5	4.472
0.833	6	5.477
0.857	7	6.481
0.875	8	7.483
0.888	9	8.485
0.9	10	9.487
0.95	20	19.494
0.98	50	49.497
0.99	100	99.499
1.0	∞	∞

Table 1: A list of user persistence values (p) and the associated expected number of pages examined in a ranked list of search results ($\mu_L(p)$) and standard deviation ($\sigma_L(p)$).

The standard deviation ($\sigma_L(p)$) of the persistence distribution (or the expected difference in the number of documents examined and the expected number of documents examined) derived in appendix A.2, is:

$$\sigma_L(p)^2 = \frac{p}{(1-p)^2} \tag{5}$$

where $\sigma_L(p)^2$ is the variance associated to the number of documents examined before leaving the search list, when given a user's persistence (p). Table 1 provides a list of user persistence values and the associated mean and standard deviation.

From the equations of mean and standard deviation, we can see that the standard deviation approaches the mean as the persistence value approached infinity:

$$\mu_L(p) = \lim_{p\to\infty} \sigma_L(p) \tag{6}$$

This implies that it is harder to predict the number of pages examined by more persistent users (users who are expected to examine many documents), due to the high standard deviation.

4 Modelling user persistence

To examine persistence, we must examine how many documents a user will examine before leaving the query.

4.1 Satisfied and unsatisfied queries

When examining the behaviour of a search engine user, we find that by observing the rank of the final document inspected does not necessarily provide us with a good estimate of the user's persistence. For example if a user, who was very persistent, found the relevant document located at rank one, they would have no reason to examine the list further. On the other hand if a very persistent user was presented with a list with no relevant documents, that user would examine many of the documents in the list. A user with a low persistence

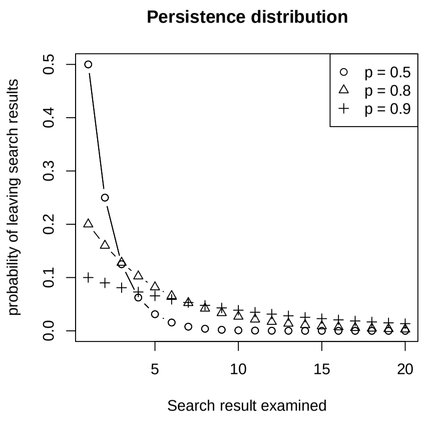


Figure 1: The persistence distribution for persistence values, $p = 0.5$, 0.8 and 0.9 on the first twenty ranked search results.

value would examine only a few documents, regardless of the relevance of those documents. So we can see from this example that if we did not take into account the relevance of the returned documents, any modelling would be biased towards a low persistence value.

As an attempt to remove bias caused by a user who stops examining documents because a relevant document is found, we split the set of queries into two. The first set, labelled *Unsatisfactory*, contains all of the queries which did not satisfy the user's information need before the user's persistence wore out. The second set, labelled *Satisfactory*, contains all queries where the user found a relevant document and hence finished scanning the list even though the user's persistence had not worn out.

From the Unsatisfactory set, we can find where the users persistence had run out by observing the last document the user examined for each query. We can compute the users persistence by fitting the model in equation 3, which involves computing the p that provides the maximum likelihood:

$$L(p) = \prod_{i \in U} p^{i-1}(1-p) \tag{7}$$

where U is the Unsatisfactory set of queries.

Computing the persistence from the Satisfactory set is not as simple , since we do not have a measure of where the user's persistence ended, and their information need was satisfied. The only information we have is how far down the results list the user examined until a relevant document was found. Therefore, in order to compute the persistence, we need to perform survival analysis.

The survival function shows that probability of an event occurring after a certain point, in our case it is the probability of the users persistence wearing out after the ith document ($P(L > i)$). To compute the survival function, we must first compute the cumulative probability function ($P(L \leq i)$). The probability that the user gives up at or before rank i is:

$$\begin{aligned} P(L \leq i) &= \sum_{x=1}^{i} p^{x-1}(1-p) && (8)\\ &= (1-p)\sum_{x=1}^{i} p^{x-1} && (9)\\ &= (1-p)\left(\frac{1-p^i}{1-p}\right) && (10)\\ &= 1 - p^i && (11) \end{aligned}$$

Therefore the survival function is:

$$\begin{aligned} P(L > i) &= 1 - P(L \leq i) && (12)\\ &= 1 - (1 - p^i) && (13)\\ &= p^i && (14) \end{aligned}$$

4.2 Maximum likelihood estimation of persistence

If we assume that for each session (where the user issues one or more queries to the retrieval system), there is a set of queries that satisfy the user and a set that do not, then we can compute the persistence of the user using the likelihood function:

$$\begin{aligned} L(p) &= \prod_{i \in \mathrm{U}} P(L = i|p) \prod_{j \in \mathrm{S}} P(L > i|p) && (15)\\ &= \prod_{i \in \mathrm{U}} p^{i-1}(1-p) \prod_{j \in \mathrm{S}} p^j && (16) \end{aligned}$$

where the first product is associated to the set of unsatisfactory queries U and the second product is associated to the set of satisfactory queries S.

Therefore, given a set of queries from a user and the last document examined for each query, the p associated to the user provides the maximum value for the likelihood function $L(p)$. To obtain the maximum, we must find where the derivative of the likelihood function is zero.

Rather than work with the products in this likelihood function, we are able to work with sums in the log-likelihood function, since $\log()$ is a monotonically increasing function (shown in appendix B.1):

$$\begin{aligned} l(p) &= \log(L(p)) = \log\left(\prod_{i \in U} p^{i-1}(1-p) \prod_{j \in S} p^j\right)\\ &= \sum_{i \in U}(i-1)\log(p) + \sum_{i \in U}\log(1-p) +\\ &\quad \sum_{j \in S} j \log(p) \end{aligned}$$

By differentiating with respect to p, we are able to locate the turning point of $l(p)$ and hence derive the equation for the most likely value of the persistence p (shown in appendix B.2):

$$p = \frac{\sum_{i \in U}(i-1) + \sum_{j \in S} j}{\sum_{i \in U \cup S} i} \tag{17}$$

where U is the set ranks of final documents examined in each search when the document was unsatisfactory to the user's information need, and S is the set ranks of final documents examined where the documents were satisfactory to the user's information need. Therefore $U \cap S = \emptyset$, the empty set.

5 Experiments

In the previous section we demonstrated how to model a set of users based on their usage history. In this section we use the modelling methods we derived to compute the persistence values for a set of typical Web search engine users. We begin by describing the data that is used and follow with the modelling of the data.

5.1 MSNSearch query log statistics

The Microsoft MSNSearch query log [5] consists of 5,684,599 user sessions, where each session consists of one or more queries to the Microsoft search engine, from a particular user. To examine the distribution of the persistence value across the set of users, we first must identify each user. Unfortunately, the queries have been anonymized, therefore we must assume that each session is associated to a unique user.

Additionally, each query is associated to a set of clickthrough information, which shows rankings of the search results that were been clicked on for that query instance. So for any given query, there may be zero or many associated clickthroughs.

To model the persistence of individual users, we obtained an estimate of p for each session (correlating to a unique user) using the maximum likelihood method. A histogram of the session lengths is shown in figure 2. We can see from the log scale for frequency, that there is an exponential decay in the frequency of sessions of length n as n increases. This implies that there is a very large proportion of sessions that contain only one query.

5.2 User modelling process

We showed earlier that we are able to compute the maximum likelihood estimate of the user persistence (p) from sampling the rank of lowest ranked document examined per query. Therefore, to compute a single user's persistence from the MSN query log, we must:

- select the the associated session
- compile the set C containing the lowest ranked clickthrough for each query in the session
- split the set C into the set of unsatisfactory queries U and satisfactory queries S

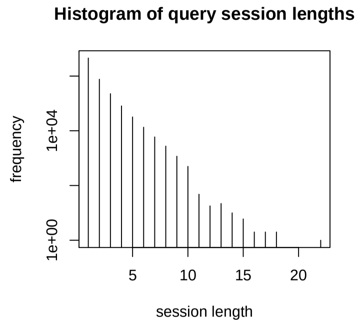


Figure 2: A histogram showing the distribution of query session lengths, using a log scale for the frequency. The linearity of the histogram shows that the frequency decreases exponentially as the session length increases.

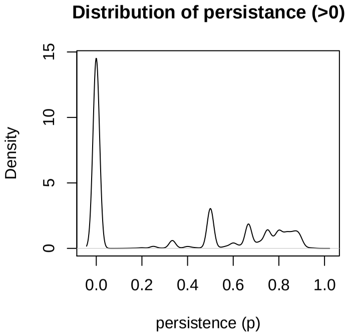


Figure 3: The kernel density estimate of the persistence (p) distribution for all sessions, assuming that all queries were unsatisfactory.

- use U and S to compute the maximum likelihood value of the user persistence from equation 17

Once this is done for each user, we are able to plot the distribution of the fitted persistence values across all users.

5.3 All queries unsatisfactory

We first examined the distribution of p using the assumption that all of the the queries in each session were unsatisfactory. The resulting distribution is shown in figure 3. The distribution shows a large peak at $p = 0$, implying that the majority of users examine only the top ranked document before beginning a new search.

This result maybe an artifact of the many sessions of length one found in the query logs, where the session

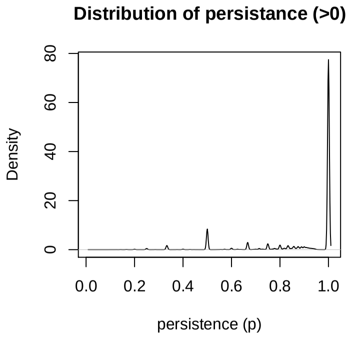


Figure 4: The kernel density estimate of the persistence (p) distribution for all sessions, assuming that all but the last query for the session was unsatisfactory.

shows the user examining only the top ranked document.

5.4 Last queries satisfactory

If a user examines only one document and finishes the session, we have to ask why the user did not issue another search. We may imply from this that the user was satisfied with the search results. From this we can assume that each session is finished with the user being satisfied with the search results. Therefore, we have generated a distribution for p where all but the last queries of each session are unsatisfactory and the last query is satisfactory. The distribution of p is shown in figure 4.

We can see that the distribution now has a large proportion of users with $p = 1$. We can easily show that this artifact is also due to the sessions of length one. If we assume that the last query in each session is satisfactory, the sessions of length one only contain one satisfactory query and no unsatisfactory queries. Therefore the maximum likelihood estimate of p reduces to:

$$p = \frac{i}{i} = 1 \tag{18}$$

where i is the lowest ranked document examined for the satisfactory query. With no unsatisfactory queries, we are unable to compute p, since we have no cases where the users persistence has worn out. Therefore to obtain a better estimate of p, we will compute its distribution using all session of length two or greater. The distribution is shown in figure 5. This figure, we can see in three large spikes at approximately $p = 0.5$, 0.66 and 0.75. Amongst these spikes, we can see a smooth curve following a reverse lognormal distribution that peaks at around $p = 0.9$.

To obtain an estimate of p for a user, we need instances of many issued queries and the resulting last document examined for each query. Therefore the smaller the number of queries per session, the rougher the estimate of p obtained for the user. By choosing only those sessions that contain more than n queries, we will be able to obtain a better estimate of p for the chosen sessions, but we will also be sampling a subset of the population. Table 2 shows the number of sessions used when constrained to a minimum session lengths.

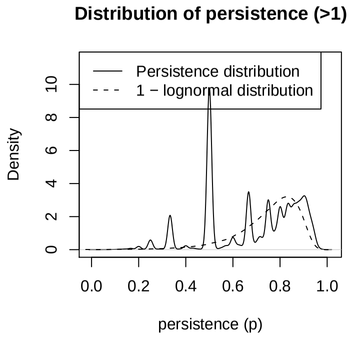


Figure 5: The kernel density estimate of the persistence (p) distribution for all sessions containing at least two search attempts, assuming that all but the last query for the session was unsatisfactory.

Session length	Total sessions
> 0	5,684,599
> 1	1,675,949
> 2	679,111
> 3	321,272

Table 2: Total number of sessions in the query log with a certain session length.

To examine the effect of using only sessions of length greater than two and three on the user persistence, we have provided the distributions for these data subsets in figures 6 and 7. We can see that as we increase the session length threshold, the resulting persistence distribution becomes smoother and becomes more like a reversed lognormal distribution. We can also see the large spike that was at 0.5 in figure 5, move towards $p = 0$ as we limit our analysis to longer sessions. This spike is to due to the set of sessions where the user has clicked on the top ranked result only, providing a p of $1/\text{session length}$.

It is interesting to note that the fitted reversed lognormal distribution provides a mean close to $p = 0.78$ and standard deviation that provides a 95% confidence interval of $(0.387, 0.920)$.

6 Conclusion

Rank-biased precision (RBP) is a new method of information retrieval system evaluation that takes into

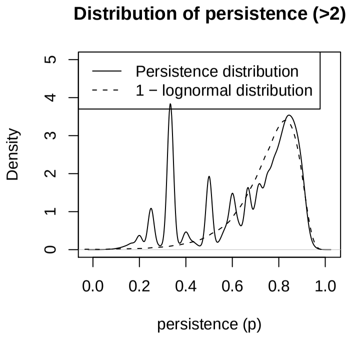


Figure 6: The kernel density estimate of the persistence (p) distribution for all sessions containing at least three search attempts, assuming that all but the last query for the session was unsatisfactory.

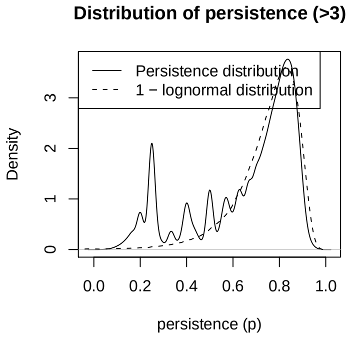


Figure 7: The kernel density estimate of the persistence (p) distribution for all sessions containing at least four search attempts, assuming that all but the last query for the session was unsatisfactory.

account any uncertainty due to incomplete relevance judgements for a given document and query set. To do so, RBP uses a model of user persistence.

In this article, we presented a statistical analysis of the user persistence model to observe how the user persistence value affects the user persistence distribution. We followed this with a method of modelling the user persistence value from user statistics.

Using the Microsoft MSN query log, we were able to demonstrate a typical distribution of the user persistence value and show that it closely resembles a reverse lognormal distribution, with a mean of $p = 0.78$.

References

[1] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32, New York, NY, USA, 2004. ACM Press.

[2] Alistair Moffat, William Webber and Justin Zobel. Strategic system comparisons via targeted relevance judgments. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 375–382, New York, NY, USA, 2007. ACM Press.

[3] Alistair Moffat and Justin Zobel. Rank-biased precision for measurement of retrieval effectiveness. *Under review*, 2007.

[4] Ellen M. Voorhees and Donna K. Harman (editors). *The Ninth Text REtrieval Conference (TREC-9)*, Gaithersburg, Md. 20899, December 2000. National Institute of Standards and Technology Special Publication 500-249, Department of Commerce, National Institute of Standards and Technology.

[5] Yuye Zhang and Alistair Moffat. Some observations on user search behavior. In *Proc. 11th Australasian Document Computing Symposium*, pages 1–8, 2006.

A Derivation of the persistence distribution statistical properties

In this section we derive the mean and variance of the persistence distribution. To perform the derivations, we use the equation:

$$\sum_{i=0}^{\infty} p^i = \frac{1}{1-p} \tag{19}$$

A.1 Expected pages examined

The expected number of pages examined is simply the sum of the page rank times the probability of examining the page:

$$\begin{aligned}
\mathbb{E}[L] &= \sum_{i=1}^{\infty} i p^{i-1}(1-p) \\
&= (1-p)\sum_{i=1}^{\infty} i p^{i-1} \\
&= (1-p)\sum_{i=1}^{\infty} \frac{d(p^i)}{dp} \\
&= (1-p)\frac{d(\sum_{i=1}^{\infty} p^i)}{dp} \\
&= (1-p)\frac{d(\sum_{i=0}^{\infty} p^i - p^0)}{dp} \\
&= (1-p)\frac{d(1/(1-p)-1)}{dp} \\
&= (1-p)\frac{1}{(1-p)^2} \\
&= \frac{1}{(1-p)}
\end{aligned}$$

A.2 Variance of pages examined

The variance is the mean deviation of the pages examined from the expected number of pages examined. This can be simplified to:

$$\sigma^2 = \mathbb{E}[L^2] - \mathbb{E}[L]^2 \tag{20}$$

where σ^2 is the variance. Therefore, to compute the variance, we must first obtain the value of $\mathbb{E}[L^2]$:

$$\begin{aligned}
\mathbb{E}[L^2] &= \sum_{i=1}^{\infty} i^2 p^{i-1}(1-p) \\
&= (1-p)\sum_{i=1}^{\infty} i^2 p^{i-1} \\
&= (1-p)\sum_{i=1}^{\infty}\left[\frac{d^2(p^{i+1})}{dp^2} - \frac{d(p^i)}{dp}\right] \\
&= (1-p)\left[\frac{d^2(\sum_{i=1}^{\infty} p^{i+1})}{dp^2} - \frac{d(\sum_{i=1}^{\infty} p^i)}{dp}\right] \\
&= (1-p)\left[\frac{d^2(\sum_{j=2}^{\infty} p^j)}{dp^2} - \frac{d(\sum_{i=1}^{\infty} p^i)}{dp}\right] \\
&= (1-p)\left[\frac{d^2(\sum_{j=0}^{\infty} p^j - p^0 - p^1)}{dp^2} - \right. \\
&\qquad \left. \frac{d(\sum_{i=0}^{\infty} p^i - p^0)}{dp}\right] \\
&= (1-p)\left[\frac{d^2(1/(1-p) - 1 - p)}{dp^2} - \right. \\
&\qquad \left. \frac{d(1/(1-p) - 1)}{dp}\right] \\
&= (1-p)\left[\frac{d(1/(1-p)^2 - 1)}{dp} - \frac{1}{(1-p)^2}\right] \\
&= (1-p)\left[\frac{2}{(1-p)^3} - \frac{1}{(1-p)^2}\right] \\
&= \frac{2}{(1-p)^2} - \frac{1}{(1-p)}
\end{aligned}$$

where $j = i + 1$. Using the value of $\mathbb{E}[L^2]$, we can compute the variance (σ^2) using the mean shown in appendix A.1:

$$\begin{aligned}
\sigma^2 &= \mathbb{E}[L^2] - \mathbb{E}[L]^2 \\
&= \frac{2}{(1-p)^2} - \frac{1}{(1-p)} - \frac{1}{(1-p)^2} \\
&= \frac{1}{(1-p)^2} - \frac{1}{(1-p)} \\
&= \frac{1}{(1-p)^2} - \frac{1-p}{(1-p)^2} \\
&= \frac{p}{(1-p)^2}
\end{aligned}$$

B Derivation of maximum likelihood value of persistence

B.1 Simplification of log-likelihood

The log-likelihood function (equation 16) is simplified using the following process:

$$\begin{aligned}
l(p) = \log(L(p)) &= \log\left(\prod_{i\in U} p^{i-1}(1-p)\prod_{j\in S} p^j\right) \\
&= \sum_{i\in U}\log\left(p^{i-1}(1-p)\right) + \sum_{j\in S}\log\left(p^j\right) \\
&= \sum_{i\in U}\log\left(p^{i-1}\right) + \sum_{i\in U}\log\left(1-p\right) + \\
&\quad \sum_{j\in S}\log\left(p^j\right) \\
&= \sum_{i\in U}(i-1)\log\left(p\right) + \sum_{i\in U}\log\left(1-p\right) + \\
&\quad \sum_{j\in S} j\log\left(p\right)
\end{aligned}$$

B.2 Maximum of log-likelihood function

To obtain the maximum p for the given log-likelihood function $l(p)$, we must first find the derivative of the log-likelihood function:

$$\frac{dl(p)}{dp} = \frac{\sum_{i\in U}(i-1)}{p} + \frac{\sum_{i\in U} -1}{1-p} + \frac{\sum_{j\in S} j}{p} \tag{21}$$

By equating the derivative to zero and solving for p, we obtain the equation to compute the maximum likelihood of p:

$$\begin{aligned}
\Rightarrow \frac{\sum_{i\in U} 1}{1-p} &= \frac{\sum_{i\in U}(i-1) + \sum_{j\in S} j}{p} \\
p\sum_{i\in U} 1 &= (1-p)\left(\sum_{i\in U}(i-1) + \sum_{j\in S} j\right) \\
p\sum_{i\in U} 1 + p\left(\sum_{i\in U}(i-1) + \sum_{j\in S} j\right) &= \sum_{i\in U}(i-1) + \sum_{j\in S} j \\
p\left(\sum_{i\in U} 1 + \sum_{i\in U}(i-1) + \sum_{j\in S} j\right) &= \sum_{i\in U}(i-1) + \sum_{j\in S} j \\
p\left(\sum_{i\in U} i + \sum_{j\in S} j\right) &= \sum_{i\in U}(i-1) + \sum_{j\in S} j \\
p\sum_{i\in U\cup S} i &= \sum_{i\in U}(i-1) + \sum_{j\in S} j \\
p &= \frac{\sum_{i\in U}(i-1) + \sum_{j\in S} j}{\sum_{i\in U\cup S} i}
\end{aligned}$$

Predicting Query Performance for User-based Search Tasks

Ying Zhao *Falk Scholer*

School of Computer Science and IT
RMIT University
Melbourne, Australia

$\{$*ying.zhao,falk.scholer*$\}$*@rmit.edu.au*

Abstract *Query performance prediction aims to determine in advance whether a user's search request will return a useful answer set. The success of such prediction attempts are currently evaluated by calculating the correlation between the predicted performance and standard information retrieval metrics of system performance such as average precision. However, recent work suggests that there is little relationship between average precision and the performance of users when carrying out search tasks. Direct measures of user performance offer another way of evaluating the effectiveness of search systems; this is of particular importance in the framework of query prediction, since one of the goals of prediction is to warn users when search results are likely to be poor. We therefore investigate the relationship between current prediction techniques and user-based performance measures. Our preliminary results show that the performance of the predictors differs strongly when using system-based compared to user-based performance measures: predictors that are significantly correlated with one measurement are often not correlated with the other. In general, the predictors are more correlated with average precision rather than with user performance.*

Keywords Query performance prediction, information retrieval, user study

1 Introduction

Query performance prediction has a wide range of potential applications that aim to improve search performance, seeking to provide users with knowledge about whether a set of search results are likely to contain useful answers for their information needs [2, 8]. The assumed benefit of prediction stems from the fact that the behaviour of a retrieval system could be changed dynamically based on the expected success of the query. For example, when a user enters a search request that is likely to lead to poor answers, the search system might prompt the user to re-formulate their query, without the user first having to work their way through a poor result list.

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

Current prediction techniques can be grouped into three categories: pre-retrieval prediction [4], post-retrieval prediction [11, 12], and learning prediction [9]. Despite differences in the prediction approaches, the evaluation of such techniques follows a common methodology: a set of topics is run over a chosen collection and, based on available relevance judgements, a per-topic performance metric is calculated. A prediction technique is then used to estimate the performance of each of the topics. The correlation between the predicted and "actual" performance values is then calculated, usually together with a statistical test to demonstrate that the correlation is significant. The higher the correlation, the better the predictor is deemed to be.

The "actual" performance metric that is to be predicted is a precision-based measurement; in all recent studies of query performance prediction the *average precision* (AP) of the search system [3, 4, 11, 12]. However, it has been shown that there is little relationship between AP scores and actual user performance on a variety of search tasks [7]. Since query performance prediction aims to support the *user* in resolving an information need, it is important to investigate whether current prediction techniques behave differently when considering user-based measures of search performance.

In this work, we report on initial experiments that examine prediction techniques using both the conventional evaluation metric (AP), and user-based performance. A total of 9 predictors are tested from different perspectives; the results demonstrate that using AP and user-based performance—different ways of measuring search system effectiveness—results in different evaluation outcomes for prediction techniques.

2 Background

Given a query $Q(t_1, \ldots, t_n)$ and a collection C, prediction methods compute a score to estimate the performance of a query. Pre-retrieval predictors are calculated based on information that is available at indexing time; there is no need to the search system to evaluate the full result set for a query, giving advantages in terms of simplicity and efficiency. Queries are distinguished by exploring term statistics and distributions in the collection. Several pre-retrieval predictors have been proposed in recent years, and we investigate 9 state-of-art predictors in this preliminary study. Since this work focuses on methodological aspects of prediction, we only provide brief details of the predictors themselves.

Clarity evidence. He and Ounis proposed a variety of pre-retrieval predictors [4]; based on their experimental results, the simplified clarity score (SCS) and average inverse collection term frequency (AveICTF) showed the best performance. SCS is a variation of the classic clarity score, a post-retrieval predictor originally proposed by Cronen-Townsend et. al [3].

$$\begin{aligned} SCS &= \sum_{t \in Q} \theta_q \cdot \log_2 \frac{\theta_q}{\theta_c} \\ AveICTF &= \frac{|C|}{f_{c,t}} \end{aligned}$$

where θ represents a language model (see He and Ounis [4] for estimation details); N is the number of terms in the collection; $|C|$ is the number of documents in the collection; and $f_{c,t}$ is the term frequency within the collection. We also consider the maximum inverse document frequency (MaxIDF) as a predictor [5]:

$$MaxIDF = \frac{N}{f_t}$$

where f_t is the number of documents that contain t.

Similarity evidence. This family of predictors computes a similarity score between a query vector and a collection vector [10]. The *SCQ* score combines evidence from the frequency with which terms occur in the collection, and the inverse document frequency. We also consider two variations—the normalised score (*NSCQ*), and the maximum SCQ score (*MaxSCQ*). The intuition behind *MaxSCQ* is that, since web search queries tend to be short, if at least one of the terms has a high score then the query as a whole can be expected to perform well. The three predictors are defined as:

$$\begin{aligned} SCQ &= \sum_{t \in Q} (1 + \ln(f_{c,t})) \times \ln\left(1 + \frac{N}{f_t}\right) \\ NSCQ &= \frac{SCQ}{|Q|_{t \in \mathcal{V}}} \\ MaxSCQ &= argmax \left[\forall_{t \in Q}\ SCQ_t\right] \end{aligned}$$

Variability evidence. Variability evidence is collected as the term distribution over the entire collection [10]. The standard deviation is a statistical measure of dispersion, which reflects how widely spread the values in a data set are around the mean. In the context of prediction, intuitively if the standard deviation of the distribution of term weights over the entire collection is low, then the retrieval system will be less able to distinguish between highly relevant and less relevant documents, and the query is therefore likely to be more difficult to evaluate. Again, we also consider the normalised and maximum versions of this measure:

$$\begin{aligned} \sigma_1 &= \sum_{t_1}^{t_n} \sqrt{\frac{1}{f_t} \sum_{d \in \mathcal{D}_t} (w_{d,t} \overline{w}_t)^2} \\ \sigma_2 &= \frac{\sigma_1}{|Q|_{t \in \mathcal{V}}} \\ \sigma_3 &= argmax \left[\forall_{t \in Q}\ \sigma_{1,t}\right] \end{aligned}$$

where $\overline{w}_t$ is the mean in-document term occurrence, $\frac{\sum_{d \in \mathcal{D}_t} w_{d,t}}{N}$.

3 Experimental Methodology

We investigate the difference in the performance of predictors when evaluated based on a standard IR metric, average precision (AP), and when using user-based measures. For our experiments we use the WT10g collection—a 10Gb crawl of the web in 1997 [1]. This collection was used in the TREC 9 and 10 Web tracks, and has a corresponding set of TREC topics and relevance judgements, with which the AP of retrieval systems can be calculated. For user-based measures, we use data collected in a user study by Turpin and Scholer [7]. 30 subjects conducted searches on the WT10g collection using 47 queries, and 5 different search systems, where each system returned answer lists at a pre-determined AP level.

A precision-oriented user-based measure of performance is the length of time that it takes to find the first relevant document for an information need. Analysis of the user data indicated statistically significant user effects (that is, some users take consistently longer than others to find the first answer), as well as statistically significant topic effects (across all search systems and users, it takes significantly longer to find a relevant answer for some topics than for others) [7]. As a user-centric measure of query difficulty, we use the average time required to find a relevant document for a topic. Since the raw time data does not appear to be normally distributed, we use the median time required to find an answer to measure the system effectiveness.

In the query performance prediction literature, three different correlation coefficients are widely used: the Pearson product-moment correlation; Spearman's rank order correlation; and, Kendall's tau. Correlation coefficients vary in the range $[+1, -1]$; a value of zero indicates that there is no relationship between the two groups of data. For each correlation, a statistical test can be performed to test whether the relationship is significant at a specified level of confidence (in this paper we use a standard significance level of 0.05). For a comprehensive treatment of the properties of the different correlation coefficients the reader is referred to Sheskin[6].

We note that there is no consensus about which correlation coefficient is the most appropriate for query performance prediction; individual papers often report

Table 1: *Pearson, Kendall, and Spearman correlation coefficients between user-based difficulty measure (median time to find a relevant document) and pre-retrieval predictors. Bold entries indicate that the correlation is statistically significant at the 0.05 level.*

	1	2	3	4	5	6	7	8	9
Predictor	SCS	AveICTF	MaxIdf	SCQ	NSCQ	MaxSCQ	σ_1	σ_2	σ_3
Pearson	0.107	-0.122	**0.390**	0.257	-0.127	0.208	**0.355**	0.142	**0.343**
p-value	*0.480*	*0.419*	*0.007*	*0.084*	*0.399*	*0.165*	*0.015*	*0.347*	*0.019*
Kendall	0.138	-0.098	0.197	0.163	-0.080	0.143	**0.208**	0.076	0.166
p-value	*0.179*	*0.346*	*0.053*	*0.112*	*0.428*	*0.161*	*0.042*	*0.462*	*0.103*
Spearman	0.204	-0.147	0.288	0.234	-0.128	0.212	**0.313**	0.112	0.236
p-value	*0.173*	*0.330*	*0.052*	*0.117*	*0.396*	*0.157*	*0.035*	*0.454*	*0.115*

only one or two of the available variants. However, the choice of correlation coefficient can lead to different conclusions about the performance of a predictor. Therefore in this study, we apply all three correlation methods to examine whether different test methods lead to different results with the same data sets.

4 Results and Discussion

We first analyse the correlation between the pre-retrieval predictors of query difficulty and our user-based measure of topic difficulty, the median time to find the first relevant document. The results are shown in Table 1. The σ_1 predictor is the most strongly correlated, and is the only predictor for which the correlation is statistically significant across all three correlation co-efficients. The correlations of the $MaxIDF$ and σ_3 predictors are significant only with Pearson correlation, and the other predictors show no significant relationship at all with the user-based measure of topic difficulty.

All of the selected predictors have been reported to be significantly correlated with a system-based performance measure, AP. However, these results are based on different collections or topics [4, 10]. We therefore conducted a second experiment, running the 47 topics for which we have user data as standard queries (using TREC title fields only), and calculating the AP of system performance on each. In this experiment, we used the Indri search engine with Dirichlet smoothing, where μ is set to 1000[1].

We compare the results of using different system performance metrics (AP and median search time) in evaluations of query prediction techniques. The results are presented in Figure 1 (a) to (c) for the three correlation coefficients. The numbers from 1 to 9 on the x-axis correspond to the 9 predictors (the ordering is the same as in Table 1). The y-axis shows the correlation coefficients. For each predictor, a pair of bars are shown: the left bar indicates the correlation with AP, and the right shows the correlation with the median time to find the first relevant document. Statistically significant correlations are shown as in dark gray, while non-significant results are shown as light gray.

There is a lack of consistency in the correlation results using the different difficulty measures: across all 3 correlation types, there are many cases (for example $MaxSCQ$ and σ_2) where a predictor is significantly correlated with AP but not with median time. The main exceptions are some poorly performing predictors such as SCS and SCQ that don't have a relationship with either difficulty measure. The only predictor that has a significant relationship with both measures is σ_1.

We note that not only the choice of difficulty measure, but also the choice of correlation technique, can affect the conclusions about predictor performance. For example, the Pearson correlation shows that $MaxIDF$ is significantly correlated with the median time measure; however, the other two correlations suggest that the relationship is not significant. Despite the inconsistencies observed, in general the selected predictors are more correlated with the average precision rather than user performance.

In a third experiment, we examine the direct relationship between the two measures of difficulty that we have used. The results for all three correlation coefficients are shown in Figure 1 (d). Although the Pearson correlation indicates a statistically significant relationship, the two rank-based correlation techniques do not identify a significant effect. The cause of this inconsistency may be an underlying assumption of the Pearson correlation, namely that the relationship is linear. There is no *a priori* reason to believe that a linear relationship should hold between the average precision of a retrieval system and the time taken to find the first relevant document. The results therefore do not provide evidence of a significant relationship between the user- and system-based measures of topic difficulty.

5 Conclusions

In this paper, we have conducted a preliminary investigation of using user-based measures of query difficulty for evaluating the effectiveness of query performance predictors. Experiments with 9 state-of-the-art pre-retrieval predictors showed that there is a lack of consistency in correlation; in general, previously proposed predictors tend to be related with only one or none of the difficulty measures. The only exception is

[1] Indri is available from `www.lemur.org`.

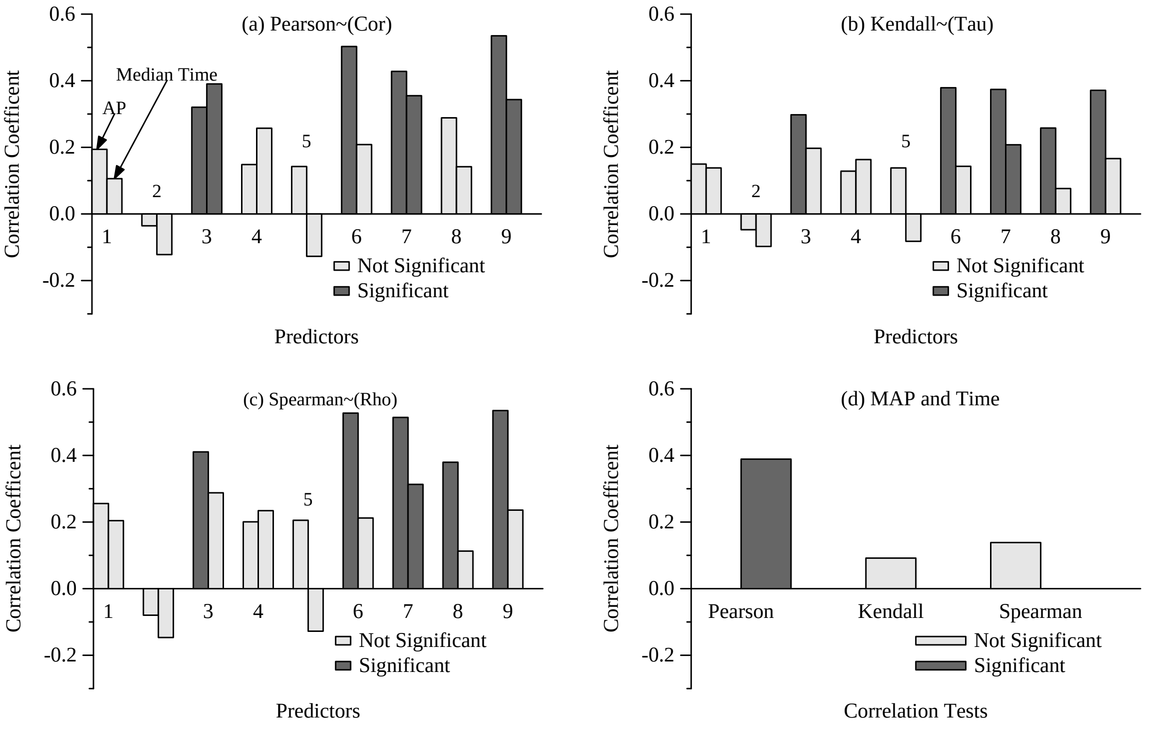


Figure 1: *The evaluation of prediction effectiveness using both average precision and user performance data. Three correlation tests are applied: Pearson shown as figure (a), Kendall as shown in figure (b), and Spearman as shown in figure (c). The relationship between two system performance metrics is shown in figure (d). In all figures, bars in dark gray indicate the results statistically significant at 0.05 confidence level.*

the σ_1 predictor—based on the variability of the distribution of query terms across documents—which had a significant relationship with both measures.

These findings have implications for the methodology that is currently used to evaluate the effectiveness of query performance predictors—it is not clear that simply showing a correlation with AP, as has been standard in the literature, is a suitable reflection of the actual difficulty that *users* face when searching on different topics.

In future work we plan to expand our investigation to include post-retrieval predictors, and to further examine the query performance prediction methodology. We also plan to investigate the use of the different correlation coefficients more thoroughly: since these sometimes give conflicting results, it needs to be established which measures are the most appropriate in the context of query performance prediction.

References

[1] P. Bailey, N. Craswell and D. Hawking. Engineering a multi-purpose test collection for web retrieval experiments. *Information Processing and Management*, Volume 39, Number 6, pages 853–871, 2003.

[2] D. Carmel, E. Yom-Tov and I. Soboroff. SIGIR workshop report: predicting query difficulty—methods and applications. *SIGIR Forum*, Volume 39, Number 2, pages 25–28, 2005.

[3] S. Cronen-Townsend, Y. Zhou and W. B. Croft. Predicting query performance. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2002.

[4] B. He and I. Ounis. Query performance prediction. *Information System*, Volume 31, Number 7, pages 585–594, 2006.

[5] F. Scholer, H. E. Williams and A. Turpin. Query association surrogates for web search. *American Society for Information Science and Technology*, Volume 55, Number 7, pages 637–650, 2004.

[6] D. Sheskin. *Handbook of parametric and nonparametric statistical proceedures*. CRC Press, 1997.

[7] A. Turpin and F. Scholer. User performance versus precision measures for simple search tasks. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 11–18, Seattle, USA, 2006.

[8] E. M. Voorhees. Overview of the TREC 2005 robust retrieval track. In *The Fourteenth Text REtrieval Conference (TREC 2005)*, Gaithersburg, MD, 2006. National Institute of Standards and Technology Special Publication 500-266.

[9] E. Yom-Tov, S. Fine, D. Carmel and A. Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 512–519, Salvador, Brazil, 2005.

[10] Y. Zhao, F. Scholer and Y. Tsegay. Effective pre-retrieval query performance prediction using similarity and variability evidence. In submission.

[11] Y. Zhou and W. B. Croft. Ranking robustness: a novel framework to predict query performance. In *Proceedings of the 15th ACM CIKM International Conference on Information Knowledge Management*, pages 567–574, Arlington, Virginia, USA, 2006.

[12] Y. Zhou and W. B. Croft. Query performance prediction in web search environments. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 543–550, Amsterdam, The Netherlands, 2007.

Use of Wikipedia Categories in Entity Ranking

James A. Thom
RMIT University
Melbourne, Australia
james.thom@rmit.edu.au

Jovan Pehcevski *Anne-Marie Vercoustre*
INRIA
Rocquencourt, France
{jovan.pehcevski, anne-marie.vercoustre}@inria.fr

Abstract *Wikipedia is a useful source of knowledge that has many applications in language processing and knowledge representation. The Wikipedia category graph can be compared with the class hierarchy in an ontology; it has some characteristics in common as well as some differences. In this paper, we present our approach for answering entity ranking queries from the Wikipedia. In particular, we explore how to make use of Wikipedia categories to improve entity ranking effectiveness. Our experiments show that using categories of example entities works significantly better than using loosely defined target categories.*

1 Introduction

Semi-structured text documents contain references to many named entities but, unlike fields in a well-structured database system, it is hard to identify the named entities within text. An entity could be, for example, an organisation, a person, a location, or a date. Because of the importance of named entities, several very active and related research areas have emerged in recent years, including: entity extraction/tagging from texts, entity reference solving (e.g. "The president of the Republic"), entity disambiguation (e.g. which Michael Jackson), question-answering, expert search, and entity ranking (also known as entity retrieval).

Entity ranking is very different from the related problem of entity extraction. The objective of *entity extraction* is to identify named entities from plain text and tag each and every occurrence; whereas the objective of *entity ranking* is to search for entities in a semi-structured collection and to get back a list of the relevant entity names as answers to a query (with possibly a page or some description associated with each entity).

The Initiative for the Evaluation of XML retrieval (INEX) is running a new track on entity ranking in 2007 [7], using Wikipedia as its document collection. In Wikipedia, pages correspond to entities which are organised into (or attached to) categories. References to entities and their categories occur frequently in natural language. For example, "France" is an named entity that corresponds to the Wikipedia page about France, belonging to categories such as "European Countries" and "Republics".

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright of this article remains with the authors.

There are two tasks in the INEX 2007 entity ranking track: a task where the category of the expected entity answers is provided; and a task where a few (two or three) of the expected entity answers are provided. The inclusion of target categories (in the first task) and example entities (in the second task) makes these quite different tasks from the task of full-text retrieval, and the combination of the query and example entities (in the second task) makes it a task quite different from the task addressed by an application such as Google Sets[1] where only entity examples are provided.

In this paper, we present our approach to entity ranking that augments the initial full-text information retrieval approach with information based on hypertext links and Wikipedia categories. In our previous work we have shown the benefits of using categories in entity ranking compared to full-text retrieval [19]. Here we particularly focus on how best to use the Wikipedia category information to improve entity ranking.

2 Related Work

The traditional entity extraction problem lies in the ability to extract named entities from plain text using natural language processing techniques or statistical methods and intensive training from large collections. Benchmarks for evaluation of entity extraction have been performed for the Message Understanding Conference (MUC) [17] and for the Automatic Content Extraction (ACE) program [11].

Entity extraction

McNamee and Mayfield [14] developed a system for entity extraction based on training on a large set of very low level textual patterns found in tokens. Their main objective was to identify entities in multilingual texts and classify them into one of four classes (location, person, organisation, or "others"). Cucerzan and Yarowsky [6] describe and evaluate a language-independent bootstrapping algorithm based on iterative learning and re-estimation of contextual and morphological patterns. It achieves competitive performance when trained on a very short labelled name list.

[1] http://labs.google.com/sets

Other approaches for entity extraction are based on the use of external resources, such as an ontology or a dictionary. Popov et al. [16] use a populated ontology for entity extraction, while Cohen and Sarawagi [4] exploit a dictionary for named entity extraction. Tenier et al. [18] use an ontology for automatic semantic annotation of web pages. Their system first identifies the syntactic structure that characterises an entity in a page. It then uses subsumption to identify the more specific concept for this entity, combined with reasoning exploiting the formal structure of the ontology.

Using ontology for entity disambiguation

Hassell et al. [12] use a “populated ontology” to assist in disambiguation of entities, for example names of authors using their published papers or domain of interest. They use text proximity between entities to disambiguate names (e.g. organisation name would be close to author’s name). They also use text co-occurrence, for example for topics relevant to an author. So their algorithm is tuned for their actual ontology, while our algorithm is more based on the structural properties of the Wikipedia.

Cucerzan [5] uses Wikipedia data for named entity disambiguation. He first pre-processed a version of the Wikipedia collection (September 2006), and extracted more than 1.4 millions entities with an average of 2.4 surface forms by entities. He also extracted more than one million (entities, category) pairs that were further filtered out to 540 thousand pairs. Lexico-syntactic patterns, such as titles, links, paragraphs and lists, are used to build co-references of entities in limited contexts. The knowledge extracted from Wikipedia is then used for improving entity disambiguation in the context of web and news search.

Ontology similarity

Since Wikipedia has some but not all characteristics associated with an ontology, one could think of adapting some of the similarity measures proposed for comparing concepts in an ontology and use those for comparing categories in Wikipedia. Ehrig et al. [9] and Blanchard et al. [2] have surveyed various such similarity measures. These measures are mostly reflexive and symmetric [9] and take into account the distance (in the path) between the concepts, the depth from the root of the ontology and the common ancestor of the concepts, and the density of concepts on the paths between the concepts and from the root of the ontology [2].

All these measures rely on a strong hierarchy of the ontology concepts and a subsumption hypothesis in the parent-child relationship. Since those hypothesis are nor verified in Wikipedia (see Section 4), we could not use those similarity functions. Instead we experimented with similarities between sets of categories and lexical similarities between category names.

```
<inex_topic>
<title>
European countries where I can pay with Euros
</title>
<description>
I want a list of European countries where
I can pay with Euros.
</description>
<narrative>
Each answer should be the article about a
specific European country that uses the
Euro as currency.
</narrative>
<entities>
   <entity ID="10581">France</entity>
   <entity ID="11867">Germany</entity>
   <entity ID="26667">Spain</entity>
</entities>
<categories>
   <category ID="185">european countries
   </category>
</categories>
</inex_topic>
```

Figure 1: Example INEX 2007 XML entity ranking topic

Entity ranking

Fissaha Adafre et al. [10] form entity neighbourhoods for every entity, which are based on clustering of similar Wikipedia pages using a combination of extracts from text content and following both incoming and outgoing page links. These entity neighbourhoods are then used as the basis for retrieval for the two entity ranking tasks.

Our approach is similar in that it uses XML structural patterns (links) rather than textual ones to identify potential entities. It also relies on the co-location of entity names with some of the entity examples (when provided). However, we also make use of the category hierarchy to better match the result entities with the expected class of the entities to retrieve.

3 INEX 2007 XML entity ranking track

The INEX XML entity ranking track is a new track that is being proposed in 2007. The track will use the Wikipedia XML document collection as its test collection.

Two tasks are planned for the INEX Entity ranking track in 2007 [7]:

task 1: *entity ranking*, which aims to retrieve entities of a given category that satisfy a topic described in natural language text; and

task 2: *list completion*, where given a topic text and a number of examples, the aim is to complete this partial list of answers.

An example of an INEX entity ranking topic is shown in Figure 1. In this example, the `title` field contains the plain content only query, the

> "The **euro** . . . is the official currency of the Eurozone (also known as the Euro Area), which consists of the European states of Austria, Belgium, Finland, France, Germany, Greece, Ireland, Italy, Luxembourg, the Netherlands, Portugal, Slovenia and Spain, and will extend to include Cyprus and Malta from 1 January 2008."

Figure 2: Extract from the Euro Wikipedia page

`description` provides a natural language description of the information need, and the `narrative` provides a detailed explanation of what makes an entity answer relevant. In addition to these fields, the `entities` field provides a few of the expected entity answers for the topic (task 2), while the `categories` field provides the category of the expected entity answers (task 1).

4 Wikipedia XML document collection

As Wikipedia is fast growing and evolving it is not possible to use the actual online Wikipedia for experiments, and so there is a need to use a stable collection to do evaluation experiments that can be compared over time. Denoyer and Gallinari [8] have developed an XML-based corpus based on a snapshot of the Wikipedia, which has been used by various INEX tracks in 2006 and 2007. It differs from the real Wikipedia in some respects (size, document format, category tables), but it is a very realistic approximation.

Entities in Wikipedia

In Wikipedia, an *entity* is generally associated with an article (a Wikipedia page) describing this entity. For example, there is a page for every country, most famous people or organisations, places to visit, and so forth. In Wikipedia nearly everything can be seen as an entity with an associated page. The Wikipedia is also a rich source of links, though some are to pages that do not exist yet! When mentioning entities in a new Wikipedia article, authors are encouraged to link at least the first occurrence of the entity name to the page describing this entity. This is an important feature as it allows to easily locate potential entities, which is a major issue in entity extraction from plain text.

For example, in the small extract from Euro page shown in Figure 2, there are 18 links (shown as underlined) to other pages in the Wikipedia, of which 15 are links to *countries*.

Categories in Wikipedia

Wikipedia also offers categories that authors can associate with Wikipedia pages. There are 113,483 categories in the INEX Wikipedia XML collection, which are organised in a graph of categories. Each page can be associated with many categories (2.28 as an average).

Wikipedia categories have unique names (e.g. "France", "European Countries"). New categories can also be created by authors, although they have to follow Wikipedia recommendations in both creating new categories and associating them with pages. For example, the Spain page is associated with the following categories: "Spain", "European Union member states", "Spanish-speaking countries", "Constitutional monarchies" (and some other Wikipedia administrative categories).

Some properties of Wikipedia categories include:

- a category may have many subcategories and parent categories;
- some categories have many associated pages (i.e. large *extension*), while others have smaller extension;
- a page that belongs to a given category extension generally does not belong to its ancestors' extension; for example, the page Spain does not belong to the category "European countries";
- the sub-category relation is not always a subsumption relationship; for example, "Maps of Europe" is a sub-category of "Europe", but the two categories are not in an *is-a* relationship; and
- there are cycles in the category graph.

Yu et al. [20] explore these properties in more detail.

5 Category similarity approaches

To make use of the Wikipedia categories in entity ranking, we define similarity functions between:

- categories of answer entities and target categories (for task 1), and
- categories of answer entities and a set of categories attached to the entity examples (for task 2).

Task 1

We first define a very basic similarity function that computes the ratio of common categories between the set of categories $\mathsf{cat}(t)$ associated to an answer entity page t and the set $\mathsf{cat}(C)$ which is the union of the provided target categories C:

$$S_C(t) = \frac{|\mathsf{cat}(t) \cap \mathsf{cat}(C)|}{|\mathsf{cat}(C)|} \qquad (1)$$

The target categories will be generally very broad, so it is to be expected that the answer entities would not generally belong to these broad categories. Accordingly, we defined several extensions of the set of categories, both for the target categories and the categories attached to answer entities.

The extensions are based on using sub-categories and parent categories in the graph of Wikipedia categories. We define $\mathsf{cat_d}(C)$ as the set containing the target category and its sub-categories (one level down) and $\mathsf{cat_u}(t)$ as the set containing the categories attached to an answer entity t and their parent categories (one level up). Similarity function can then be defined using the same ratio as above except that $\mathsf{cat}(t)$ is replaced with $\mathsf{cat_u}(t)$ and $\mathsf{cat}(C)$ with $\mathsf{cat_d}(C)$.

Another approach is to use lexical similarity between categories. For example, "european countries" is lexically similar to "countries" since they both contain the word "countries" in their names. We use an information retrieval approach to retrieve similar categories: we have indexed all the categories using their names as corresponding documents. By sending the category names C as a query to the search engine, we then retrieve all the categories that are lexically similar to C.

We keep the top M ranked categories and add them to C to form the set $\mathsf{Ccat}(C)$. We then use the same similarity function as before, where $\mathsf{cat}(C)$ is replaced with $\mathsf{Ccat}(C)$. We also experiment with two alternative approaches: by sending the title of the topic T as a query to the search engine (denoted as $\mathsf{Tcat}(C)$); and by sending both the title of the topic T and the category names C as a query to the search engine (denoted as $\mathsf{TCcat}(C)$).

An alternative approach of using lexical similarity between categories is to index the categories using their names and the names of all their attached entities as corresponding documents. For example, if C="countries", the retrieved set of categories $\mathsf{Ccat}(C)$ may contain not only the categories that contain "countries" in their names, but also categories attached to entities with names lexically similar to "countries".

Task 2

In task 2, the categories attached to entity examples are likely to correspond to very specific categories, just like those attached to the answer entities. We define a similarity function that computes the ratio of common categories between the set of categories attached to an answer entity page $\mathsf{cat}(t)$ and the set of the union of the categories attached to entity examples $\mathsf{cat}(E)$:

$$S_C(t) = \frac{|\mathsf{cat}(t) \cap \mathsf{cat}(E)|}{|\mathsf{cat}(E)|} \tag{2}$$

We also expand the two sets of categories by adding the parent categories to calculate $\mathsf{cat_u}(t)$ and $\mathsf{cat_u}(E)$ and apply the same similarity function as above.

6 Our approach to entity ranking

Our approach to identifying and ranking entities combines: (1) the full-text similarity of the entity page with the query; (2) the similarity of the page's categories with the target categories or the categories of the entity examples; and (3) the links to a page from the top ranked pages returned by a search engine for the query.

6.1 Architecture

The approach involves several modules and functions that are used for processing a query, submitting it to the search engine, applying our entity ranking algorithms, and finally returning a ranked list of entities. We use Zettair[2] as our choice for a full-text search engine. Zettair is a full-text information retrieval system developed by RMIT, which returns pages ranked by their similarity score to the query. We used the Okapi BM25 similarity measure that has proved to work well on the INEX 2006 Wikipedia test collection [1].

Our approach involves the following modules:

- The search module sends the query to Zettair and returns a list of scored Wikipedia pages. The assumption is that a good entity page is a page that answers the query.

- The link extraction module extracts the links from a selected number of highly ranked pages,[3] together with the information about the paths of the links (XML paths). The assumption is that a good entity page is a page that is referred to by a page answering the query; this is an adaptation of the Google PageRank [3] and HITS [13] algorithms to the problem of entity ranking.

- The linkrank module calculates a weight for a page based (among other things) on the number of links to this page (see 6.2). The assumption is that a good entity page is a page that is referred to from contexts with many occurrences of the entity examples. A coarse context would be the full page that contains the entity examples. Smaller and better contexts may be elements such as paragraphs, lists, or tables [15].

- The category similarity module calculates a weight for a page based on the similarity of the page categories with that of the target categories or the categories attached to the entity examples (see 6.3). The assumption is that a good entity page is a page associated with a category close to the target categories or categories of the entity examples.

- The full-text retrieval module calculates a weight for a page based on its initial Zettair score (see 6.4).

The global score for a page is calculated as a linear combination of three normalised scores coming out of the last three modules (see 6.5).

The above architecture provides a general framework for evaluating entity ranking.

6.2 LinkRank score

The linkrank function calculates a score for a page, based on the number of links to this page, from the first N pages returned by the search engine in response

[2] http://www.seg.rmit.edu.au/zettair/

[3] We discarded external links and some internal collection links that do not refer to existing pages in the INEX Wikipedia collection.

to the query. We carried out some experiments with different values of N and found that N=20 was an acceptable compromise between performance and discovering more potentially good entities.

We use a very basic linkrank function that, for an answer entity page t that is pointed to by a page p, takes into account the Zettair score of the referring page $z(p)$, and the number of reference links to the answer entity page $\#links(p,t)$:

$$S_L(t) = \sum_{r=1}^{N} z(p_r) * f(\#links(p_r,t)) \qquad (3)$$

where $f(x) = x$ (when there is no reference link to the answer entity page, $f(x) = 0$).

The linkrank function can be implemented in a variety of ways; for task 2 where entity examples are provided, we have also experimented by weighting pages containing a number of entity examples, or by exploiting the locality of links around the entity examples [15]. This more complex implementation of the linkrank function is outside the scope of this paper.

6.3 Category score

The basic category score $S_C(t)$ is calculated for the two tasks as follows:

task 1

$$S_C(t) = \frac{|\mathsf{cat}(t) \cap \mathsf{cat}(C)|}{|\mathsf{cat}(C)|} \qquad (4)$$

task 2

$$S_C(t) = \frac{|\mathsf{cat}(t) \cap \mathsf{cat}(E)|}{|\mathsf{cat}(E)|} \qquad (5)$$

We then consider variations on the category score $S_C(t)$ given the considerations in Section 5, using various combinations of replacing $\mathsf{cat}(t)$ with $\mathsf{cat_u}(t)$, replacing $\mathsf{cat}(C)$ with $\mathsf{cat_d}(C)$, $\mathsf{Ccat}(C)$, $\mathsf{Tcat}(C)$ or $\mathsf{TCcat}(C)$, and replacing $\mathsf{cat}(E)$ with $\mathsf{cat_u}(E)$.

6.4 Z score

The Z score assigns the initial Zettair score to an answer entity page. If the answer page does not appear in the final list of ranked pages returned by Zettair, then its Z score is zero:

$$S_Z(t) = \begin{cases} z(t) & \text{if page } t \text{ was returned by Zettair} \\ 0 & \text{otherwise} \end{cases} \qquad (6)$$

6.5 Global score

The global score $S(t)$ for an answer entity page is calculated as a linear combination of three normalised scores, the linkrank score $S_L(t)$, the category similarity score $S_C(t)$, and the Z score $S_Z(t)$:

$$S(t) = \alpha S_L(t) + \beta S_C(t) + (1 - \alpha - \beta) S_Z(t) \qquad (7)$$

where α and β are parameters that can be tuned.

A special case of interest here is when only the category score is used ($\alpha = 0.0$, $\beta = 1.0$). It allows us to evaluate the effectiveness of various category similarity functions and the overall benefit of using categories.

7 Experimental results

We now present results that investigate the effectiveness of our entity ranking approach for the two entity ranking tasks. We start by describing the test collection we developed for entity ranking.

7.1 Test collection

There is no existing set of topics with relevance assessments for entity ranking, although such a set will be developed for the INEX XML entity ranking track in 2007. So for these experiments we developed our own test collection based on a selection of topics from the INEX 2006 ad hoc track. We chose 27 topics that we considered were of an “entity ranking” nature, where for each page that had been assessed as containing relevant information, we reassessed whether or not it was an entity answer, and whether it *loosely* belonged to a category of entities we had *loosely* identified as being the target of the topic. If there were entity examples mentioned in the original topic these were usually used as entity examples in the entity topic. Otherwise, a selected number (typically 2 or 3) of entity examples were chosen somewhat arbitrarily from the relevance assessments. To this set of 27 topics we also added the Euro topic example (shown in Figure 1) that we had created by hand from the original INEX description of the entity ranking track [7], resulting in total of 28 entity ranking topics.

We use mean average precision (MAP) as our primary method of evaluation, but also report results using several alternative information retrieval measures: mean of P[5] and P[10] (mean precision at top 5 or 10 entities returned), and mean R-precision (R-precision for a topic is the P[R], where R is the number of entities that have been judged relevant for the topic). When dealing with entity ranking, the ultimate goal is to retrieve all the answer entities at the top of the ranking. Although we believe that MAP may be more suitable than the other measures in capturing these aspects, part of the track at INEX 2007 will involve determining what is the most suitable measure.

7.2 Task 1

For this task we carried out three separate investigations. First, we wanted to investigate the effectiveness of our category similarity module when varying the extensions of the set of categories attached to both the target categories and the answer entities. We also investigated the impact that this variation had on the effectiveness when the two different category indexes are

Table 1: Performance scores for runs using different retrieval strategies in our category similarity module (α0.0–β1.0), obtained for task 1 by different evaluation measures. For the three runs using lexical similarity, the Zettair index comprises documents containing category names (C), or documents containing category names and names of entities associated with the category (CE). The number of category answers retrieved by Zettair is M=10. For each measure, the best performing score is shown in bold.

	P[r]			
Run	5	10	R-prec	MAP
cat(C)-cat(t)	0.229	0.250	0.215	0.196
cat$_\mathsf{d}(C)$-cat$_\mathsf{u}(t)$	0.243	0.246	0.209	0.185
Ccat(C)-cat(t)	0.214	0.250	0.214	0.197
Tcat(C)-cat(t)	**0.264**	0.261	0.239	0.216
TCcat(C)-cat(t)	**0.264**	**0.286**	**0.247**	**0.226**

(C) Index of category names

	P[r]			
Run	5	10	R-prec	MAP
cat(C)-cat(t)	0.229	**0.250**	**0.215**	**0.196**
cat$_\mathsf{d}(C)$-cat$_\mathsf{u}(t)$	**0.243**	0.246	0.209	0.185
Ccat(C)-cat(t)	0.157	0.171	0.149	0.148
Tcat(C)-cat(t)	0.171	0.182	0.170	0.157
TCcat(C)-cat(t)	0.207	0.214	0.175	0.173

(CE) Index of category and entity names

used by Zettair. Second, for the best category similarity approach we investigated the optimal value for the parameter M (the number of category answers retrieved by Zettair). Last, for the best category similarity approach using the optimal M value, we investigated the optimal values for the α and β parameters. The aim of this investigation is to find the best score that could be achieved by our entity ranking approach for task 1.

Investigating category similarity approaches

The results of these investigations are shown in Tables 1(C) and 1(CE).[4] Several observations can be drawn from these results.

First, the choice of using the Zettair category index can dramatically influence the entity ranking performance. When cross-comparing the results in the two tables, we observe that the three lexical similarity runs using the Zettair index of category names substantially outperform the corresponding runs using the Zettair index of category and entity names. The differences in performance are all statistically significant ($p < 0.05$). Second, the run that uses the query that combines the terms from the title and the category fields of an INEX topic (TCcat(C)-cat(t)) performs the best among the three runs using lexical similarity, and overall it also performs the best among the five runs when using the Zettair index of category names. However, the differences in performance between this and the other four runs are not statistically significant. Third, extending the set of categories attached to both the target categories and the answer entities overall does not result in an improved performance, although there are some (non-significant) early precision improvements.

Investigating the parameter M

The above results show that the best effectiveness for our category similarity module (α0.0–β1.0) is achieved when using the Zettair index of category names, together with the query strategy that combines the terms from the title and the category fields of an INEX topic.

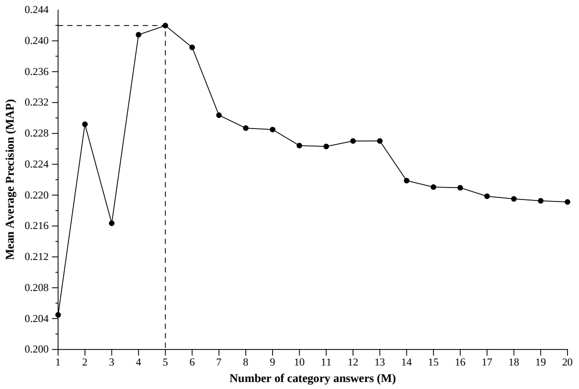


Figure 3: Investigating the optimal value for the number of category answers retrieved by Zettair, when using the run TCcat(C)-cat(t).

For these experiments we used a fixed value M=10 for the parameter M that represents the number of category answers retrieved by Zettair. However, since this was an arbitrary choice we also investigated whether a different value of M could also have a positive impact on the retrieval effectiveness. We therefore varied M from 1 to 20 in steps of 1, and measured the MAP scores achieved by our best performing TCcat(C)-cat(t) run using the Zettair index of category names.

Figure 3 shows the results of this investigation. We observe that a value of 5 for the parameter M yields the highest MAP score (0.242) for our category similarity module, which is a 7% relative performance improvement over the MAP score obtained with M=10. This performance improvement is statistically significant ($p < 0.05$).

Investigating the combining parameters α and β

To find the best score that could be achieved by our entity ranking approach for task 1, we used the run TCcat(C)-cat(t) with the optimal value M=5 and investigated various combinations of scores obtained from the three modules. We calculated MAP over the 28 topics in our test collection, as we varied α from 0 to 1 in steps of 0.1. For each value of α, we also varied β from 0 to $(1-\alpha)$ in steps of 0.1. We

[4] The first two runs do not use any of Zettair's category indexes and are included for comparison.

Table 2: Performance scores for runs using different retrieval strategies in our category similarity module (α0.0–β1.0), obtained for task 2 by different evaluation measures. For each measure, the best performing score is shown in bold.

	P[r]			
Run	5	10	R-prec	MAP
$\mathsf{cat}(E)$-$\mathsf{cat}(t)$	**0.536**	**0.393**	**0.332**	**0.338**
$\mathsf{cat}(E)$-$\mathsf{cat_u}(t)$	0.493	0.361	0.294	0.313
$\mathsf{cat_u}(E)$-$\mathsf{cat}(t)$	0.407	0.336	0.275	0.255
$\mathsf{cat_u}(E)$-$\mathsf{cat_u}(t)$	0.357	0.332	0.269	0.261

found that the highest MAP score (0.287) is achieved for $\alpha = 0.1$ and $\beta = 0.8$. This is a 19% relative performance improvement over the best score achieved by using only the category module (α0.0–β1.0). This performance improvement is statistically significant ($p < 0.05$). We also calculated the scores using mean R-precision instead of MAP as our evaluation measure, and we again observed the same performance behaviour and optimal values for the two parameters.

7.3 Task 2

For this task we carried out two separate investigations. First, as with task 1 we wanted to investigate the effectiveness of our category similarity module when varying the extensions of the set of categories attached to both the example and the answer entities. Second, for the best category similarity approach we investigated the optimal values for the α and β parameters, with the aim of finding the best score that could be achieved by our entity ranking approach for task 2.

Investigating category similarity approaches

The results of these investigations are shown in Table 2. We observe that, as with task 1, extending the set of categories attached to either (or both) of the example and answer entities does not result in an improved performance. The differences in performance between the best performing run that does not use the extended category sets and the other three runs that use any (or both) of these sets are all statistically significant ($p < 0.05$).

Investigating the combining parameters α and β

To find the best score that could be achieved by our entity ranking approach for task 2, we used the run $\mathsf{cat}(E)$-$\mathsf{cat}(t)$ and investigated various combinations of scores obtained from the three modules. We calculated MAP over the 28 topics in our test collection, as we used the 66 combined values for parameters α and β. We found that the highest MAP score (0.396) was again achieved for $\alpha = 0.1$ and $\beta = 0.8$. This score is a 17% relative performance improvement over the best score achieved by using only the category module (α0.0–β1.0). The performance improvement is statistically significant ($p < 0.05$).

Table 3: Comparing best performing runs for task 1 and task 2 for two distinct cases (using either category or global scores). The number of category answers retrieved by Zettair for run $\mathsf{TCcat}(C)$-$\mathsf{cat}(t)$ is M=5. For each case, the best results are shown in bold.

	P[r]			
Run	5	10	R-prec	MAP
Category score: α0.0–β1.0				
$\mathsf{TCcat}(C)$-$\mathsf{cat}(t)$	0.307	0.318	0.263	0.242
$\mathsf{cat}(E)$-$\mathsf{cat}(t)$	**0.536**	**0.393**	**0.332**	**0.338**
Global score: α0.1–β0.8				
$\mathsf{TCcat}(C)$-$\mathsf{cat}(t)$	0.379	0.361	0.338	0.287
$\mathsf{cat}(E)$-$\mathsf{cat}(t)$	**0.607**	**0.457**	**0.412**	**0.396**

7.4 Comparing Task 1 and Task 2

To investigate which of the two query strategies (target categories or example entities) is more effective for entity ranking, we compared the scores of the best performing runs across the two tasks. Table 3 shows the results of this comparison, when separately taking into account two distinct cases: a case when using scores coming out of the category module only (α0.0–β1.0); and a case when using optimal global scores coming out of the three modules (α0.1–β0.8).

We observe that, irrespective of whether category or global scores are used by our entity ranking approach, the run that uses the set of categories attached to example entities (task 2) substantially outperforms the run that uses the set of categories identified by Zettair using the topic title and the target categories (task 1). The differences in performance between the two runs are statistically significant ($p < 0.05$). This finding shows that using example entities is much more effective query strategy than using the loosely defined target categories, which allows for the answer entities to be identified and ranked more accurately.

8 Conclusions and future work

In this paper, we have presented our entity ranking approach for the INEX Wikipedia XML document collection. We focused on different entity ranking strategies that can be used by our category similarity module, and evaluated these strategies on the two entity ranking tasks. For task 1, we demonstrated that using lexical similarity between category names results in an effective entity ranking approach, so long as the category index comprises documents containing only category names. For task 2, we demonstrated that the best approach is the one that uses the sets of categories directly attached to both the example and the answer entities, and that using various extensions of these two sets significantly decreases the entity ranking performance. For the two tasks, combining scores coming out of the three modules significantly improves the performance compared to that achieved when using scores only from the category module. Importantly, when comparing the scores of the best performing runs across the two tasks, we found that the query strategy that uses example entities to identify the set of target categories is significantly more effective than the strategy that uses the set of loosely defined target categories.

In the future, we plan to further improve the global score of our entity ranking approach by using a better linkrank function that exploits different (static and dynamic) contexts identified around the entity examples. Preliminary results demonstrate that the locality of links around entity examples can indeed be exploited to significantly improve the entity ranking performance compared to the performance achieved when using the full page context [15]. To improve the category similarity function, we plan to introduce different category weighting rules that we hope would better distinguish the answer entities that are more similar to the entity examples. We will also be participating in the INEX 2007 entity ranking track, which we expect would enable us to test our approach using a larger set of topics.

Acknowledgements Part of this work was completed while James Thom was visiting INRIA in 2007.

References

[1] D.N.F. Awang Iskandar, Jovan Pehcevski, James A. Thom and S. M. M. Tahaghoghi. Social media retrieval using image features and structured text. In *Comparative Evaluation of XML Information Retrieval Systems: Fifth Workshop of the INitiative for the Evaluation of XML Retrieval, INEX 2006*, Volume 4518 of *Lecture Notes in Computer Science*, pages 358–372, 2007.

[2] Emmanuel Blanchard, Mounira Harzallah and Pascale Kuntz Henri Briand. A typology of ontology-based semantic measures. In *EMOI-INTEROP'05, Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability*, Porto, Portugal, 2005.

[3] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. In *Proceedings of the 7th International Conference on World Wide Web*, pages 107–117, Brisbane, Australia, 1998.

[4] William W. Cohen and Sunita Sarawagi. Exploiting dictionaries in named entity extraction: combining semi-Markov extraction processes and data integration methods. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, Seattle, WA, USA, 2004.

[5] Silviu Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on EMNLP and CoNLL*, pages 708–716, Prague, Czech Republic, 2007.

[6] Silviu Cucerzan and David Yarowsky. Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of the 1999 Joint SIGDAT Conference on EMNLP and VLC*, pages 90–99, Maryland, MD, USA, 1999.

[7] Arjen P. de Vries, James A. Thom, Anne-Marie Vercoustre, Nick Craswell and Mounia Lalmas. INEX 2007 Entity ranking track guidelines. In *INEX 2007 Workshop Pre-Proceedings*, 2007 (to appear).

[8] Ludovic Denoyer and Patrick Gallinari. The Wikipedia XML corpus. *SIGIR Forum*, Volume 40, Number 1, pages 64–69, 2006.

[9] Marc Ehrig, Peter Haase, Nenad Stojanovic and Mark Hefke. Similarity for ontologies — a comprehensive framework. In *Proceedings of the 13th European Conference on Information Systems*, 2005.

[10] Sisay Fissaha Adafre, Maarten de Rijke and Erik Tjong Kim Sang. Entity retrieval. In *Proceedings of International Conference on Recent Advances in Natural Language Processing (RANLP - 2007), September 27-29, Borovets, Bulgaria*, 2007.

[11] NIST Speech Group. The ACE 2006 evaluation plan: Evaluation of the detection and recognition of ACE entities, values, temporal expressions, relations, and events, 2006.

[12] Joseph Hassell, Boanerges Aleman-Meza and I. Budak Arpinar. Ontology-driven automatic entity disambiguation in unstructured text. In *Proceedings of the 5th International Semantic Web Conference (ISWC)*, Volume 4273 of *Lecture Notes in Computer Science*, pages 44–57, Athens, GA, USA, 2006.

[13] Jon M. Kleinberg. Authoritative sources in hyperlinked environment. *Journal of the ACM*, Volume 46, Number 5, pages 604–632, 1999.

[14] Paul McNamee and James Mayfield. Entity extraction without language-specific resources. In *COLING-02: Proceedings of the 6th Conference on Natural Language Learning*, pages 1–4, Morristown, NJ, USA, 2002.

[15] Jovan Pehcevski, Anne-Marie Vercoustre and James A. Thom. Exploiting locality of Wikipedia links in entity ranking. Submitted for publication, 2007.

[16] Borislav Popov, Atanas Kiryakov, Dimitar Manov, Angel Kirilov, Damyan Ognyanoff and Miroslav Goranov. Towards semantic web information extraction. In *2nd International Semantic Web Conference: Workshop on Human Language Technology for the Semantic Web and Web Services*, pages 1–22, Florida, USA, 2003.

[17] B. Sundheim (editor). *Proceedings of the Third Message Understanding Conference (MUC)*, Los Altos, CA, 1991. Morgan Kaufmann.

[18] Sylvain Tenier, Amedeo Napoli, Xavier Polanco and Yannick Toussaint. Annotation semantique de pages web. In *6emes journees francophones Extraction et Gestion de Connaissances (EGC)*, Lille, France, 2006.

[19] Anne-Marie Vercoustre, James A. Thom and Jovan Pehcevski. Entity ranking in Wikipedia. In *Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC08)*, Fortaleza, Brazil, 2008 (to appear).

[20] Jonathan Yu, James A. Thom and Audrey Tam. Ontology evaluation using Wikipedia categories for browsing. In *Proceedings of 16th ACM Conference on Information and Knowledge Management (CIKM 2007)*, pages 223–232, Lisboa, Portugal, 2007.

A Comparison of Evaluation Measures Given How Users Perform on Search Tasks

James A. Thom　　*Falk Scholer*

School of Computer Science and IT
RMIT University
Vic 3001 Australia

{*james.thom,falk.scholer*}*@rmit.edu.au*

Abstract *Information retrieval has a strong foundation of empirical investigation: based on the position of relevant resources in a ranked answer list, a variety of system performance metrics can be calculated. One of the most widely reported measures, mean average precision (MAP), provides a single numerical value that aims to capture the overall performance of a retrieval system. However, recent work has suggested that broad measures such as MAP do not relate to actual user performance on a number of search tasks. In this paper, we investigate the relationship between various retrieval metrics, and consider how these reflect user search performance. Our results suggest that there are two distinct categories of measures: those that focus on high precision in an answer list, and those that attempt to capture a broader summary, for example by including a recall component. Analysis of runs submitted to the TREC terabyte track in 2006 suggests that the relative performance of systems can differ significantly depending on which group of measures is being used.*

Keywords Information Retrieval, evaluation, metrics

1 Introduction

Information retrieval (IR) has a long history of experimental evaluation, following the “Cranfield” methodology: a set of queries (or topics) are run over a static collection of documents. For each returned query and document combination, a human judges whether the document is *relevant* to the query. This methodology is widely applied in IR research, and forms the basis of the on-going series of Text Retrieval Conferences (TREC).

Based on the relevance judgements, a variety of metrics can be calculated, aiming to reflect the performance of the retrieval system. Such metrics are generally based on two underlying concepts: the *precision* of a retrieval system, defined as the number of relevant documents retrieved as a proportion of the total number of documents that have been retrieved; and the *recall*, defined as the number of relevant documents that have been retrieved as a proportion of the total number of relevant documents in the collection. Precision therefore reflects the accuracy of an answer list, while recall measures the completeness.

Since information retrieval experiments generally focus on the performance of a system across a set of 50 or more queries, various summary measures are widely used. However, recent work has suggested that some of the most widely reported IR metrics have no relationship with user-based evaluation measures on precision-based search tasks [7]. Motivated by these findings, we investigate the relationship between different retrieval measures. Our results indicate that there is a distinct difference between measures that focus only on high-precision and those that aim to provide a more inclusive summary of retrieval performance.

In Section 2 we survey related work on comparison of information retrieval measures. Five commonly used information retrieval measures—precision at 1 (P@1), precision at 10 (P@10), mean average precision (MAP), mean reciprocal rank (MRR), and R-precision (RP)—are presented in Section 3. These measures are compared with each other on retrieval system runs submitted to the TREC Terabyte track. Finally, in Section 4 we discuss the implications of these results on how to determine what measures should be used when evaluating IR systems.

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

2 Related Work

A variety of information retrieval metrics have been proposed in the literature. While the relationship between some metrics has been considered previously [2, 8], such studies have not focused on precision at 1 and mean reciprocal rank. Recent studies have investigated correlations between retrieval metrics and user performance [7] or user satisfaction [1, 5].

Turpin and Scholer [7] found that commonly reported measures, in particular mean average precision, do not correspond well with user performance on simple information-finding web search tasks. Their results suggest that measures such as precision at 1 are more likely to reflect actual user performance.

Huffman and Hochster [5] found that for navigational queries to a search engine there was a close correlation between the relevance of the first result returned (P@1) and user satisfaction. While relevance of second and third results in the ranked list contributed a little to user satisfaction for navigational queries, these latter ranked results contributed more to user satisfaction for non-navigational queries. Al-Maskari et al. [1] compare precision and three cumulative gain measures with user satisfaction of accuracy, coverage, and ranking of results. They were not able to find one measure that captured all these aspects of user satisfaction.

3 Comparison of Measures

In order to compare whether different evaluation measures are measuring similar or different things, we compare the relative performance of all 80 runs that were submitted for the 2006 Terabyte track *adhoc* retrieval task. This task consisted of 149 informational queries run on the GOV2 collection (TREC topics 701–850).

The runs are compared using 5 standard IR evaluation measures. These five measures are defined for a given run (that is a ranked list of answers retrieved by a system) over the 150 topics as follows.

Precision at 1 (P@1): is the mean (calculated over all topics) of the precision of the top ranked document retrieved.

Precision at 10 (P@10): is the mean (calculated over all topics) of the precision of the first ten documents retrieved.

Mean average precision (MAP): is the mean (calculated over all topics) of average precision, where the average precision of a single query is the mean of the precision scores at each relevant item returned in a search results list.

Mean reciprocal rank (MRR): is the mean (calculated over all topics) of the reciprocal rank of the highest ranking relevant document (zero for a topic if no relevant documents were returned by the system).

R-precision (RP): is the mean (calculated over all topics) of the precision after R_t documents have been retrieved for topic t, where R_t is number of relevant documents available for topic t.

For each of the retrieval measures, the set of submitted runs can be *ordered* from the best-performing run (highest value for a metric) to the worst (lowest value for the same metric). We compare the obtained orderings between different metrics using Kendall's τ correlation coefficient. This coefficient measures the agreement between two sets of ranked data [6].

The correlation between different pairs of retrieval metrics is shown in Table 1, and is also presented graphically in Figures 1 to 7. The straight line in the figures is the line of best fit (if the relationship between the metrics is assumed to be linear; note that Kendall's τ does not make this assumption, since it is based on ranks).

Metrics		Kendall's τ
P@1	MAP	0.386
P@1	MRR	0.865
P@1	P@10	0.659
P@1	RP	0.376
MAP	MRR	0.350
MAP	P@10	0.583
MAP	RP	0.899

Table 1: Kendall's τ correlation between different retrieval metrics for 80 TREC runs. All correlations are statistically significant at $\alpha = 0.01$.

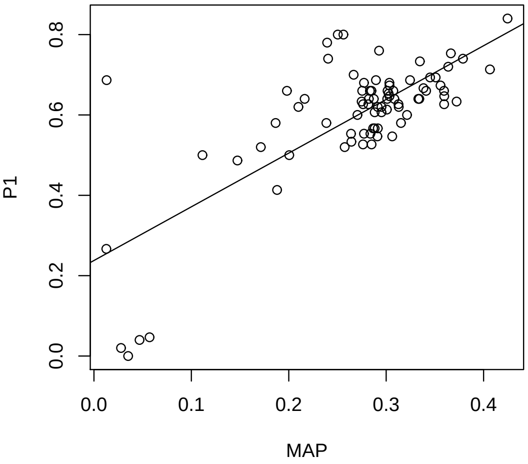


Figure 1: Correlation of P@1 with MAP

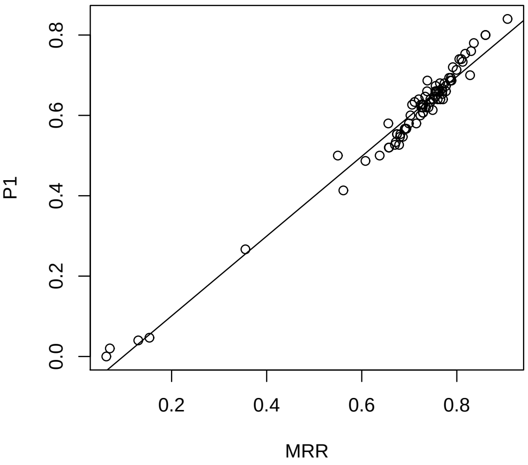


Figure 2: Correlation of P@1 with MRR

Figure 2 shows that there is a strong relationship between the two measures P@1 and MRR ($\tau = 0.865$); this is not surprising since both measures have a strong bias to systems that highly rank one relevant document.

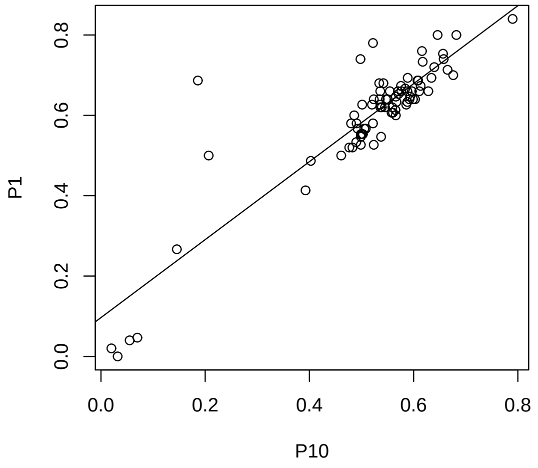


Figure 3: Correlation of P@1 with P@10

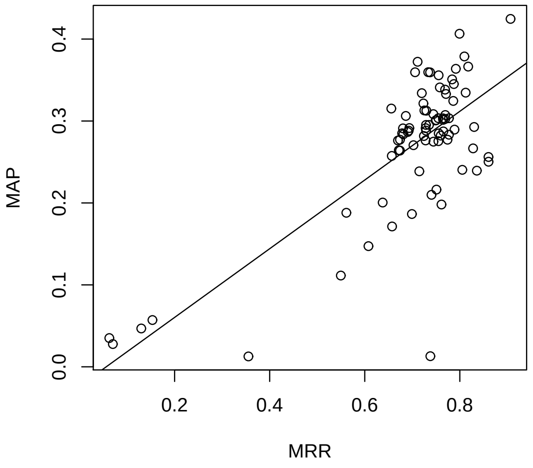


Figure 5: Correlation of MAP with MRR

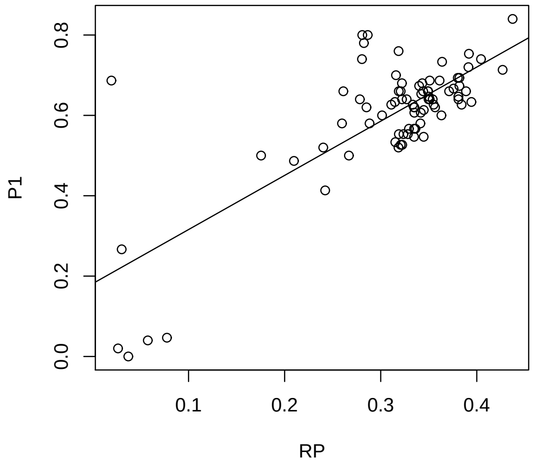


Figure 4: Correlation of P@1 with R-Precision

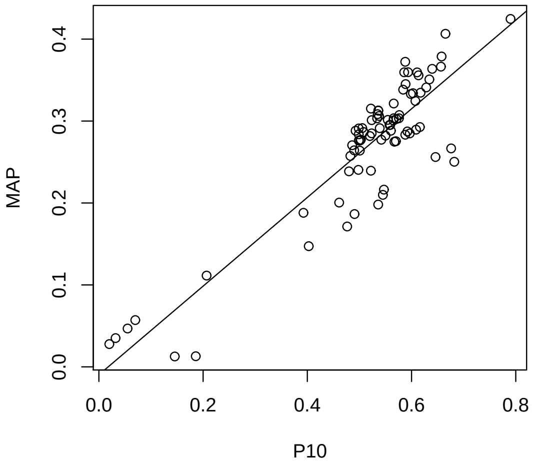


Figure 6: Correlation of MAP with P@10

Similarly, Figure 7 shows that there is a very close relationship between the two measures MAP and RP ($\tau = 0.899$); both of these measures incorporate a recall component.

As can be seen from Figures 3 and 6, there is not as strong a relationship between P@10 and either P@1 or MAP (the correlation coefficient is 0.659 and 0.583, respectively).

Figures 1 and 4 compare P@1 with MAP and RP and Figures 5 compares MAP with MRR; these figures show that systems that perform well at finding one relevant document either as the first answer (P@1), or highly ranked (MRR), do not necessarily perform so well in terms of measures of overall performance such as MAP or RP (while the correlations are statistically significant, τ is below 0.4 in each case, much lower than for other pairs of metrics).

We have also calculated correlations using the *bpref* measure, a retrieval metric that is intended for use in situations where only incomplete relevance judgements are available. When relevance judgements are mostly complete, bpref and MAP are closely correlated [3]. As a result, the correlations between bpref and other metrics closely reflect those between MAP and the other metrics, and are not reported here for brevity.

4 Discussion

Our results suggest at least two distinct categories of measures: those with a strong bias to highly ranking a relevant document (P@1 and MRR), and those that attempt to capture a broader summary of the performance (MAP, RP). The measure P@10 appears to share properties of both categories.

Although MAP has been widely accepted as the defacto standard for evaluation of information retrieval systems, it does not necessarily correspond to how users actually perform on search tasks. This is of particular concern because our results show that the correlations

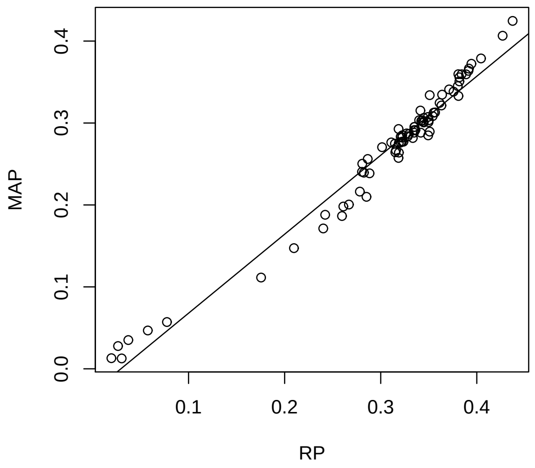


Figure 7: Correlation of MAP with R-Precision

between the two categories of metrics are weak – therefore, the relative ordering of systems that is commonly used in the TREC framework may not reflect user performance.

This preliminary analysis has considered only informational search tasks. Additional user studies are required to determine the wide range of situations—including navigational search tasks and question answering—in which measures such as P@1 and MRR are the more appropriate evaluation measures. Some of these different situations were investigated in the TREC Web track [4], which found that standard IR evaluation methodology did not adequately evaluate web search and different search tasks need specific evaluation metrics.

Furthermore, the retrieval metrics considered in this short paper are all based on binary relevance criteria (a document is either relevant, or it isn't). In future work, we intend to consider other evaluation metrics such as cumulative gain measures which take into account multi-level relevance judgements.

References

[1] Azzah Al-Maskari, Mark Sanderson and Paul Clough. The relationship between IR effectiveness measures and user satisfaction. In *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 773–774, Amsterdam, The Netherlands, 2007.

[2] C. Buckley and E. Voorhees. Evaluating evaluation measure stability. In Emmanuel Yannakoudakis, Nicholas J. Belkin, Mun-Kew Leong and Peter Ingwersen (editors), *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 33–40, Athens, Greece, 2000.

[3] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In Kalervo Järvelin, James Allan, Peter Bruza and Mark Sanderson (editors), *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 25–32, Sheffield, UK, 2004.

[4] David Hawking and Nick Craswell. Very large scale retrieval and web search. In Ellen Voorhees and Donna Harman (editors), *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press, 2005. `http://es.csiro.au/pubs/trecbook_for_website.pdf`.

[5] Scott B. Huffman and Michael Hochster. How well does result relevance predict session satisfaction? In *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 567–573, Amsterdam, The Netherlands, 2007.

[6] D. Sheskin. *Handbook of parametric and nonparametric statistical proceedures*. CRC Press, 1997.

[7] A. Turpin and F. Scholer. User performance versus precision measures for simple search tasks. In Charles L. A. Clarke, Norbert Fuhr, Noriko Kando, Wessel Kraaij and Arjen P. de Vries (editors), *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 11–18, Seattle, Washington, August 2006.

[8] E. Voorhees. Evaluation by highly relevant documents. In Donald H. Kraft, W. Bruce Croft, David J. Harper and Justin Zobel (editors), *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 74–82, New Orleans, LA, 2001.

Score Standardization for Robust Comparison of Retrieval Systems

William Webber *Alistair Moffat*
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria, Australia 3010
wew@csse.unimelb.edu.au, *alistair@csse.unimelb.edu.au*

Justin Zobel
NICTA Victoria Research Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria, Australia 3010
jz@csse.unimelb.edu.au

Abstract *Information retrieval systems are evaluated by applying them to standard test collections of documents, topics, and relevance judgements. An evaluation metric is then used to score a system's output for each topic; these scores are averaged to obtain an overall measure of effectiveness. However, different topics have differing degrees of difficulty and differing variability in scores, leading to inconsistent contributions to aggregate system scores and problems in comparing scores between different test collections. In this paper, we propose that per-topic scores be standardized on the observed score distributions of the runs submitted to the original experiment from which the test collection was created. We demonstrate that standardization equalizes topic contributions to system effectiveness scores and improves inter-collection comparability.*

Keywords Retrieval system evaluation, average precision, standardization.

1 Introduction

The effectiveness of information retrieval (IR) systems is traditionally evaluated using the Cranfield methodology [Cleverdon, 1991], which involves the use of a *test collection* consisting of a *document corpus*, a set of *topics*, and, for each topic–document pair, a judgment as to whether the document is relevant to that topic; these judgments are referred to as *qrels*. To evaluate an IR system, the topics are formulated as queries, and processed against the document corpus. For each topic, the system produces a ranked list of documents or *run*, ordering the documents by decreasing estimated relevance to the topic. The position of a document in a run is known as its *rank*.

Determining the effectiveness of a run involves applying an *evaluation metric* that uses the judged relevance of the document at each rank in the run to produce a score. Run scores for the topic set are averaged to give a system score. The statistical significance of a difference in system scores is then assessed using a paired significance test [Zobel, 1998, Sanderson and Zobel, 2005, Smucker et al., 2007].

Current test collections contain millions of documents, so it is not feasible to assess every document for relevance to every topic. Instead, test collections are formed in the context of a community *retrieval experiment* [Voorhees and Harman, 2005]. The runs from each participating system are submitted to the experiment, and documents for judging are selected from some or all of the submitted runs to form a *pool* [Sparck Jones and van Rijsbergen, 1975].

Different topics in a test collection can have quite different characteristics. Some have many relevant documents, others have few. For some topics, it will be easy for retrieval algorithms to identify relevant documents, while other topics may be ambiguous, causing systems to erroneously retrieve many documents that are plausible, but irrelevant. Human assessment of relevance also varies between topics, with the assessors sometimes employing strict criteria, while at other times being more liberal.

Variations in topic characteristics result in variability in the distribution of run scores across topics. As a result, different topics make different contributions to aggregate scores, and to tests for statistical significance. Topic score variability makes it particularly difficult to compare results obtained on different topic sets.

This paper proposes a topic score adjustment technique to ameliorate the problems caused by topic variability. The technique we describe is well known in other fields of testing, but until now has not, somewhat surprisingly, been suggested for use in the evaluation of retrieval systems. The essence of the approach is that individual run scores should be *standardized* before statistical tests are applied, with the adjustment to each run

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

score based upon the mean and standard deviation observed for the full set of run scores for that topic across the systems participating in the retrieval experiment. Once calculated, these *standardization factors* would be published as part of the test collection, and used to standardize the scores of future runs made against that collection.

Experimental results based on the data collected during one of the TREC[1] retrieval experiments show that standardization reduces per-topic variability, and increases the ability of statistical tests to discriminate between systems. The results also show that standardization allows systems to be compared even when they have been tested on different topics. The latter observation is of particular importance for real-world web search engines, where documents, queries, and retrieval algorithms are constantly changing, and having retrieval effectiveness measures that remain comparable over time is crucial.

2 Evaluation metrics

Many evaluation metrics for information retrieval have been proposed. Probably the most widely used is *average precision* (AP). Calculation of AP involves averaging the *precision* at each rank (the proportion of documents up to that position that are relevant) at which a run has a relevant document, for every known relevant document for that topic. Unretrieved relevant documents are assigned a precision of zero. To calculate AP it is thus sufficient to sum the precisions at each known retrieved relevant document in the run, down to some limiting depth that is set as part of the experimental design, and then divide by R, the total number of relevant documents. That is, AP includes an adjustment based on the "difficulty" of the topic, and can be thought of as being an R-normalized version of a more fundamental metric, the *sum of precisions at relevance* (SP) score of the run.

Other metrics also incorporate some form of score *normalization*, so as to reduce scores for "easy" topics and increase them for "hard" ones. For example, the *normalized discounted cumulative gain* (NDCG) method of Järvelin and Kekäläinen [2002] is derived from the *discounted cumulative gain* of a run (the sum of the relevance contributions of documents in a run, each discounted by the logarithm of its rank). In this case the normalization factor is the maximum possible DCG score that could have been achieved to that evaluation depth. On the other hand, some metrics make no attempt to adjust for topic variability. In precision-at-depth-d, for example, a run is scored on the proportion of top-d documents that are relevant, no matter what the total number of relevant documents for the topic are. Topics with many, easily-identified relevant documents get higher precision-at-d scores than topics with few, hard-to-find relevant documents.

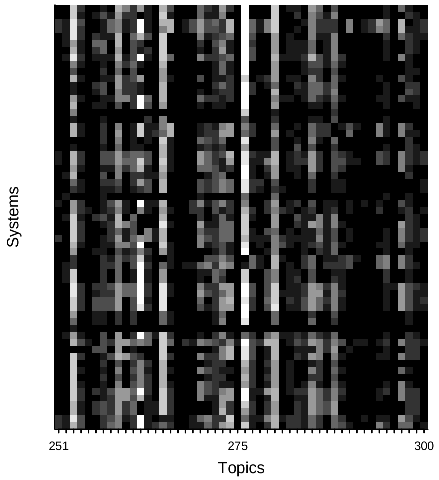


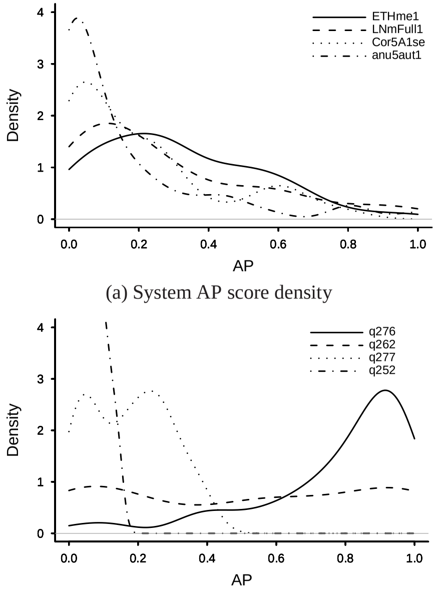
Figure 1: *Intensity visualization of run AP scores from the TREC5 Adhoc Track. The columns represent topics, ordered by topic number, and the rows represent systems, in ASCII order of system name. Each cell represents the AP score of a single run, with white cells indicating scores above* 0.9*; black cells scores below* 0.1*; and shades of grey indicating intermediate run scores. Easy topics stand out more clearly than good systems.*

The normalization methods of AP and NDCG can be considered separately from the metrics themselves. Any metric could be normalized by dividing by the total number of relevant documents (as in AP), or by the maximum possible score that could be achieved under that metric (as in NDCG). But note that both of these normalization methods require that the size of the set of relevant documents be known or estimated. Where qrels have been formed by pooling, the actual number of relevant documents is not known, and instead the number of pooled relevant documents is used. This makes the possibility of performing new relevance judgments for a new system problematic. If the normalization factor is updated when new relevant documents are found, then all previously reported scores have to be modified downwards; and if the normalization factor is not updated, then it is possible for a new system to achieve a score higher than the nominal maximum.

3 Topic variability

A community evaluation experiment involves running T topics against S systems. Each system thus produces T runs, one for each topic; and each topic receives S runs, one from each system. The runs are scored using an IR evaluation metric, such as AP. The resulting set of scores can be thought of as forming an $S \times T$ matrix. Figure 1 visualizes this matrix for the AP scores of

[1] http://trec.nist.gov



(a) System AP score density

(b) Topic AP score density

Figure 2: *Kernel density estimates of AP scores for the (a) systems, and (b) topics, at the first, tenth, twentieth, and seventy-fifth percentile (top to bottom in the legend) when ordered by average (a) system, or (b) topic, AP scores. Systems behave more like each other than do topics. All data is from the the TREC5 Adhoc Track.*

the TREC5 Adhoc Track as an intensity image, with lighter shades being higher scores. Topics are columns; systems are rows. The easy topics give rise to vertical white lines in the plot, and are easy to spot. Good systems should similarly give rise to horizontal white lines, but are much harder to pick out. Only the very worst systems, those with some programming bug or serious algorithmic misapplication, stand out as dark horizontal lines. This simple visualization makes it clear that the score matrix holds more information about the topics than it does about the systems.

The scores of the T topic runs for each system (in a row of the visualization matrix) form a distribution, as do the scores of the S system runs against each topic (in a column of the matrix). Figure 2 displays topic and system score distribution kernel density estimates (a form of smoothed histogram) from the TREC5 Adhoc Track for the first, tenth, twentieth, and seventy-fifth percentile topic and system, as ordered (in part (a)) by row averages, and (in part (b)) by column averages. The homogeneity of the system scores, and the heterogeneity of the topic scores, is immediately obvious. The system distributions all have the same, right-skewed unimodal shape, similar dispersions, and

	System AP			
	ETHme1	LNmFull1	Cor5A1se	anu5aut1
mean	0.317	0.282	0.206	0.154
st.dev	0.231	0.271	0.220	0.232

	Topic AP			
	q276	q262	q277	q252
mean	0.771	0.506	0.175	0.056
st.dev	0.235	0.383	0.118	0.039

Table 1: *Mean and standard deviation of AP scores for sample systems and topics from the TREC5 Adhoc Track. The kernel density estimates for the same data sets are plotted in Figure 2.*

Systems	Significance test	
	Paired	Two-sample
All	0.636	0.364
Auto	0.495	0.130

Table 2: *Proportion of system pairs from the TREC5 Adhoc Track found to have significantly different average precision at $p = 0.05$ in a two-tailed t-test, either paired or two-sample, and including either all 61 systems or only the 30 automatic systems minus the four faulty ones with* MAP $<$ 0.050*.*

even relatively similar localities. In contrast, the topic score distributions vary greatly.

Table 1 summarizes the locality and dispersions of the system and topic distributions plotted in Figure 2, in terms of their means and standard deviations. All systems have similar standard deviations, and the mean of the best system is only twice that of the seventy-fifth percentile. In contrast, topic means and standard deviations each vary tenfold from lowest to highest value.

Clearly, the mean AP scores for a test collection depend greatly on which topics happen to be selected, making it difficult to compare AP scores from different test collections. This is reflected in the lower discrimination resulting from using two-sample rather than paired significance tests to compare systems. In a two sample test, the null hypothesis is that the two samples are independent random samples from the same population. In a paired test, each item in one sample is assumed to have a shared dependency with a corresponding item in the other sample. Where the same test collection is used in evaluating two IR systems, the runs from each system for the same topic are paired, and the values fed into the hypothesis test are the deltas between the paired run scores. Pairing helps to control the effect of topic variability on scores, whereas reverting to a two-sample test gives an indication of the ability to find statistical significance when using different test collections sampled from the same (conceptual) population.

Table 2 contrasts the discriminative power of paired and two-sample significance tests on the the TREC5 Adhoc Track systems. The paired t-test finds a signif-

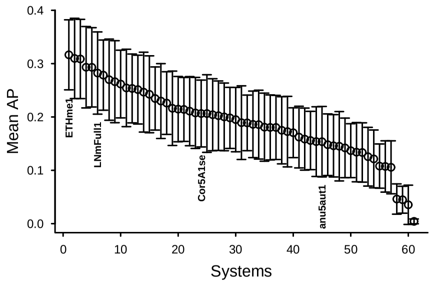


Figure 3: *The 95% confidence intervals on mean AP scores for the TREC5 Adhoc Track systems, using a t distribution. Systems are ordered by their official MAP score.*

icant difference between almost two-thirds of system pairs, whereas the two-sample test only finds significance for slightly over a third. If only the automatic runs are considered, and the four apparently faulty systems are excluded (all with a mean system AP score less than 0.050), then the outcome is even more stark. Half of the system pairs are found significantly different at the $p = 0.05$ level by the paired test, but only one eighth by the two-sample test. These results highlight the difficulty caused by inter-topic score variability when making comparisons between systems tested on different but similarly constituted test collections.

Figure 3 displays another effect of high inter-topic score variability. Here, 95% confidence intervals have been plotted on the "true" mean system AP (MAP) scores of the TREC5 Adhoc Track systems. These confidence intervals are statistically related to (though not identical with) the results of a two-sample significance test. The intervals on the MAP scores are wide: the best system could have a true MAP score of anywhere between 0.25 and 0.38; the median system's confidence interval is from 0.12 to 0.26; and the worst system (excluding the four faulty ones) sits in the range 0.06 to 0.16. With such wide confidence intervals, the strength of any conclusions based on mean AP score comparisons is extremely limited.

The marked difference in mean topic scores constrains experimenters to evaluate systems on the same test collection, so that they can employ paired significance tests. However, the high degree of difference in intra-topic AP standard deviations for different topics causes problems that even the one test collection and paired hypothesis testing cannot control. Consider once again the topic score standard deviations displayed in Table 1. Topic q262, the tenth-percentile topic by mean AP in the collection, has roughly ten times the AP score standard deviation of topic q252, the seventy-fifth-percentile topic. As a result, topic q262 will on average have ten times as much influence on the difference between the mean AP scores of any two systems as topic q252, and will have a similar impact in paired significance tests. Now, it happens that topic q262 is the highest-variance topic in the TREC5 Adhoc Track. However, even the twenty-fifth percentile topic when ordered by AP standard deviation has two and a half times the standard deviation of the seventy-fifth percentile topic (0.170 against 0.067), and therefore two and a half times the influence.

It might be supposed that topics with higher score variance are better at discriminating good from bad systems than are topics with lower variance, based on the intuition that the low-variance topics suffer more from "random noise" than the high-variance ones, or have too few relevant documents, or are too hard. The problem with testing such a conjecture is, of course, its circularity: we must first determine which are the good and which the bad systems before the reliable discrimination of a topic's scores can be determined. One way of approaching the issue is to consider how well the system scores for a topic correlate with the mean system scores for the rest of the topic set excluding that topic. This *item-total correlation* is used in Test Theory to indicate the *reliability* of a test component [Bodoff and Li, 2007]. Topic reliability can then be correlated with topic score standard deviation. Doing so for the TREC5 Adhoc Track systems (after excluding the four faulty runs, which score at or near zero for every topic and thus artificially inflate both the standard deviation and the reliability of topics with high mean scores) gives a Pearson correlation of 0.091 – that is, there is essentially no correlation. And even with this figure, we are not controlling for the fact that high-variance topics as a class still have more influence in determining the total scores even when individual high-variance topics are excluded. Thus, although high variance topics have greater influence on system mean AP scores and on paired significance tests, they are not inherently more reliable, and so their greater influence is neither deserved nor desirable.

4 Standardization

Topic score variability and the problems it causes can be addressed using score adjustment, and as was noted above, many effectiveness metrics have some kind of normalization built in. They are, however, rather indirect in their application to the problem of score variability. As the preceding discussion has demonstrated, AP's embedded normalization by R, the number of relevant documents, is not particularly effective.

An alternative method for reducing topic score variability – and one widely used in other experimental settings – is to normalize scores for a topic by the observed score mean and standard deviation of that topic. If a topic t has an unnormalized score mean of μ_t and an unnormalized score standard deviation of σ_t, and if a run for that topic receives the unnormalized score of x, then the normalized score x' for that run is:

$$x' = \frac{x - \mu_t}{\sigma_t}$$

Topic	Unstandardized AP				Standardized AP			
	ETHme1	LNmFull1	Cor5A1se	anu5aut1	ETHme1	LNmFull1	Cor5A1se	anu5aut1
q276	0.968	1.000	0.615	0.814	0.840	0.975	−0.667	0.180
q262	0.500	0.950	0.017	1.000	−0.015	1.161	−1.277	1.291
q277	0.344	0.301	0.256	0.059	1.434	1.067	0.689	−0.985
q252	0.045	0.058	0.030	0.109	−0.275	0.059	−0.665	1.340

Table 3: *Selected topics from the TREC5 Adhoc Track, and selected system AP scores, before and after standardization. The parameters μ_t and σ_t for these four topics are as listed in the bottom section of Table 1.*

Such a value is known as a z *score*, and the process of deriving it is commonly called *standardization* [Hays, 1991, chapter 4].

Note that a standardized distribution has the same shape as the unstandardized one, and that standardization only affects locality and dispersion. After standardization the mean score for each topic is zero, and the standard deviation is one. There are no fixed upper or lower bounds on a topic's scores. Chebyshev's inequality, however, guarantees that at least 75% of standardized scores for a topic will be between -2.0 and 2.0, and in practice the proportion will generally be much higher; for the TREC5 Adhoc Track standardized AP scores, it is 96%.

Unfortunately, the true mean and true standard deviation of each topic – the values that would be obtained from the conceptual population of runs of which the actually observed runs are only considered to be a sample – is not known. However, it can be estimated in the usual way from the statistics of the observed sample. That is, the standardization factors μ_t and σ_t for a topic t are estimated as the mean and standard deviation of the runs made against t in the original retrieval experiment. These estimates rely on there being an original retrieval experiment, of course, whereas normalization by total number of relevant documents or maximum achievable score could theoretically be performed in the absence of an experiment, by assessing every document for relevance. In practice, though, full assessment is impractical, and qrel sets are derived from pooling experimental systems. Therefore, the requirements for standardization are in practice no greater than for other forms of normalization.

It has already been mentioned that AP incorporates a crude form of normalization, and can be thought of as an unnormalized metric, sum of precisions at relevance, or SP, which is then normalized by the total number of known relevant documents, R. It is also possible to directly standardize the SP run scores rather than the AP scores, and in fact *exactly the same values* result from standardizing the SP scores as the AP scores. In other words, standardized SP is the same metric as standardized AP, freeing us from the need to know or estimate R. This observation alone would be sufficient to justify our interest in standardization of effectiveness scores. Similarly, standardized DCG and standardized NDCG are numerically identical measures.

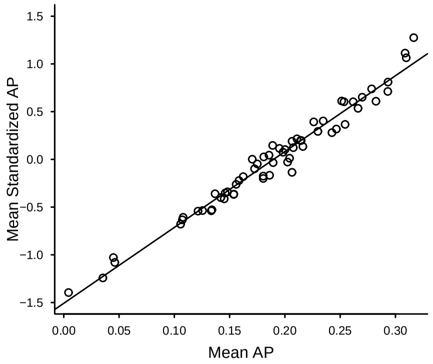


Figure 4: *Mean unstandardized AP and mean standardized AP scores for the TREC5 Adhoc Track systems, with the line of best fit.*

Table 3 shows the unstandardized and standardized AP scores for the previously examined topics and systems. The standardization factors for the topics are the sample mean and standard deviations previously reported in Table 1. The unstandardized figures are difficult to interpret and compare. For example, ETHme1 scored 0.344 for topic q277 and 0.500 for topic q262; but does the latter score represent a better result than the former, or was it simply an easier topic? Similarly, Cor5A1se scored 0.2 higher than anu5aut1 on topic q277, but 0.08 lower on q252; is the former result more significant than the latter? In contrast, the standardized results are directly informative. A positive score indicates the run outperformed the community mean for that topic, a negative score that it underperformed it; a score of 1.0 means the run is one standard deviation above the mean, and so on. So, without examining any other figures, we can immediately see that anu5aut1 has done well on topic q252, and Cor5A1se poorly on topic q262.

5 Experiments

Figure 4 plots the mean unstandardized AP and mean standardized AP scores for the TREC5 Adhoc Track systems against each other. The two metrics correlate closely. The Pearson correlation on the scores is 0.985, and the Kendall's τ correlation on the system ranks is

Systems	Significance test	
	Paired	Two-sample
All	0.683 +*0.047*	0.683 +*0.319*
Auto	0.561 +*0.066*	0.542 +*0.412*

Table 4: *Proportion of system pairs from the TREC5 Adhoc Track found to be significantly different using standardized AP at $p = 0.05$ in a two-tailed t-test, either paired or two-sample, and including either all 61 systems or only the 30 automatic systems minus the 4 faulty ones with* MAP < 0.05. *The improvement over the results with unstandardized AP reported in Table 2 is shown in italics.*

0.903. By way of comparison, the respective correlations of NDCG with mean (unstandardized) AP are 0.973 and 0.915. There are some local perturbations of ordering, however, which may represent improvements if we accept the thesis that topics should have similar impacts. The standardized AP scores separate out the three best systems, in the top right-hand corner, better than the unstandardized AP scores do. Note also that standardization shows the three best systems to perform almost as well as the four faulty systems do poorly, the former being on average one standard deviation better than the mean, the latter one standard deviation worse. It is not possible to make such a judgment simply by looking at the unstandardized MAP scores.

Table 4 compares the discriminative power of two-sample and paired significance tests on the standardized AP scores. In contrast to the unstandardized scores (see Table 2), with standardized scores, the two-sample significance test approaches the discriminative power of the paired test. The level of agreement between the two is high, the overlap (size of intersection divided by size of union) being 0.9 for the full system set, which is acceptable given the $p = 0.05$ significance level. Note also that the paired test itself is more discriminative for standardized than for unstandardized AP scores, rising for the full system set from 0.636 to 0.683 of pairs. This is the result of standardizing intra-topic score variance: deltas become comparable between topics, the variance of deltas falls on average, and the paired test is more confident about differences.

Figure 5 displays the 95% confidence intervals on the mean standardized AP scores for the TREC5 Adhoc Track systems, which are considerably narrower than for the mean unstandardized AP scores (Figure 3). Before standardization, the top system's confidence interval overlapped with the median's; after standardization it is well clear by the end of the first quartile.

One of the stated goals of score standardization is to improve the comparability of scores between different topic sets, provided that they are (actually or conceptually) randomly sampled from the same population of topics. As mentioned, when comparing two systems run against different topic sets, a two-sample significance test must be used. For significance testing, comparability can be considered in two aspects: the rate of false positives, and the rate of false negatives.

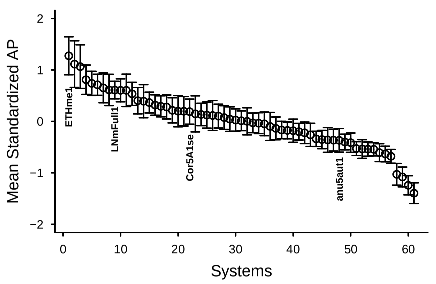


Figure 5: *The 95% confidence intervals on mean standardized AP scores for the TREC5 Adhoc Track systems, using a t distribution.*

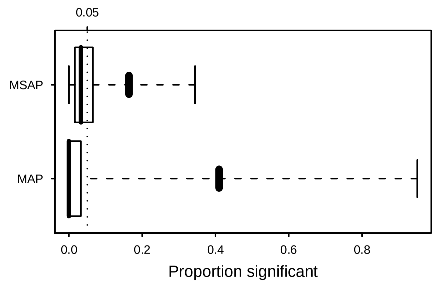


Figure 6: *False positives on two-tailed two-sample t-tests at $p = 0.05$ for the TREC5 Adhoc Track systems using 25-topic randomly-sampled subsets, repeated 5,000 times. The line within the box is the median; the left and right box edges are the 25th and 75th percentiles, respectively; the dotted whiskers extend to the extreme values; and the thick line on the right whiskers marks the upper bound of the 95% confidence interval. The dotted vertical line is the expected and approximate mean of both distributions.*

The rate of false positives can be investigated by testing a system against itself. Obviously, a system is not significantly different from itself, so if a significance test finds that it is, it must be a false positive. To explore this aspect, we took all 50 of the TREC5 Adhoc Track topics and randomly sampled two subsets of 25 topics each. The two subsets were not required to be disjoint; a disjoint partitioning would distort the extreme results, since if one subset happened to get all the hardest topics, then necessarily the other subset would get all the easiest ones [Sanderson and Zobel, 2005]. Then a two-tailed two-sample t-test was performed for each of the TREC5 Adhoc Track systems using the two topic subsets, and the proportion of false positives was recorded, using both the unstandardized and the standardized AP scores. This process was then repeated $5{,}000$ times, to provide a distribution of false positives. Figure 6 displays the results of this experiment. The mean rate of false positives is 0.043 and 0.049 for the unstandardized and standardized AP scores respectively, as is to be expected with $p = 0.05$. However, the variability of false positive rates for the unstandardized AP scores is considerably greater than for the standardized ones. Most topic partitionings show no false positives on unstandardized AP, indicating an insufficiently sensitive test (a rate of 0.05 is expected), but at the other extreme, there is a partitioning in which 95% of the systems are found to be better than themselves – a real boon for IR researchers wishing to publish papers. These results also demonstrate an important point about significance tests based around fixed test collections, which is that the chance of error (false positives or false negatives) is not independent between different pairs of systems; that is, if a collection makes an error on one pair, then it is more likely to make an error on the others. The seriousness of an error bias in a test collection is, however, greatly reduced by standardization.

Determining the rate of false negatives is more difficult, in that we do not know what the true positives are – that is, which systems truly are better than which. The approach taken here is to consider systems pairs that a paired, two-tailed t-test finds significant at $p = 0.001$ for both unstandardized and standardized AP – that is, pairs in which one system is almost certainly better than the other. One of these system pairs was chosen at random, along with a topic partitioning; then, a two-tailed, two-sample t-test was performed between the two systems, the first system using one topic subset, the second using the other, with both standardized and unstandardized AP scores used to evaluate effectiveness. Repeating this above experiment $5{,}000$ times, no significant difference at $p = 0.05$ was found in 41.6% of the pairs using unstandardized AP, whereas with standardized AP, only 2.6% of pairs were not found to be significantly different. Moreover, in only 0.2% of cases was significance found with unstandardized scores but not with standardized ones. These results provide compelling evidence that standardization enormously boosts the power of two-sample significance tests.

6 Related Work

Bodoff and Li [2007] introduce Test Theory concepts such as topic and test reliability into the evaluation of IR test collections. They find the TREC collections they investigate to be reliable in a test-theoretic sense, when AP is the evaluation metric. However, the standard of reliability is a rule-of-thumb, and what may be acceptable reliability in a test applied to human subjects may not be equally acceptable in IR evaluation. Reliability is assessed in terms of average correlation between topic scores, and inter-topic variation is not addressed.

Determining whether one metric is superior to another is problematic in the absence of a gold standard. Aslam et al. [2005] suggest the use of a maximum entropy method, finding AP to be superior to R-precision and precision-at-depth-d. Aslam et al. do not address the question of comparability of scores between systems and topics. Sakai [2006] proposes that metrics should be assessed based on the proportion of all system pairs in an experiment whose means are found to be significantly different in a paired bootstrap significance test. Smucker et al. [2007] compare the bootstrap, randomization, and t-tests, and find them to give similar results. Earlier, Voorhees and Buckley [2002] proposed an ad-hoc variant.

A popular alternative to AP is NDCG, proposed by Järvelin and Kekäläinen [2002], which incorporates its own normalization method, as described in Section 2. Another alternative is rank-biased precision or RBP [Moffat et al., 2007], which is bounded to the range $[0, 1]$, but is not normalized in the sense used here.

Mean AP scores a system by the arithmetic mean of the run AP scores. An alternative is to take the geometric mean, resulting in GMAP. Robertson [2006] discusses GMAP, observing that it is equivalent to the exponentiated average of the log of the per-topic AP values. Robertson further observes that taking the mean of a set of scores implies the assumption that score intervals, rather than score ratios, are the important unit of comparison; taking the log of the scores before averaging converts ratios to intervals. The GMAP metric is used in the Robust track of TREC, as it gives more emphasis to topics with low mean AP, which are regarded as “hard” [Voorhees, 2004]. We believe that score standardization may be a better solution.

Mizzaro and Robertson [2007] investigate normalizing per-run AP (or log AP) scores either by topic or by system, by subtracting the mean score for that topic or system (but not adjusting for the standard deviation). The adjusted AP scores are used as edge weights in a system-topic graph. Mizzaro and Robertson find considerable variation in topic difficulty and system performance on topics; that on the whole effective systems are better at distinguishing easy topics from hard ones; and that easy topics are better at distinguishing between more or less effective systems. The last finding they regard as undesirable. Their finding that easy topics distinguish good from bad systems may be a consequence of easy topics having larger variances under AP, and therefore more influence on final scores. Finally, Tague-Sutcliffe and Blustein [1994] undertake an analysis-of-variance on the TREC3 experimental data, and find that there is a stronger topic than system effect; Banks et al. [1999] discuss the analysis in more detail. Our experiments confirm their observations.

7 Conclusions and future work

We have shown that the power of a text retrieval experiment to discriminate between systems can be improved by standardizing run scores across a set of systems. Our application of standardization to AP, chosen because it is the commonest metric in current work, shows that not only does it allow better discrimination, but should allow comparison of results between different test collections. We plan to undertake similar experiments with other metrics, including discounted cumulative gain and rank biased precision.

Standardization removes what on reflection is a surprising limitation on the way TREC test collections have been used: although the runs submitted by these systems provide a wealth of potential information about the test topics and document corpus, the submitted runs are used solely to determine a pool of documents for relevance assessment, and are then ignored. Using these runs to estimate topic difficulty, and then standardizing topic scores based on these estimates, is one way in which the information in the submitted runs can be used to improve the reliability of the test collection. There are doubtless others.

There are also issues with some of the underlying assumptions in standard approaches to measurement of statistical significance. First, the assumption of random sampling from an underlying population is problematic. Second, significance testing is undertaken between pairs of systems in isolation, ignoring the wealth of data that is available from the other runs that are part of the experimental collection. Score standardization perhaps helps to resolve some of this oddity, but it still seems a strange procedure.

Several authors have, using AP and other measures, pursued investigations such as sampling the topic sets to see how robust the original pooling experiments were [Zobel, 1998, Sanderson and Zobel, 2005, Buckley and Voorhees, 2000, Voorhees and Buckley, 2002]. We expect that standardization would lead to substantial improvements, an avenue we also plan to explore. Finally, we will investigate the comparability of standardized results between different collections.

Acknowledgements This work was supported by the Australian Research Council.

References

J. A. Aslam, E. Yilmaz, and V. Pavlu. The maximum entropy method for analyzing retrieval measures. In *Proc. 28th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 27–34, Salvador, Brazil, 2005.

D. Banks, P. Over, and N.-F. Zhang. Blind men and elephants: six approaches to TREC data. *Information Retrieval*, 1(1): 7–34, April 1999.

D. Bodoff and P. Li. Test theory for assessing IR test collections. In *Proc. 30th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 367–374, Amsterdam, The Netherlands, July 2007.

C. Buckley and E. M. Voorhees. Evaluating evaluation measure stability. In *Proc. 23rd Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 33–40, Athens, Greece, 2000.

C. W. Cleverdon. The significance of the Cranfield tests on index languages. In *Proc. 14th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 3–12, Chicago, IL, USA, 1991.

W. L. Hays. *Statistics*. Harcourt Brace, Fort Worth, 4th edition, 1991.

K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

S. Mizzaro and S. Robertson. Hits hits TREC: exploring IR evaluation results with network analysis. In *Proc. 30th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 479–486, Amsterdam, The Netherlands, July 2007.

A. Moffat, W. Webber, and J. Zobel. Strategic system comparisons via targeted relevance judgments. In *Proc. 30th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 375–382, Amsterdam, The Netherlands, July 2007.

S. Robertson. On GMAP: and other transformations. In *Proc. 15th ACM Int. Conf. on Information and Knowledge Management*, pages 78–83, Arlington, VA, USA, November 2006.

T. Sakai. Evaluating evaluation metrics based on the bootstrap. In *Proc. 29th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 525–532, Seattle, WA, USA, 2006.

M. Sanderson and J. Zobel. Information retrieval system evaluation: effort, sensitivity, and reliability. In *Proc. 28th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 162–169, Salvador, Brazil, 2005.

M. D. Smucker, J. Allan, and B. Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *Proc. 16th ACM Int. Conf. on Information and Knowledge Management*, Lisboa, Portugal, 2007.

K. Sparck Jones and C. J. van Rijsbergen. Report on the need for and provision of an 'ideal' test collection. Technical report, University Computer Laboratory, Cambridge, 1975.

J. Tague-Sutcliffe and J. Blustein. A statistical analysis of the TREC-3 data. In *Proc. TREC-3*, November 1994. NIST Special Publication 500-225.

E. Voorhees and D. Harman, editors. *TREC: Experiment and Evaluation in Information Retrieval*. The MIT Press, 2005.

E. M. Voorhees. Overview of the TREC 2004 robust retrieval track. In *Proc. TREC-13*, November 2004. NIST Special Publication 500-261.

E. M. Voorhees and C. Buckley. The effect of topic set size on retrieval experiment error. In *Proc. 25th Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 316–323, Tampere, Finland, 2002.

J. Zobel. How reliable are the results of large-scale information retrieval experiments? In *Proc. 21st Ann. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 307–314, Melbourne, Australia, August 1998.

Efficient Neighbourhood Estimation for Recommenders with Large Datasets

Li-Tung Weng[*]*,Yue Xu, Yuefeng Li, Richi Nayak*
Faulty of Information Technology
Queensland University of Technology
QLD 4001, Australia
[*]soloman1124@hotmail.com, *{yue.xu, y2.li, r.nayak}@qut.edu.au*

Abstract *In this paper, we present a novel neighbourhood estimation method which is not only both memory and computation efficient but can also achieves better estimation accuracy than other cluster based neighbourhood formation techniques. In this paper we have successfully incorporated the proposed technique with a taxonomy based product recommender, and with the proposed neighbourhood formation technique both time efficiency and recommendation quality of the recommender are improved.*

Keywords Recommender Systems, Neighbourhood Estimation, Product Taxonomy

1. Introduction

Our main contribution in this paper is a novel neighbourhood estimation method called “relative distance filtering” (RDF), it is based on pre-computing a small set of relative distances between users, and using the pre-computed distances to eliminate most unnecessary similarity comparisons between users. The proposed RDF method is also capable of dynamic handling frequent data update; whenever the user preferences in the dataset are added, deleted or modified, the pre-computed structure cache can also be efficiently updated.

In this paper, we applied the proposed RDF technique to a well-known taxonomy recommender, namely taxonomy-driven product recommender (TPR), proposed by Ziegler[1]. TPR utilizes the taxonomy information of the products to solve the data sparsity and cold-start problems, and it outperforms standard collaborative filtering systems with respect to the recommendation accuracy when producing recommendations for sites with data sparsity. However, despite these benefits in TPR, the time efficiency of TPR drops significantly when dealing with huge number of users, because the user preferences in TPR are represented by very high dimensional vectors. After combining RDF with TPR, our experiment result shows that both the accuracy and efficiency of TPR are improved.

2. System Model

We envision a world with a finite set of users $U = \{u_1, u_2, \ldots, u_n\}$ and a finite set of items $T = \{t_1, t_2, \ldots, t_m\}$. For each user $u_i \in U$, he or she is associated with a set of corresponding implicit ratings R_i, where $R_i \subseteq T$. Unlike explicit ratings in which users are asked to supply their perceptions to items explicitly in a numeric scale, implicit ratings such as transaction histories, browsing histories, product mentions, etc., are more common and obtainable for most e-commerce sites and communities.

The TPR technique proposed by Ziegler[1] represents user profiles as taxonomy vectors, and the taxonomy vectors are constructed using users’ implicit ratings. Next, TPR makes recommendations to a given target user based on the common opinion of the user’s neighbourhood. For more information about the taxonomy vector construction and the TPR technique, please refer to [1].

3. Proposed Approach

In this paper, we propose a novel neighbourhood estimation method which is both memory and computation efficient. By substituting the proposed technique with the standard “best-n-neighbours” in TPR, the following two improvements are achieved:

- The computation efficiency of TPR is greatly improved.
- The recommendation quality of TPR is also improved as the impact of the “fixed n neighbours” problem has been reduced. That is, the proposed technique can help TPR locate the true neighbours for a given target user (the number of true neighbours might be smaller than n), therefore the recommendation quality can be improved as only these truly closed neighbours of the target user can be included into the computation.

3.1. Relative Distance Filtering

Forming neighbourhood for a given user $u_i \in U$ with standard “best-n-neighbours” technique involves computing the distances between u_i and all other users and selecting the top n neighbours with shortest distances to u_i. However, unless the distances between all users can be pre-computed offline or the number of users in the dataset is small, forming neighbourhood dynamically can be an expensive operation.

Clearly, for the standard neighbourhood formation technique described above, there is a significant amount of overhead in computing distances for users that are obviously far away (i.e., dissimilar users). The performance of the neighbourhood formation can be drastically improved if we exclude most of these very dissimilar users from the detailed distance computation. In the proposed RDF method, this exclusion or filtering process is achieved with a simple geometrical implication: if two points are very close to each other in a space, then their distances to a given randomly selected point in the space should be similar.

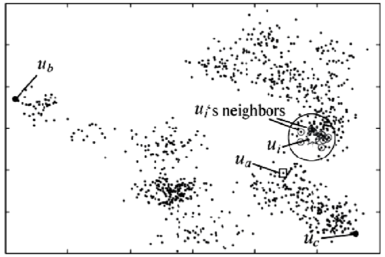


Figure 1: projected user profiles

In Figure 1, a user set U is projected onto a two-dimensional plane where each user is depicted as a dot on the plane. In the figure, u_i is the target user, and the dots embraced by small circles are the top 15 neighbours ofu_i. The RDF method starts by randomly selecting a reference user u_a in the user set, and thenu_a's distances to all other users are computed and sorted.

Based on the triangle inequality theme, it is easy to observe that all u_i's neighbours have similar distances to u_a. This means, in the process of formingu_i's neighbourhood, we only need to compute distances between u_i and the users in set U_ϑ^a which is defined as:

$$U_\vartheta^a = \{u_j \in U | u_j \neq u_i, (|\bar{a}_i - \bar{a}_j| < \vartheta)\} \quad (1)$$

where $\bar{a}_i$denotes the distance from u_a to u_i.

In equation (1), $|\bar{a}_i - \bar{a}_j|$ is the difference of the distances from u_i to u_a and u_j to u_a. ϑ is a distance threshold. If $|\bar{a}_i - \bar{a}_j|$ is larger than ϑ, the user $u_j \in U$ can be excluded from the u_i's neighbourhood. If ϑ is set to a larger value, the distance threshold is relaxed, thus more users can be included in the neighbourhood. In this case, the performance will be decreased because more users will be included in the actual distance computations. In our experiment, ϑ is set to the one tenth of the distance between the reference user and its furthest neighbour $u_j \in U$.

By incorporating more than one reference users, the performance of RDF can be further improved. For example, by introducing two other reference users u_b and u_c, the actually search space can be reduced to $U_\vartheta^a \cup U_\vartheta^b \cup U_\vartheta^c$.

3.2. Proposed RDF Implementation

In the RDF implementation, the distances between users and reference users are computed offline into a data structure called RDF searching cache, and it will be loaded into the memory in the initialization stage of the online recommendation process.

In the searching cache, each user is associated with a data structure called "user node". For any user $u_j \in U$, η_j denotes u_j's user node. A user node basically stores two types of information for a user:

1. **User ID**.
2. **Distances to the reference users**. The distances from the user node's corresponding user to the reference users are stored in a vector. In our implementation, we have only three reference users u_a, u_b andu_c, and therefore the distance vector for user node η_j is $(\bar{a}_j, \bar{b}_j, \bar{c}_j)$. We denote the distance vector of η_j as $\lambda_i = (\lambda_i^a, \lambda_i^b, \lambda_i^c)$ where λ_i^a corresponds t o$\bar{a}_j$, λ_i^b corresponds to$\bar{b}_j$and λ_i^ccorresponds to $\bar{c}_j$ respectively.

In order to efficiently retrieve the estimated searching space as described in equation (1), a binary tree structure is used to index and sort the user nodes. The index keys used for each user node are the distance between the user and the reference users, that is, the index keys for η_j are λ_i^a, λ_i^b and λ_i^c. With the three different index keys, the user nodes can be efficiently sorted with different index key settings, that is, the user nodes can be sorted by any one of the three index keys. An example of RDF searching cache is shown in Figure 2.

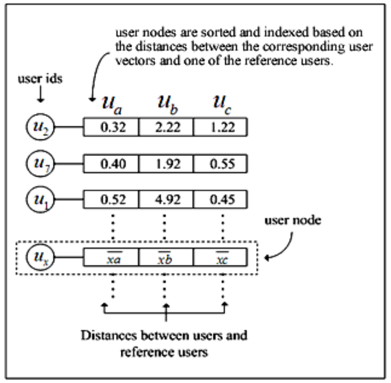

user nodes are sorted and indexed based on the distances between the corresponding user vectors and one of the reference users.
user ids
u_a
u_b
u_c
u_2
0.32
2.22
1.22
u_7
0.40
1.92
0.55
u_1
0.52
4.92
0.45
user node
u_x
$\overline{xa}$
$\overline{xb}$
$\overline{xc}$
Distances between users and reference users

Figure 2: structure for the RDF searching cache

Given that the RDF searching cache is properly initialized, the detailed RDF procedure is described below:

RDF Algorithm

1) Let u_i be the target user, n be the pre-specified number of neighbours for u_i.

2) Use the indexed tree structure to locate the minimal user nodes set within the given boundary:

$$\xi_i = \{\eta_j | u_j \in U, (\bar{x}_i - \vartheta) < \lambda_j^x < (\bar{x}_i + \vartheta)\}$$

where $u_x \in \{u_a, u_b, u_c\}$ which achieves minimal search space. Note, the actual implementation of u_x's computation can be very efficient. By utilizing the pre-computed searching cache, the estimation of user nodes size does not involve looping through the user nodes one by one.

3) Based on step 2, u_x is the primary index key used to sort and retrieve ξ_i, and it is one of u_a, u_b and u_c. The rest two index keys (also in $\{u_a, u_b, u_c\}$) are denoted as u_y and u_z.

4) We refine the searching space ξ_i by using reference users u_y and u_z.

```
FOR $\eta_x \in \xi_i$ DO
  IF $\lambda_x^y < (\bar{y}_i - \vartheta)$ or $\lambda_x^y > (\bar{y}_i + \vartheta)$ or
     $\lambda_x^z < (\bar{z}_i - \vartheta)$ or $\lambda_x^z > (\bar{z}_i + \vartheta)$
  THEN  remove  $\eta_x$ from $\xi_i$
```

5) Do the standard "best-n-neighbours" search against the estimated searching space ξ_i, and return the result neighbourhood for u_i.

4. Experiments and Evaluation

This section presents empirical results obtained from our experiment.

4.1. Data Acquisition

The dataset used in this experiment is the "Book-Crossing" dataset (http://www.informatik.uni-freiburg.de/~cziegler/BX/), which contains 278,858 users providing 1,149,780 ratings about 271,379 books. Because the TPR uses only implicit user ratings, therefore we further removed all explicit user ratings from the dataset and kept the remaining 716,109 implicit ratings for the experiment.

The taxonomy tree and book descriptors for our experiment are obtained from Amazon.com. Amazon.com's book classification taxonomy is tree-structured (i.e. limited to "single inheritance") and therefore is perfectly suitable to TPR.

4.2. Experiment Setup

The goal of our experiment in this paper is to compare the recommendation performance and computation efficiency between standard TPR [1] and the RDF-based TPR proposed in this paper.

The k-folding technique is applied (where k is set to 5 in our setting) for the recommendation performance evaluation. With k-folding, every user u_j's implicit rating list R_j is divided into 5 equal size portions. With these portions, one of them is selected as u_j's training set R_j^x, and the rest 4 portions are combined into a test set $T_j^x = R_j \backslash R_j^x$. Totally we have five combinations (R_j^x, T_j^x), $1 \leq x \leq 5$ for user u_j. In the experiment, the recommenders will use the training set R_j^x to learn u_j's interest, and the recommendation list P_j^x generated for u_j will then be evaluated according to T_j^x. Moreover, the size for the neighbourhood formation is set to 20 and the number of items within each recommendation list is set to 20 too.

For the computation efficiency evaluation, we implemented four different versions of TPRs, each of them is equipped with different neighbourhood formation algorithms. The four TPR versions are:

- **Standard TPR:** the neighbourhood formation method is based on comparing the target user to all users in the dataset.
- **RDF based TPR:** the proposed RDF method is used to find the neighbourhood.
- **RTree based TPR:** the RTree[2] is used to find the neighbourhood. RTree is a tree structure based neighbourhood formation method, and it has been widely applied in many applications.
- **Random TPR:** this TPR forms its neighbourhood with randomly chosen users. It is used as the baseline for the recommendation quality evaluation.

The average time required by standard, RTree based and the RDF based TPRs to make a recommendation will be compared. We incrementally increase the number of users in the dataset (from 1000, 2000, 3000 until 14000), and observe how the computation times are affected by the increments.

4.3. Evaluation Metrics

In this paper, the precision and recall metric is used for the evaluation of TPR, and its formulas are listed below:

$$Recall = 100 \times (|T_j^x \cap P_j^x| / |T_j^x|) \qquad (2)$$

$$Precision = 100 \times (|T_j^x \cap P_j^x| / |P_j^x|) \qquad (3)$$

4.4. Result Analysis

Figure 3 shows the performance comparison between the standard TPR and the proposed RDF based TPR using the precision and recall metrics. The horizontal axis for both precision and recall charts indicates the minimum number of ratings in the user's profile. Therefore larger x-coordinates imply that fewer users are considered for the evaluation. It can be seen from the result that the proposed RDF based TPR outperformed standard TPR for both recall and precision. The result confirms that when the dissimilar users are removed from the neighbourhood, the quality of the result recommendations become better. RTree based TPR performs much worse than both the

RDF based TPR and the standard TPR, as it is unable to accurately allocate neighbours for target users

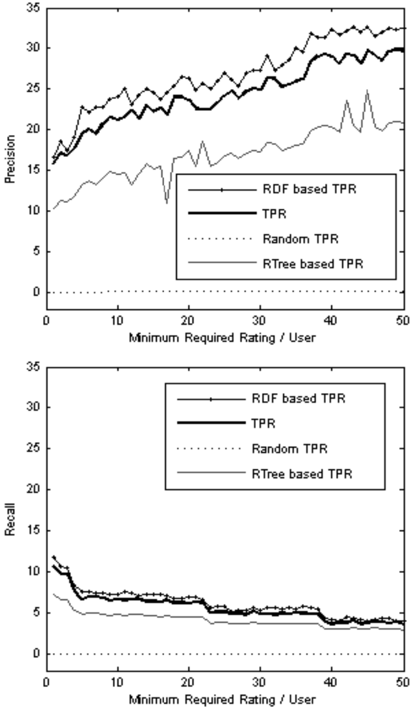


Figure 3: Recommendation precision and recall

The efficiency evaluation is shown in Figure 4.The time efficiency for standard TPR drops drastically when the number of users in the dataset increases. For dataset with 15000 users, the system needs about 14 seconds to produce a recommendation for a user, and it is not acceptable for most commercial systems. By comparison, the RDF based TPR is much efficient, and it only needs less than 4 seconds to produce a recommendation for dataset with 15000 users. The RTree based TPR greatly outperforms the proposed method when the number of users in the dataset is under 8000. However, as the number of users increases in the dataset, the differences between RDF and RTree based TPR becomes smaller, and RDF starts outperforms RTree when the number of users in the dataset is over 9000. This is because RTree is only efficient when the tree level is small. However, as the tree level increases (i.e. when number of users increases) RTree’s performance drops drastically because the chance for high dimensional vector comparison increases quadratically in accordance to the number of tree level. The proposed RDF method outperforms RTree method because its indexing strategy is single value based, and it reduces the possibility for the high dimensional vector correlation computation.

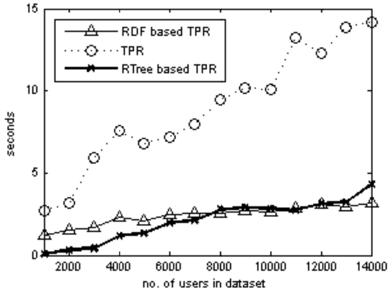


Figure 4: Average recommendation time

5. Conclusion

In this paper, we presented a novel neighbourhood estimation method for recommenders, namely RDF. By embedding RDF with a TPR based recommender, not only the computation efficiency of the system is improved, the recommendation quality is also improved. The RDF method is different from the clustering based neighbourhood formation methods that use offline computed clusters as the neighbourhoods. Instead, our method forms neighbourhood for any given target users dynamically from scratch (thus is more accurate than cluster based approaches) in an efficient manner.

In our experiment, it is shown that the proposed method improves both recommendation quality and computation efficiency for the standard TPR recommender.

References

[1] C.-N. Ziegler, G. Lausen, and L. Schmidt Thieme, "Taxonomy-driven Computation of Product Recommendations" in *International Conference on Information and Knowledge Management* Washington D.C., USA 2004.

[2] Y. Manolopoulos, A. Nanopoulos, A. N. Papadopoulos, and Y. Theodoridis, *R-Trees: Theory and Applications*: Springer, 2005.

Querying Image Ontology

D. N. F. Awang Iskandar *James A. Thom* *S. M. M. Tahaghoghi*

School of Computer Science and Information Technology,
RMIT University,
Melbourne, Australia

{*dayang.awgiskandar, james.thom, seyed.tahaghoghi*}@*rmit.edu.au*

Abstract *Content-based image retrieval has been used in various application domains, but the semantic gap problem remains a challenge to be overcome. One possible way to overcome this problem is to represent the knowledge extracted from the low-level image features through semantic concepts. In this paper we describe how we use an image ontology to this end. We show that we are able to retrieve desired images by using basic ontology queries.*

Keywords Ontology, CBIR, semantic gap

1 Introduction

Advances in digital imaging technology have led to a large volume of digital visual content being produced. However, this has exacerbated the problem of locating a desired image in a large and varied collection. Content-based image retrieval (CBIR) aims to retrieve images on the basis of features automatically extracted from the images themselves.

The main goal of CBIR is to find an image or a set of images that best satisfy a user's information need within an image database or collection. However, the fundamental unsolved problem in CBIR is the semantic gap [14] – the missing link between the semantic categories that a user is looking for and the low-level features that CBIR systems offer.

High-level retrieval involves retrieval of an image based on the name of objects, emotions and actions that can be associated with the image. Low-level retrieval involves retrieval of basic features such as colour, texture, shape and object location. For example, the concepts "president" and "happiness" are considered to be high-level, whereas "red circle" is a low-level concept. Users generally wish to pose high-level queries [8, 9, 14] whereas present CBIR systems can only index and retrieve images based on low-level features.

Approaches to bridge the semantic gap can be top-down, bottom-up or a combination [4] of these. In this work, we adopt a combined approach. For the top-down approach we create an ontology that contains the high level semantic concepts derived from the image regions' low-level features. For the bottom up approach, we automatically learn the semantic concepts using the technique described in earlier work [1]. The main technical contribution of this work is the ontology that we have engineered. We also show that we are able to retrieved desired images using the ontology.

Next, we describe existing research on CBIR and ontology. In Section 3, we describe our approach for querying an image ontology. In Section 4, we explain the ontology's structure and content. In Section 5, we present a query example and the retrieved images. We conclude with a discussion of our findings and suggestions for future work.

2 Background

Most effort to minimize the semantic gap has focused on automatic image annotation [2, 7, 17]. The images are annotated by using keywords or described formally using an ontology [5, 13, 17]. According to Brewster et al. [3], an ontology defines concepts properties and the relationships among the concepts.

Web ontology languages have been proposed as part of research related to the Semantic Web. XML, RDF, RDF Schema, OIL and DAML+OIL are among the earliest web ontology languages, while OWL is the current W3C recommendation.[1] A combination of RDF and OWL (RDF/OWL) can accurately describe the instances and their constraints in an ontology. RDF is used to represent information and to exchange knowledge on the Web. At a higher level, OWL is used to publish and share sets of terms called ontologies, supporting advanced Web search, software agents and knowledge management.

Ontology query languages allow expressions to be written that can be evaluated against an ontology. The queries can be used by knowledge management applications as a basis for inference actions. Existing ontology query languages include OntoQL, SPARQL, DQL (previous version of DAML+OIL), SeRQL, TRIPLE, RDQL, N3, and Versa. The SPARQL query language has been adopted by W3C as the means to query ontologies built using RDF[2] and has been extended to support OWL format. SPARQL is based on SQL and has the ca-

Proceedings of the 12th Australasian Document Computing Symposium, Melbourne, Australia, December 10, 2007. Copyright for this article remains with the authors.

[1] `http://www.w3.org/TR/owl-guide`

[2] `http://www.w3.org/TR/rdf-sparql-query`

pabilities for querying visual graph patterns along with their conjunctions and disjunctions.

Town [15, 16] shows that the use of ontologies to relate semantic descriptors to their parametric representations for visual image processing leads to an effective computational and representational mechanism. The ontology implements the hierarchical representation of the domain knowledge for a surveillance system. Preannotated surveillance video training data and its visual descriptors are incorporated in the ontology. The ontology is used to feed information to the Bayesian inference network for tracking movement. Town also proposed an ontological query language, OQUEL. The query is express using a prescriptive ontology of image content descriptors. Query sentences are grounded through a range of image analysis methods that represent the image content in low, intermediate and high semantic levels. The central role of the ontology is to provide a means for users to define the ontological domain of discourse and for the system to execute the query by grounding and assessing the particular ontological sentence with respect to actual image data.

The query approach using OQUEL is similar to the approach presented by Hyvönen et al. [6] who implement a web-based system to retrieve the images using an ontology — known as Ontagator. Image query is done using a view-based search followed by image recommendations. In the search process, users view the ontology and select the class of interest. The system will return all images related to the class. After finding a class of interest, the semantic ontology model together with image instance data are used to discover the relations between a selected image and other images in the repository. These images are then presented to the user.

Liu et al. [10] also implemented a web-based system to retrieve the images with an ontology. Search for the matching images is done by processing a text-based query. The ontology query engine is written in RDF Data Query Language (RQDL) provided by the Jena Toolkit.[3]

Mezaris et al. [11, 12] propose an approach for region-based image retrieval using an object ontology and relevance feedback. The approach utilises an unsupervised segmentation method for dividing the images into regions that are later indexed. The object ontology is used to represent the low-level features and act as a object relation identifier — for example the shape features are represented as *slightly oblong, moderately oblong, very oblong*. This ontology is not built using any ontology language, but is instead simply a vocabulary listing. The query is done using keywords in the object ontology to provide qualitative information and relationships between objects. The regions that match the query based on the object ontology are retrieved and presented to user. The user can give feedback on the retrieved images and the system will learn using Support Vector Machines (SVMs) and Constraint Similarity Measure (CSM) to filter out the unrelated images.

3 Our Approach

To reduce the problem of object segmentation, we test our approach on a domain where regions are easily separated: a collection of comic strips. In this domain, objects and characters comprise multiple regions of approximately uniform colour.

We have created an image collection that consists of comic strip panels from the Goats comic.[4] These include extended information that describes every panel. The description assists us in performing relevance judgements on our retrieval results. The collection consists of 452 coloured strips, each containing one to five panels. Dividing the strips into panels gives us a total of 1440 panels. We tested the retrieval effectiveness using 1115 regions extracted from 202 panels. The remaining panels are reserved as a validation set for future work. From this point onwards, we will refer to individual panels as images.

The objects in the comics have relatively consistent size and orientation, guiding our choice of the following region-based and contour-based shape features: the region area; the mean grey level value of the pixels in the region; circularity; and shape boundary. We did not use any texture features in this work since colour is a much more prominent feature in the comic image collection we are using.

We adopted the equal-weight linear combination technique from our previous work [1] to recognise and label five concepts representing the main characters in the Goats comic strips — *Bob* (an alien), *Diablo* (a chicken), *Fineas* (a fish), *Jon* (a person) and *Oliver* (a chick). This technique was compared with classification using machine learning algorithms in combining shape features, and we found that an equal-weight linear combination of shape features is simpler and at least as effective as using a machine learning algorithm [1].

4 The Image Ontology

To build the ontology, we incorporate the concepts that were recognised using the above mentioned method. The ontology is automatically augmented when new concepts are derived from the images. The image retrieval is performed using SPARQL.

We divided the ontology structure into two general classes – Concept and Graphic. The class Concept has a subclass Character that further contains character instances – *Bob*, *Diablo*, *Fineas*, *Oliver* and *Jon*. The Graphic subclasses are Strip and Panel. The subclass Panel contains instances that describes the panel sequence (first panel labelled as “a” and so forth) and the character or characters in it. Currently, we have

[3] `http://jena.sourceforge.net`

[4] `http://www.goats.com`

```
<?xml version="1.0"?>
 <rdf:RDF
 xmlns="http://dayang.cs.rmit.edu.au/~dayang/ComicOntology#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xml:base="http://dayang.cs.rmit.edu.au/~dayang/ComicOntology">
 <owl:Ontology rdf:about="Goats Comic"/>
 <owl:Class rdf:ID="Concept"><rdfs:subClassOf><owl:Class rdf:ID="Comic"/></rdfs:subClassOf></owl:Class>
 <owl:Class rdf:ID="Graphic"><rdfs:subClassOf rdf:resource="#Comic"/></owl:Class>
 <owl:Class rdf:ID="Strip"><rdfs:subClassOf rdf:resource="#Graphic"/></owl:Class>
 <owl:Class rdf:ID="Panel"><rdfs:subClassOf rdf:resource="#Graphic"/></owl:Class>
 <owl:Class rdf:ID="Character"><rdfs:subClassOf rdf:resource="#Concept"/></owl:Class>
 <owl:ObjectProperty rdf:ID="HasCharacter"/>
 <owl:ObjectProperty rdf:ID="PartOf"/>
 <owl:ObjectProperty rdf:ID="InPanel"/>
 <owl:DatatypeProperty rdf:ID="name"/>
 <owl:DatatypeProperty rdf:ID="image_Ref"/>
 <Strip rdf:ID="goats031226.png-a.jpg">
    <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">goats031226.png-a.jpg</name></Strip>
 <Panel rdf:ID="a">
    <PartOf rdf:resource="#goats031226.png-a.jpg"/>
    goats031226.png-a.jpg</image_Ref>
      <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">a</name>
    <HasCharacter><Character rdf:ID="Oliver">
      <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Oliver</name>
    <InPanel rdf:resource="#a"/></Character>
    <Character rdf:ID="Diablo">
      <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Diablo</name>
    <InPanel rdf:resource="#a"/></Character></HasCharacter></Panel>
    ...
 </rdf:RDF>
```

Figure 1: Snippet of the image ontology in RDF/OWL format.

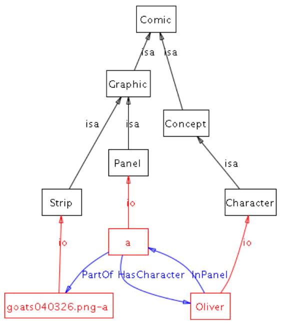


Figure 2: Visual graph of the image ontology. This figure is best viewed in colour. Class nodes are black and instance nodes are red in colour. Relationships — `isa`: is-a; `io`: instance of, `InPanel`, `HasCharacter`, `PartOf`.

successfully recognised the characters. We adopted this ontology structure so the graphic elements of the image collection are separated from the semantic concepts.

Referring to Figure 1, we can see that the character Oliver is in panel "a" of the strip in the RDF/OWL format. The visual graph generated using OntoViz[5] of this RDF/OWL format is depicted in Figure 2. The relation between the strip, panel and character instances are `PartOf` (a panel is part of a strip), `hasCharacter` (panel has character) and `InPanel` (a character is in a particular panel).

5 Queries and Answers

A query to the ontology is done using SPARQL that is available as a built-in component in Protégé.[6] A sample SPARQL query to "Find images depicting Oliver and Diablo" is shown in Figure 3.

The query consists of three parts: the SELECT clause which identifies the criteria to appear in the query results; the WHERE clause specifies the criteria for selecting results from the database; and the FILTER clause restrict the results according the expression. In this case, we want the results that have both characters in the same panel.

This query returns a list of image references that contain the characters Oliver and Diablo. A sample of retrieved images is depicted in Figure 4. This is preliminary work. We have tested the ontology with simple queries and shown that our approach is promising. We plan to extend the ontology to support more complex queries.

6 Conclusion and Future Work

In this paper, we have presented an overview of our approach towards bridging the semantic gap in CBIR. We have built an ontology by incorporating the knowledge

[5] `http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz`

[6] `http://protege.stanford.edu`

```
PREFIX comic:
<http://dayang.cs.rmit.edu.au/~dayang/ComicOntology#>
SELECT ?Panel
FROM
<http://dayang.cs.rmit.edu.au/~dayang/ComicOntology.owl>
WHERE {
?x comic:name ?CharacterName1;
     comic:InPanel  ?Panel.
?y comic:name ?CharacterName2;
     comic:InPanel  ?Panel.

FILTER regex( ?CharacterName1, "^Diablo", "i")
FILTER regex( ?CharacterName2, "^Oliver", "i")
FILTER (?CharacterName1 && ?CharacterName2)
}
```

Figure 3: Example of a SPARQL query to retrieve image panels that contain the character Oliver and Diablo.

learned from our region-based image retrieval method. We have shown that using our image ontology, we are able to pose a query that retrieves desired images containing a particular comic character.

This is a preliminary work; we plan to extend this work to support positioning queries such as "find images of Diablo, where Oliver is stage left", and to adapt it to accommodate real photographic images. Another interesting direction for this work is to implement a visual region-based ontology querying interface.

Acknowledgements This research was undertaken with a scholarship provided by the Malaysian Ministry of Higher Education. We thank Jonathan Rosenberg, author of the Goats Comic for permission to publish the comic images.

References

[1] D. N. F. Awang Iskandar, J. A. Thom and S. M. M. Tahaghoghi. Content-based image retrieval using image regions as query examples. In *Nineteenth Australasian Database Conference (ADC 2008)*, Volume 75 of *CRPIT*, Wollongong, Australia, 2008. ACS. (To appear).

[2] K. Barnard and D.A. Forsyth. Learning the semantics of words and pictures. In *Proceedings of the 8th International Conference on Computer Vision*, pages 408–415, 2001.

[3] C. Brewster and K. O'Hara. Knowledge representation with ontologies: The present and future. *IEEE Intelligent Systems*, Volume 19, Number 1, pages 72–73, January/February 2004.

[4] J. S. Hare, P. H. Lewis, P. G. B. Enser and C. J. Sandom. Mind the gap: Another look at the problem of the semantic gap in image retrieval. In *Proceedings of Multimedia Content Analysis, Management and Retrieval 2006 SPIE*, Volume 6073, pages 607309–1, 2006.

[5] E. Hyvönen, S. Saarela, A. Styrman and K. Viljanen. Ontology-based image retrieval. In *Proceedings of World Wide Web 2003*, May 2003. Poster papers.

[6] E. Hyvönen, S. Saarela and K. Viljanen. Ontogator: Combining view- and ontology-based search with semantic browsing. In *Proceedings of XML Finland 2003*, October 2003.

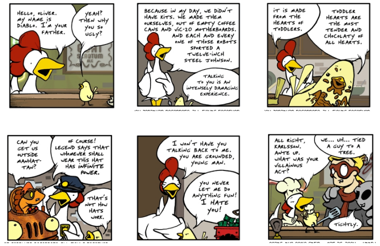

HELLO, OLIVER. MY NAME IS DIABLO. I'M YOUR FATHER.
YEAH? THEN WHY YOU SO UGLY?
BECAUSE IN MY DAY, WE DIDN'T HAVE KITS. WE MADE THEM OURSELVES, OUT OF EMPTY COFFEE CANS AND VIC-20 MOTHERBOARDS. AND EACH AND EVERY ONE OF THOSE ROBOTS SPORTED A TWELVE-INCH STEEL JOHNSON.
TALKING TO YOU IS AN INTENSELY DAMAGING EXPERIENCE.
IT IS MADE FROM THE HEARTS OF TODDLERS.
TODDLER HEARTS ARE THE MOST TENDER AND CHOCOLATY OF ALL HEARTS.
CAN YOU GET US OUTSIDE MANHAT-TAN?
OF COURSE! LEGEND SAYS THAT WHOMEVER SHALL WEAR THIS HAT HAS INFINITE POWER.
THAT'S NOT HOW HATS WORK.
I WON'T HAVE YOU TALKING BACK TO ME. YOU ARE GROUNDED, YOUNG MAN.
YOU NEVER LET ME DO ANYTHING FUN! I HATE YOU!
ALL RIGHT, KARLSSON. ANTE UP. WHAT WAS YOUR VILLAINOUS ACT?
WE... UH... TIED A GUY TO A TREE.
TIGHTLY.

Figure 4: Retrieved images of Oliver and Diablo in the same comic panel. This figure is best viewed in colour.

[7] C. Jelmini and S. Marchand-Maillet. Deva: an extensible ontology-based annotation model for visual document collections. In *SPIE*, Volume 5018, January 2003.

[8] M. L. Kherfi, D. Ziou and A. Benardi. Image retrieval from the world wide web: Issues techniques, and systems. *ACM Computing Surveys*, Volume 36, Number 1, pages 35–67, March 2004.

[9] T. Kurita and T. Kato. Learning of personal visual impression for image database systems. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 547–552, 1993.

[10] S. Liu, L. Chia and S. Chan. Ontology for nature-scene image retrieval. In *CoopIS/DOA/ODBASE (2)*, Volume 3291 of *Lecture Notes in Computer Science*, pages 1050–1061. Springer, 2004.

[11] V. Mezaris, I. Kompatsiaris and M. G. Strintzis. An ontology approach to object-based image retrieval. In *IEEE International Conference on Image Processing (ICIP03)*, pages 511–514, 2003.

[12] V. Mezaris, I. Kompatsiaris and M. G. Strintzis. Region-based image retrieval using an object ontology and relevance feedback. *EURASIP Journal on Applied Signal Processing*, Number 6, pages 886–901, 2004.

[13] A. Th. Schreiber, B. Dubbeldam, J. Wielemaker and B. Wielinga. Ontology-based photo annotation. *IEEE Intelligent Systems*, Volume 16, Number 3, pages 66–74, 2001.

[14] A.W.M Smeulders, M. Worring, S. Santini, A. Gupta and R Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 22, Number 12, pages 1349–1380, Dec 2000.

[15] C. Town. *Ontology based Visual Information Processing*. Ph.D. thesis, University of Cambridge, 2004.

[16] C. Town. Ontological inference for image and video analysis. *Machine Vision and Applications*, Volume 17, Number 2, pages 94 – 115, May 2006.

[17] R. Troncy, J. van Ossenbruggen, J. Z. Pan and G. Stamou. Image annotation on the semantic web, W3C incubator group report, 2007. `http://www.w3.org/2005/Incubator/mmsem/XGR-image-annotation/`.