Focused Retrieval

by

Kalista Yuki Itakura

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2010

Abstract

Traditional information retrieval applications, such as Web search, return atomic units of retrieval, which are generically called "documents". Depending on the application, a document may be a Web page, an email message, a journal article, or any similar object. In contrast to this traditional approach, focused retrieval helps users better pin-point their exact information needs by returning results at the sub-document level. These results may consist of predefined document components — such as pages, sections, and paragraphs — or they may consist of arbitrary passages, comprising any sub-string of a document. If a document is marked up with XML, a focused retrieval system might return individual XML elements or ranges of elements. This thesis proposes and evaluates a number of approaches to focused retrieval, including methods based on XML markup and methods based on arbitrary passages. It considers the best unit of retrieval, explores methods for efficient sub-document retrieval, and evaluates formulae for sub-document scoring. Focused retrieval is also considered in the specific context of the Wikipedia, where methods for automatic vandalism detection and automatic link generation are developed and evaluated.

Acknowledgements

I would like to thank my advisor, Charlie Clarke, examiners Gordon Cormack, Grant Weddell, Mark Smucker, and Arjen DeVries. I would also thank office mates from PLG lab and all other people who fed me.

Dedication

To everyone.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Suppose we want to learn how Japanese art affected Impressionism in Europe. Decades ago, we would go to the library, find the shelf section covering “art history”, browse through book titles, and then pick some books to read. Today, we search on the web with keywords such as “Impressionist and Japanese art”. We obtain some web pages about the topic, but they may be rather short or incomprehensive, so we may still go to the library. Even though we can search book titles in online library catalogues, we may find ourselves in the library, disappointed by the fact a book title was rather misleading for the information we are seeking.

In the ideal world, all information should be available via a quick tapping on the keyboard. Not only would this reduce time to commute, reduce environmental impacts, but it would make information more accessible to people in the rural communities such as Africa. According to 2007 Ontario Public Library statistics [1], there are 4,916,903 active cardholders, visiting libraries 66,520,750 times a year. This is about 13.5 times per person, or a slightly more than once a month commute to the library. There were 115,519,658 direct circulations; thus each time a person visits

[1] `http://www.culture.gov.on.ca/english/library/statistics/statistics2007/2007%20Summary%20and%20Comparison/Summary_and_Comparison_Report_0607.pdf`

Factor	Document Retrieval	Focused Retrieval
Unit of retrieval	document	any part of a document
Burden	User	Search System
Extra-textual Information	Helps	Unknown

Table 1.1: Document Retrieval v.s. Focused Retrieval

a library, the visitor borrows only 1.7 books. People in Africa, however, do not have the luxury to commute to the library. There are only 1,129 library access points among 13 countries in Africa during 1990 -2000 [53], while Ontario alone has 1,093 access points[1]. Even though a free digital library is available to Africans [2] only 8,000 full-text books are available.

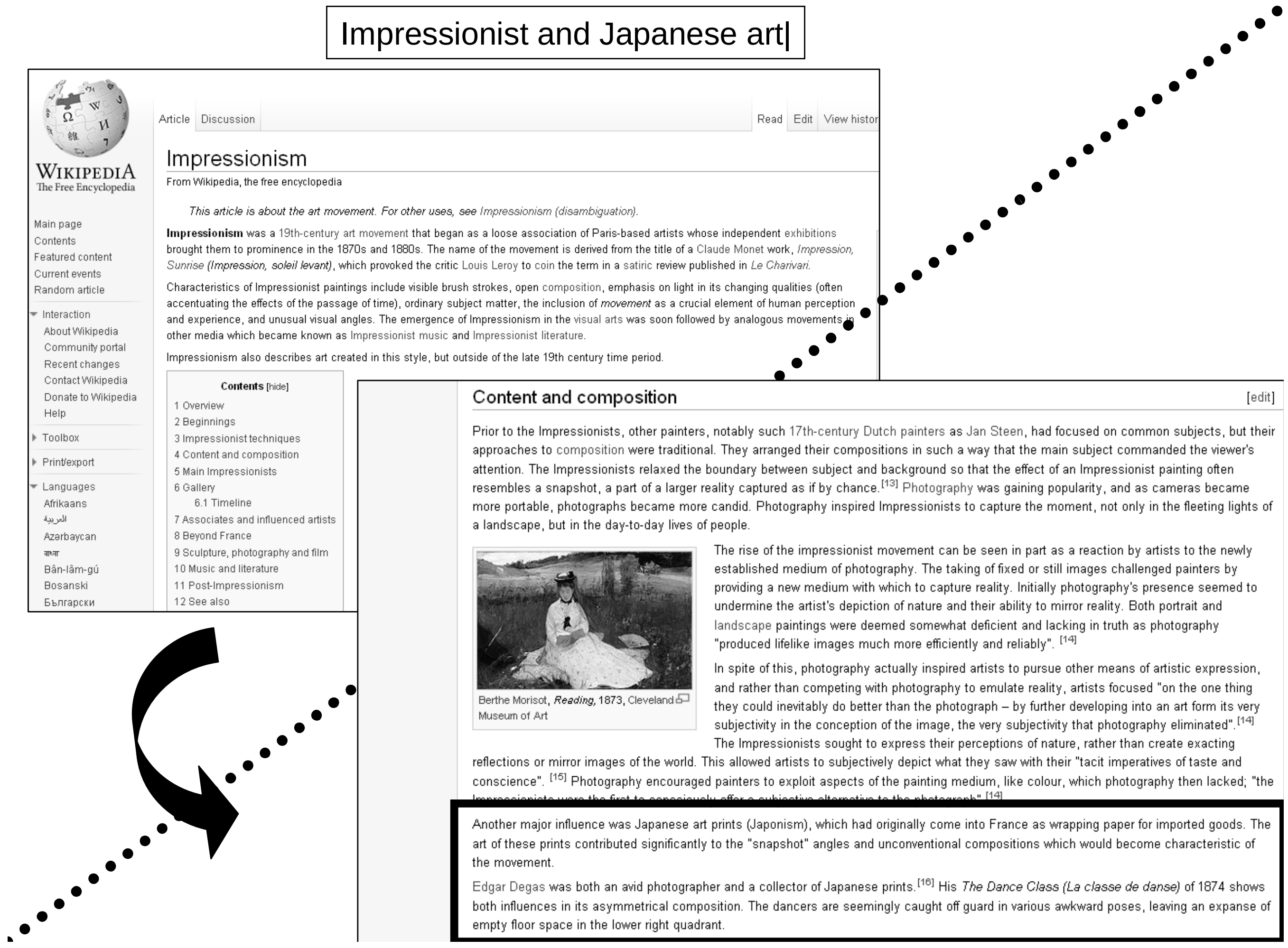
Now, suppose we live in an ideal world. We still need to flip through pages on books online and scroll through the webpages, to find the relation between Impressionism and Japanese art. It would be convenient if there is a single search engine that searches through parts of webpages and books, ranking them together by how relevant the parts are to the query.

My thesis topic is focused retrieval from heterogenous objects. Focused means that we may only be interested in parts of the objects. For example, in Figure 1.1 [3] in the top-most figure, the first search result returned points to the beginning of a document, whereas the bottom-most figure highlights the most relevant part of the same document. Heterogeneous objects implies that we want to retrieve web pages, books, news stories, and other media at the same time. Though these objects could include graphics, audio, video, and similar media, in this thesis I limit my focus to text documents such as journal articles, books and Wikipedia pages.

Focused retrieval offers distinct opportunities and challenges. Table 1.1 shows some of the differences between traditional document retrieval and focused retrieval. First, the unit of retrieval

[2]`http://www.africaeducation.org/adl/`

[3]`http://en.wikipedia.org/wiki/Impressionism`


Impressionist and Japanese art|

Article Discussion Read Edit View histor

WIKIPEDIA The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Interaction
About Wikipedia
Community portal
Recent changes
Contact Wikipedia
Donate to Wikipedia
Help
Toolbox
Print/export
Languages
Afrikaans
العربية
Azərbaycan
বাংলা
Bân-lâm-gú
Bosanski
Български

Impressionism

From Wikipedia, the free encyclopedia

This article is about the art movement. For other uses, see Impressionism (disambiguation).

Impressionism was a 19th-century art movement that began as a loose association of Paris-based artists whose independent exhibitions brought them to prominence in the 1870s and 1880s. The name of the movement is derived from the title of a Claude Monet work, *Impression, Sunrise (Impression, soleil levant)*, which provoked the critic Louis Leroy to coin the term in a satiric review published in *Le Charivari*.

Characteristics of Impressionist paintings include visible brush strokes, open composition, emphasis on light in its changing qualities (often accentuating the effects of the passage of time), ordinary subject matter, the inclusion of *movement* as a crucial element of human perception and experience, and unusual visual angles. The emergence of Impressionism in the visual arts was soon followed by analogous movements in other media which became known as Impressionist music and Impressionist literature.

Impressionism also describes art created in this style, but outside of the late 19th century time period.

Contents [hide]
1 Overview
2 Beginnings
3 Impressionist techniques
4 Content and composition
5 Main Impressionists
6 Gallery
6.1 Timeline
7 Associates and influenced artists
8 Beyond France
9 Sculpture, photography and film
10 Music and literature
11 Post-Impressionism
12 See also

Content and composition [edit]

Prior to the Impressionists, other painters, notably such 17th-century Dutch painters as Jan Steen, had focused on common subjects, but their approaches to composition were traditional. They arranged their compositions in such a way that the main subject commanded the viewer's attention. The Impressionists relaxed the boundary between subject and background so that the effect of an Impressionist painting often resembles a snapshot, a part of a larger reality captured as if by chance.[13] Photography was gaining popularity, and as cameras became more portable, photographs became more candid. Photography inspired Impressionists to capture the moment, not only in the fleeting lights of a landscape, but in the day-to-day lives of people.

Berthe Morisot, *Reading*, 1873, Cleveland Museum of Art

The rise of the impressionist movement can be seen in part as a reaction by artists to the newly established medium of photography. The taking of fixed or still images challenged painters by providing a new medium with which to capture reality. Initially photography's presence seemed to undermine the artist's depiction of nature and their ability to mirror reality. Both portrait and landscape paintings were deemed somewhat deficient and lacking in truth as photography "produced lifelike images much more efficiently and reliably".[14]

In spite of this, photography actually inspired artists to pursue other means of artistic expression, and rather than competing with photography to emulate reality, artists focused "on the one thing they could inevitably do better than the photograph – by further developing into an art form its very subjectivity in the conception of the image, the very subjectivity that photography eliminated".[14] The Impressionists sought to express their perceptions of nature, rather than create exacting reflections or mirror images of the world. This allowed artists to subjectively depict what they saw with their "tacit imperatives of taste and conscience".[15] Photography encouraged painters to exploit aspects of the painting medium, like colour, which photography then lacked; "the

Another major influence was Japanese art prints (Japonism), which had originally come into France as wrapping paper for imported goods. The art of these prints contributed significantly to the "snapshot" angles and unconventional compositions which would become characteristic of the movement.

Edgar Degas was both an avid photographer and a collector of Japanese prints.[16] His *The Dance Class (La classe de danse)* of 1874 shows both influences in its asymmetrical composition. The dancers are seemingly caught off guard in various awkward poses, leaving an expanse of empty floor space in the lower right quadrant.

Figure 1.1: Document Retrieval v.s. Focused Retrieval

is different. Document retrieval only returns entire documents, whereas focused retrieval can return any part of any document. This part returned can range from a single character to a sequence of sections, to an entire document. The ideal size for focused retrieval then must balance *exhaustivity*, the measure of the amount of pertinent information, with *specificity*, the measure of the quality of pertinent information. For example, in Figure 1.1, the top-most result has high exhaustivity with respect to the query by displaying an entire article about Impressionism. The top-most result, however, lacks specificity because most information is not pertinent to the "Japanese art" part of the query. The first sentence of the bottom result, on the other hand, has high specificity because it gives a one-sentence answer to the question as to how Japanese art influenced Impressionism. This sentence alone, however, lacks exhaustivity because it does not give supplementary information. Thus for this query, the ideal result is the paragraphs surrounded by the box.

The second difference between document retrieval and focused retrieval is associated with the burden of locating information in documents. In document retrieval, a search engine only needs to score at most the number of documents in the corpus, but a user needs to spend time scanning to spot the part of document that is relevant to their information need. Focused retrieval, on the other hand, may need to score as many times as the number of parts of documents (or *subdocuments*) in the collection. A user, however, does not need to scan and spot the relevant part of documents.

For example, there are 2,666,500 documents in one of the corpora I used for experiments reported in this thesis [4]. The average document length in this Corpus is 1,564 terms, including XML tags. In document retrieval, the scoring function is executed 2,666,500 times. In focused retrieval, if we consider all subdocuments that start and end at a term, the function could be executed $(1564 \cdot 1565/2)\,266500 = 3,263,342,695,000$ times. The number gets worse when documents are longer, such as in the INEX 2008 Book Corpus [40, 77], which I used in some experiments.

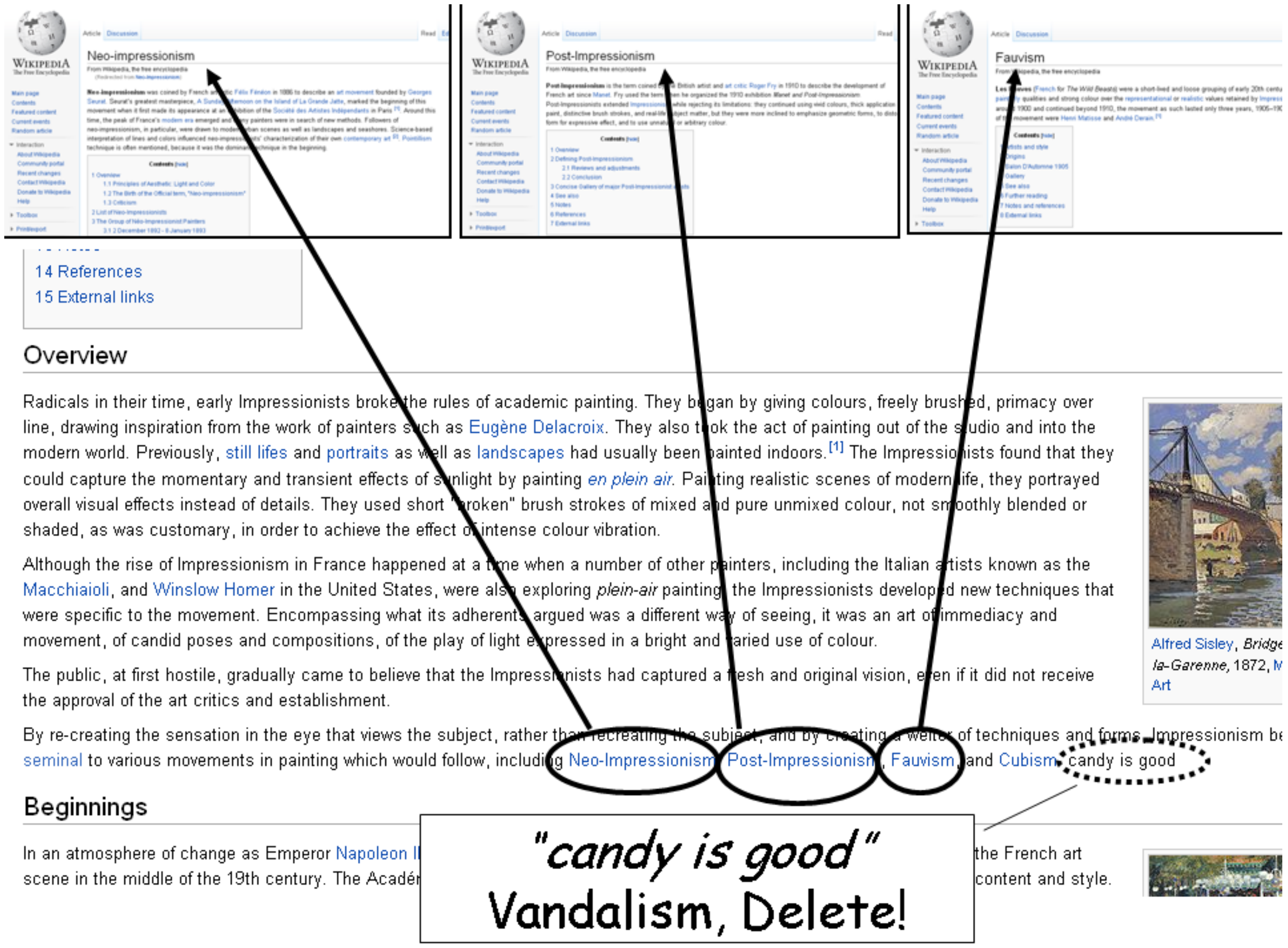
[4]INEX 2009 Wikipedia Corpus [125]

This corpus contains 50,238 books with an average document length of 143,037 words including tags, which could correspond to 513,927,867,592,314 subdocuments, or 10,229,863,203 times more executions of a scoring function than would be required for standard document scoring. In general, if there are k documents, each with an average length of n words, there may be $(n \cdot (n-1)/2)\, k = \mathcal{O}\left(kn^2\right)$ possible subdocuments, thus when compared to a document retrieval system that executes a scoring function k times, there is a quadratic increase in the running time. If the book search engine resides in a large corporate computers, and the collection is relatively small, these numbers may be acceptable. But what if a user wanted to search on an electronic book reader, such as a Kindle [5]? For portability, electronic book readers boast light weight and long battery life. For example, the above mentioned Kindle website as of August 6th, 2010, boasts a one-month battery life and a weight of less than a paperback novel. A quadratic increase in the processing time would likely be impossible for these devices. Thus, the document search engines may generate a much lighter processing load than focused search engines. Users, however, may find that focused retrieval requires less effort because they do not need to go through the document to locate information of interest.

Finally, document retrieval draws on an abundance of research to help improve the search results. For example, using link structure in the computation of document scores, using methods such as PageRank [6], helps search engines to take into account extra-textual context. Applying XML structure to text such as BM25F [120] may also help with web page retrieval. However, in focused retrieval, treating each subdocuments (sections, chapters, pages, paragraphs) as an independent document appears to be essentially the state-of-the-art [73].

Focused retrieval from heterogeneous collection means more than just having a search engine returning parts of webpages and books mixed together. Table 1.2 lists possible applications. In a link detection problem, at the document level, we only need to "retrieve documents" related to the document of interest. At the subdocument level, we need to retrieve a part of a document, an

[5]`http://www.amazon.com/gp/product/B003FSUDM4/ref=sv_kinc_0`



14 References
15 External links

Overview

Radicals in their time, early Impressionists broke the rules of academic painting. They began by giving colours, freely brushed, primacy over line, drawing inspiration from the work of painters such as Eugène Delacroix. They also took the act of painting out of the studio and into the modern world. Previously, still lifes and portraits as well as landscapes had usually been painted indoors.[1] The Impressionists found that they could capture the momentary and transient effects of sunlight by painting *en plein air*. Painting realistic scenes of modern life, they portrayed overall visual effects instead of details. They used short "broken" brush strokes of mixed and pure unmixed colour, not smoothly blended or shaded, as was customary, in order to achieve the effect of intense colour vibration.

Although the rise of Impressionism in France happened at a time when a number of other painters, including the Italian artists known as the Macchiaioli, and Winslow Homer in the United States, were also exploring *plein-air* painting, the Impressionists developed new techniques that were specific to the movement. Encompassing what its adherents argued was a different way of seeing, it was an art of immediacy and movement, of candid poses and compositions, of the play of light expressed in a bright and varied use of colour.

The public, at first hostile, gradually came to believe that the Impressionists had captured a fresh and original vision, even if it did not receive the approval of the art critics and establishment.

By re-creating the sensation in the eye that views the subject, rather than recreating the subject, and by creating a welter of techniques and forms, Impressionism be seminal to various movements in painting which would follow, including Neo-Impressionism, Post-Impressionism, Fauvism, and Cubism. candy is good

Alfred Sisley, *Bridge* *la-Garenne*, 1872, M Art

Beginnings

In an atmosphere of change as Emperor Napoleon II the French art scene in the middle of the 19th century. The Acadé content and style.

Figure 1.2: Focused Retrieval Applications

Application	Document Retrieval	Focused Retrieval
Link Detection	Document Linking	Identification of anchor phrase and the BEP
Spam Detection	Email	Wikipedia Pages

Table 1.2: Focused Retrieval Application

anchor phrase, to link to another part of a document, the best-entry-point (BEP) for reading. For example, in Figure 1.2 [6], instead of linking the entire document on "Impressionism" to related documents on "Neo-Impressionism" [7], "Post-Impressionism" [8], and "Fauvism" [9], anchor phrases such as "Neo-Impressionism" and the best entry points of the link destination, in these cases, the beginnings of the documents, are selected.

In many spam detection applications the rules of document retrieval may apply. For email and Web pages, we identify whole documents considered to be spam. However, in some applications spam detection may be required at the subdocument level. For example, vandalism (spam) in the Wikipedia must be identified at the subdocument level, as each edit is made. In Figure 1.2, a document-level spam detection might deem the entire article on "Impressionism" to be spam, whereas in focused spam detection, only the "candy is good" part is deemed to be vandalism because it does not follow from the previous sentence regarding various artistic movements influenced by Impressionism.

1.1 INEX

[6]`http://en.wikipedia.org/w/index.php?title=Impressionism&oldid=366378612`

[7]`http://en.wikipedia.org/wiki/Neo-Impressionism`

[8]`http://en.wikipedia.org/wiki/Post-Impressionism`

[9]`http://en.wikipedia.org/wiki/Fauvism`

Most of the experiments I report in this thesis are conducted as part of an XML retrieval workshop, *INEX*, the INitiative for the Evaluation of XML retrieval [10]. INEX is similar to the well-known Text REtrieval Conference (TREC) [11] in the sense that for each search task, called a *track*, participants are given a set of test queries called *topics* and a corpus to which they submit a run to compare their performance to one another. The goal of INEX builds upon TREC in the sense that its purpose is to test the utility of applying an XML structure to information retrieval. The workshop consists of several tracks, each track is divided into tasks, again with different objectives. The participants are required to create topics, submit the results of their experiments, and assess the results. The final scores for each track/task are computed by evaluation measures chosen by the organizers and the ranks are published according to the scores.

Between 2002 and 2009, INEX expanded from a single Ad Hoc Track and less than 30 participating groups [32], to eight tracks and roughly 50 participating groups mostly from Europe and Australia/New Zealand [47]. The current iteration, INEX 2010, features an Ad Hoc Track, a Book Track, a Data-Centric Track, an Interactive Track, a Link-the-Wiki Track, a Question Answering Track, a Relevance Feedback Track, a Web-Service Discovery Track, and a XML Mining Track. Of these tracks, my thesis relates to the Ad Hoc, Book, and Link-the-Wiki Tracks.

The purpose of the Ad Hoc Track is subdocument retrieval. Over the years, I participated in the following tasks: The Focused Task, Best-in-Context Task, and Relevant-in-Context Task. Focused Task, the center of my thesis, retrieves subdocuments that balance specificity and exhaustivity. In 2006, the corpus expanded from the 500MB IEEE collection donated by the IEEE Computer Society in 2002, to the 6GB Wikipedia collection [25] created in 2006. In 2009, the corpus was changed to a 50GB Wikipedia collection [125]. The set of tags expanded from basic structural tags such as `<section>` and `<p>`, to semantic tags such as `<person>` and `<country>` in 2009.

[10]`http://www.inex.otago.ac.nz/`

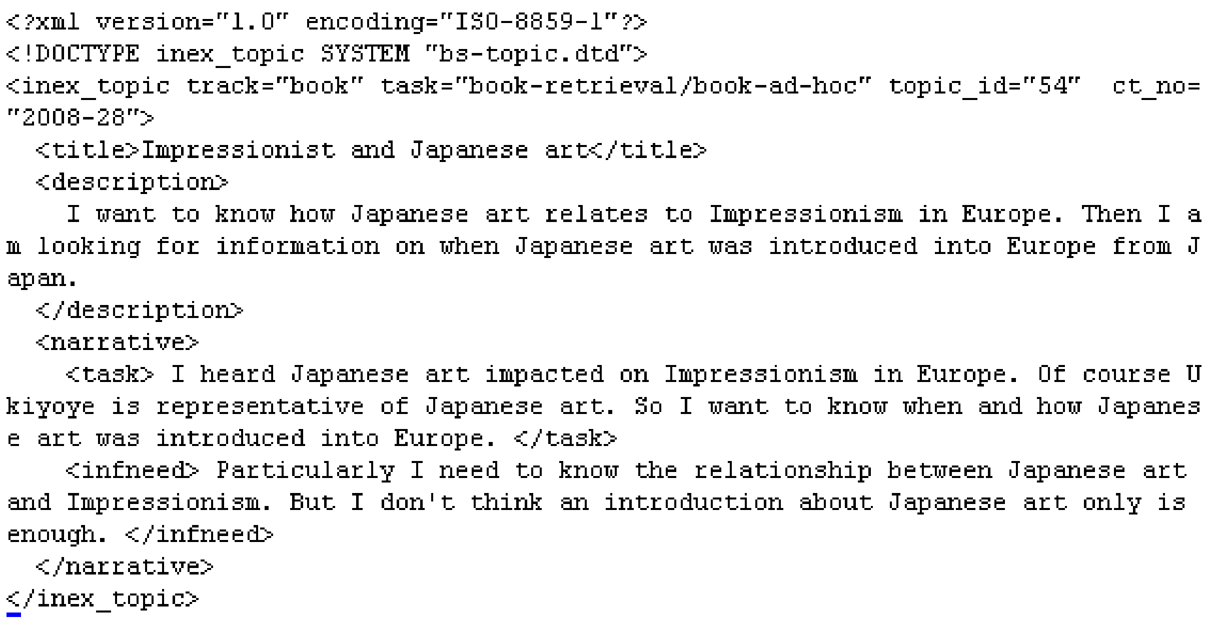
[11]`trec.nist.gov/`

Until 2007, participants were restricted to return single XML elements (a paragraph, section, abstract, etc.). This restriction was lifted in 2007, when INEX allowed participants to return subdocuments of any size to facilitate the comparison between XML and non-XML oriented retrieval strategy [33]. The evaluation measures were changed accordingly from those that take structure into account, to the simple comparison of the ratio of assessor's highlighted text to the retrieved text [33]. As of 2009, no XML-oriented search strategy has significantly outperformed a non-XML-oriented strategy [33, 45, 73].

The Book Track's goal is to facilitate digitization of books. As part of my thesis work, I have participated in the Book Retrieval Task and the Focused Search Task. The former can be considered as a book version of web retrieval, and the latter a book version of focused search. The major difference between the tasks at Ad Hoc Track and the Book Track is the sheer size of corpus. The book corpus [40, 77] (since 2008) consists of 50239 out-of-copyright books with a total size of 37GB after pre-processing. The original size was 250GB, but each word was wrapped with a `<word>` tag. Additionally, the number of XML tags are limited in the book corpus.

The topic format for Ad Hoc and Book Tracks are similar. Fig 1.3 shows a sample topic taken from INEX 2008 Book Track test set. The title field displays what a user would type into a search box. In this example, it is "Impressionist and Japanese art". The title field is followed by a description field, which may be used as a natural language query. The narrative field describes the background on the query, detailing why a user needs the information. In this example, the user became interested in the relationships between Japanese art and Impressionism after hearing about it. In addition to these fields, Ad Hoc topics contain the content-and-structure title field, which specifies a structural requirement for the retrieval.

For both Ad Hoc and Book Tracks, relevance assessments were conducted by users highlighting the relevant part of the document/book pages. For the Ad Hoc Track, the submitted runs were pooled and the documents in the pool were submitted to a participant, usually the topic creator,



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE inex_topic SYSTEM "bs-topic.dtd">
<inex_topic track="book" task="book-retrieval/book-ad-hoc" topic_id="54"  ct_no=
"2008-28">
  <title>Impressionist and Japanese art</title>
  <description>
    I want to know how Japanese art relates to Impressionism in Europe. Then I a
m looking for information on when Japanese art was introduced into Europe from J
apan.
  </description>
  <narrative>
    <task> I heard Japanese art impacted on Impressionism in Europe. Of course U
kiyoye is representative of Japanese art. So I want to know when and how Japanes
e art was introduced into Europe. </task>
    <infneed> Particularly I need to know the relationship between Japanese art
and Impressionism. But I don't think an introduction about Japanese art only is
enough. </infneed>
  </narrative>
</inex_topic>
```

Figure 1.3: INEX 2008 Book Track Topic 54

for assessment [33, 45, 73]. For the Book Track, in addition to highlighting, buttons were clicked to indicate if the pages and books are relevant or not [40, 77].

The goal of the Link-the-Wiki Track is the automatic identification of anchor phrases and best entry points for a plain text document. In 2008, the 6GB Wikipedia corpus, in 2009, the 50GB Wikipedia corpus, and in 2010, the teAra corpus [12] was used. For 2008 and 2009, the corpus came with pre-existing links, which participants could use to their advantage. In 2010, the corpus did not include any markup.

1.2 Contributions of my Thesis

My contributions may be divided into INEX-based contributions, and other published and

[12]www.teara.govt.nz

non-published contributions. The major contributions from INEX help to resolve two open problems in the Ad Hoc Track, Focused Task, and Link-the-Wiki Tracks. The non-INEX-based contributions fall into the area of focused retrieval not studied in INEX, completing a comprehensive study of focused retrieval. My publications are listed as references to this thesis [64–69].

1.2.1 INEX-based Contributions

In INEX 2007, 2008, and 2009, I participated in Ad Hoc, Book, Entity Ranking (not part of the thesis) , and Link-the-Wiki Tracks. My contributions are to study the unit of retrieval in Ad Hoc Track, to study the utility of XML structure in Ad Hoc and Book Tracks, and to present a simple baseline run for Link-the-Wiki Track that has proven difficult to beat. Because some of my contributions provide difficult-to-beat performance in Ad Hoc and Link-the-Wiki Tracks, the organizers changed the tasks in the two tracks in INEX 2010 [13].

For reference, Table 1.3 summarizes my participation and the best rank achieved by my runs.

Unit of Retrieval

At INEX, I raised an issue that a single element may not be a good unit of retrieval. Specifically, in Itakura and Clarke [64] we propose that converting a passage result into a single element result in Ad Hoc Focused retrieval by taking the smallest element that contains the most relevant passage, performs worse than simply scoring all single elements and returning the best scoring elements. Thus, I propose that the then-restriction on the retrieval unit should be liberated to include ranges of elements. In INEX 2007, the unit of retrieval was liberated into arbitrary

[13]INEX 2010 Ad Hoc Track Ad Hoc Retrieval Task and Result Submission Specification `http://www.inex.otago.ac.nz/tracks/adhoc/runsubmission.asp?action=specification`

track	year	task	rank	# groups
Ad Hoc	2007	Focused	3	26
		Best-in-Context	3	19
	2008	Focused	1	19
		Best-in-Context	1	13
		Relevant-in-Context	1	11
	2009	Focused	1	16
Link-the-Wiki	2007	Outgoing	4	4
		Outgoing (Unofficial)	1	4
		Incoming	1	4
	2008	Outgoing 6600 File2File (Wikipedia)	3	8
		Incoming 6600 File2File (Wikipedia)	2	9
		Outgoing 50 A2BbyA2F, Single Destination(manual)	2	8
	2009	Outgoing 33 A2B (Wikipedia)	2	4
		Incoming 33 A2B (Wikipedia)	1	2
		Outgoing 33 A2B (Manual)	1	4
Book	2008	Book Search	4	4
		Page-in-Context	1	2
	2009	Focused search (results not available yet)	n/a	n/a

Table 1.3: Official INEX Ranks 2007-2009. Sources: Overview Papers [33, 45, 57, 58, 60, 73, 77]

passages [33]. In INEX 2008, I submitted arbitrary-passage retrieval runs that, surprisingly, did not perform as well as my single-element retrieval runs [45,66]. Chapter 5 Section 5.3 studies the issue further and arrive at the conclusion that (at least with the current Ad Hoc Track setting) single XML elements are good retrieval units.

Taking Advantage of Structure

My basic single-element retrieval runs that does not take XML structure into account (other than defining which element to score) ranked well in INEX 2007 and 2008 [33, 65, 66, 73]. At INEX 2009, however, I used a version of BM25F [120] to increase the performance over the baseline on both Ad Hoc and Book Tracks [45,69]. In Itakura and Clarke [68], I perform further experiments on the INEX 2009's successful submission on book focused search settings and show that it gives improvements over the baseline that has consistently ranked high in the INEX Ad Hoc Focused Task. I also present the generalized framework for how to use XML markup in Ad Hoc retrieval without having too many fields to overload computations as in previous work [92,119]. Chapter 6 also discusses the topic.

Focused Links

In the INEX 2007 Link-the-Wiki track, I submitted an unofficial run [14] that used a simple statistical approach that surpassed everyone else's run [57, 65]. This simple statistical baseline approach was later adopted and modified slightly by other participants to prove a difficult-to-beat performance in INEX 2008 and 2009 [58,60,66,69]. In 2009, in order to improve on this baseline, I used K-L divergence to balance the popularity of a page with the topicality of the page.

[14]The official run I submitted assumed that the links are not ranked. After the evaluation results and software was released before the workshop, however, I learned that the links needed to be ranked. Thus I sorted the links in the official run and presented the new unofficial run during INEX 2007.

In INEX 2009, this new approach gave an improvement over the baseline at early recall points [60, 69]. However, the baseline run looked already near-perfect and difficult-to-beat that in INEX 2010, the organizers changed the objective of Link-the-Wiki track. Chapter 7 discusses my approach to the focused link problem.

1.2.2 Non-INEX Contributions

Some of the important differences between document and focused search are not yet studied at INEX. One question is the increased complexity inherent to focused retrieval. The other question is focused spam filtering.

Mathematical Framework

Chapter 3 first defines what a focused retrieval means. In order to reduce the brute-force complexity of $\mathcal{O}\left(n^2\right)$ quadratically, I devise a simple scoring function that is based on pivoted length normalization [128]. Next, I derive two famous scoring functions, BM25 [121] and language model [141] into a form similar to my function so that the reduction becomes applicable to them and to show that they both possess the similar ranking formula. While this work does not fully solve the problem of quadratic performance, it provides a framework for the problem, as well as providing a formal foundation for the rest of the thesis.

Focused Spam Detection

Chapter 8 discusses my work reported in Itakura and Clarke [67]. In this work, I apply a simple probabilistic compression model, previously employed for email spam detection, to the problem of Wikipedia vandalism detection. I show, contrary to a previous work [129], that a simple compression model can work in this limited setting.

1.3 Summary

In this thesis, I explore the topic of focused retrieval from heterogeneous objects. Focused retrieval means not ad hoc retrieval of subdocuments, but also retrieval of anchor phrases and the best-entry-points, as well as the identification of vandalism in otherwise proper text. The heterogeneous objects I study are Wikipedia articles and book pages, though the results may not necessarily be limited to those types.

For ad hoc focused retrieval, I study three main issues.

- **Computational Overload:** "Is there a way to reduce it?"
- **Unit of Retrieval:** "What is the best passage size?"
- **Extra-textual Information:** "Can we take advantage of it?"

The first two issues would not be a problem in traditional document retrieval settings. The last issue has been investigated extensively in document retrieval, producing improvement by using external information. In focused retrieval settings, however, despite being the main objective of INEX since 2002, no one has been able to clearly resolve the advantage of using XML structure.

I base most of my experiments on an XML retrieval workshop, INEX. The official runs I submitted from 2007 to 2009 are evaluated objectively by evaluation metrics chosen by the INEX coordinators. Some of my runs have performed very well, and one run was re-implemented by other participants in the following years, producing similar results. My runs are downloadable from the workshop website [15].

The roadmap of my thesis is as follows: Chapter 2 lays out previous work on document retrieval, passage retrieval, and focused retrieval, including INEX runs. Chapter 3 frames the

[15] `http://www.inex.otago.ac.nz/`

problem of passage retrieval formally and tries to derive a linear time algorithm to solve the problem. Chapter 4 describes implementation details of my approach to focused retrieval tasks. Chapter 5 discusses the issue of determining the ideal unit of retrieval in focused ad hoc search. Chapter 6 investigates how to take advantage of structures in ad hoc retrieval. Chapter 7 describes one application of focused search, automatic linking of Wikipedia pages. Chapter 8 describes another focused search application, vandalism detection in Wikipedia. Chapter 9 concludes the thesis.

Chapter 2

Related Work

2.1 Document Retrieval

In almost all IR tasks, the unit of retrieval is fixed: I call these indivisible units "documents", although they may be email messages, articles, etc. Each document is considered independent and ranked using relevance scoring functions such as the vector space model [124], BM25 [122], and the language modelling approach to IR [114].

The Vector space model (VSM) is the simplest of all these scoring functions. Typically, a document is represented by a vector of real numbers. Each entry in the vector corresponds to the term frequencies of the terms in a corpus. Similarly, a query vector is formed based on term frequencies of query terms. Ranking of a document is according to how close a document vector is, compared to the query vector. Normally the similarity measure is computed by cosine distance [124]. The drawback of this model is that it does not discriminate document length well, where in practice, given the same term frequencies, short documents may be preferred. For example, if a Wikipedia article on “Impressionism” contained all the query terms in the first paragraph, then the VSM scores of the article and the first paragraph would be the same, but the users may prefer reading the paragraph over the entire article.

Pivoted length normalization can normalize the length disparities. Singhal et al. propose that a good document retrieval algorithm should produce a retrieval probability of a document that closely matches the actual relevance of the document [128]. They show that existing document retrieval scoring algorithms that normalize length can be normalized even further by the application of pivoted length normalization.

The basic idea behind their pivoted length normalization is that given a graph that plots the probabilities of retrieval and relevance of documents of varying length, we can tilt the retrieval probability plot at some point (pivot) for some degree (slope) to make the retrieval probability plot closer to the relevance probability plot. Thus the basic pivoted length normalization formula involves the original normalization formula such as a logarithm of a length, and two variables, a pivot p, and a slope s,

$$S\left(\text{document}\right) = \sum_{t} \frac{f\left(f_t\right) \cdot g\left(w_t\right)}{\left(1.0 - s\right) \times p + s \times \text{old\_normalization}} \ , \tag{2.1}$$

[128] where $f\left(f_t\right)$ and $g\left(w_t\right)$ are functions of a term frequency and a weight for a query term t [128]. In the next section, I define the mathematical model of passage retrieval and I revisit (2.1) in our passage retrieval setting.

BM25 normalizes document lengths to counter the repeated appearance of query terms. The basic idea is that a few appearances of a high IDF term establish that the document is (probably) “about” that term. Additional appearances does not change the “aboutness” much [122]. For example, in the Wikipedia article on “Impressionism”, after the first few times the reader sees the terms in the article, the reader would think the article is about the Impressionism. The user is unlikely to change mind after seeing the term for 791025 more times in the same article. The version I use in this thesis is as follows:

$$S\left(\text{document}\right) = \sum_t w_t \frac{f_t\left(k_1 + 1\right)}{f_t + k_1\left(1 - b + b\frac{l}{avgl}\right)} \quad , \tag{2.2}$$

where t is a query term, w_t is an inverted document frequency for t, f_t is a term frequency of t in the document, and $avgl$ is an average document length in the collection. Parameters $k_1 > 0$ and $b = \{b : 0 < b < 1\}$ are tuned to optimize the scoring scheme.

Language modeling also normalizes document length and scores documents using a different paradigm than the first two, which are variations of counting term frequencies. In language modeling-based document retrieval, a document's probability to produce a query is computed using the document's language model [114]. The version I use in this thesis is as follows:

$$S\left(\text{document}\right) = \sum_t \log\left(1 + \frac{f_t}{\mu}\frac{l_{\mathcal{C}}}{n_t}\right) - m\log\left(1 + \frac{l}{\mu}\right) \quad , \tag{2.3}$$

where μ is a parameter, $l_{\mathcal{C}}$ is the length of the corpus $\mathcal{C}$, n_t is frequency of t in the entire corpus $\mathcal{C}$, and m is the number of terms in the query.

In Chapter 3 I examine the details of BM25 and language modeling.

Finally, although not used in the thesis, a recent trend of ranking documents is to use a machine-learning approach, called “learning-to-rank”. The approach learns a ranking function through a set of training data along with its relevance judgements. The queries and documents are represented by features to train on a machine learning algorithm [90].

2.2 Focused Retrieval

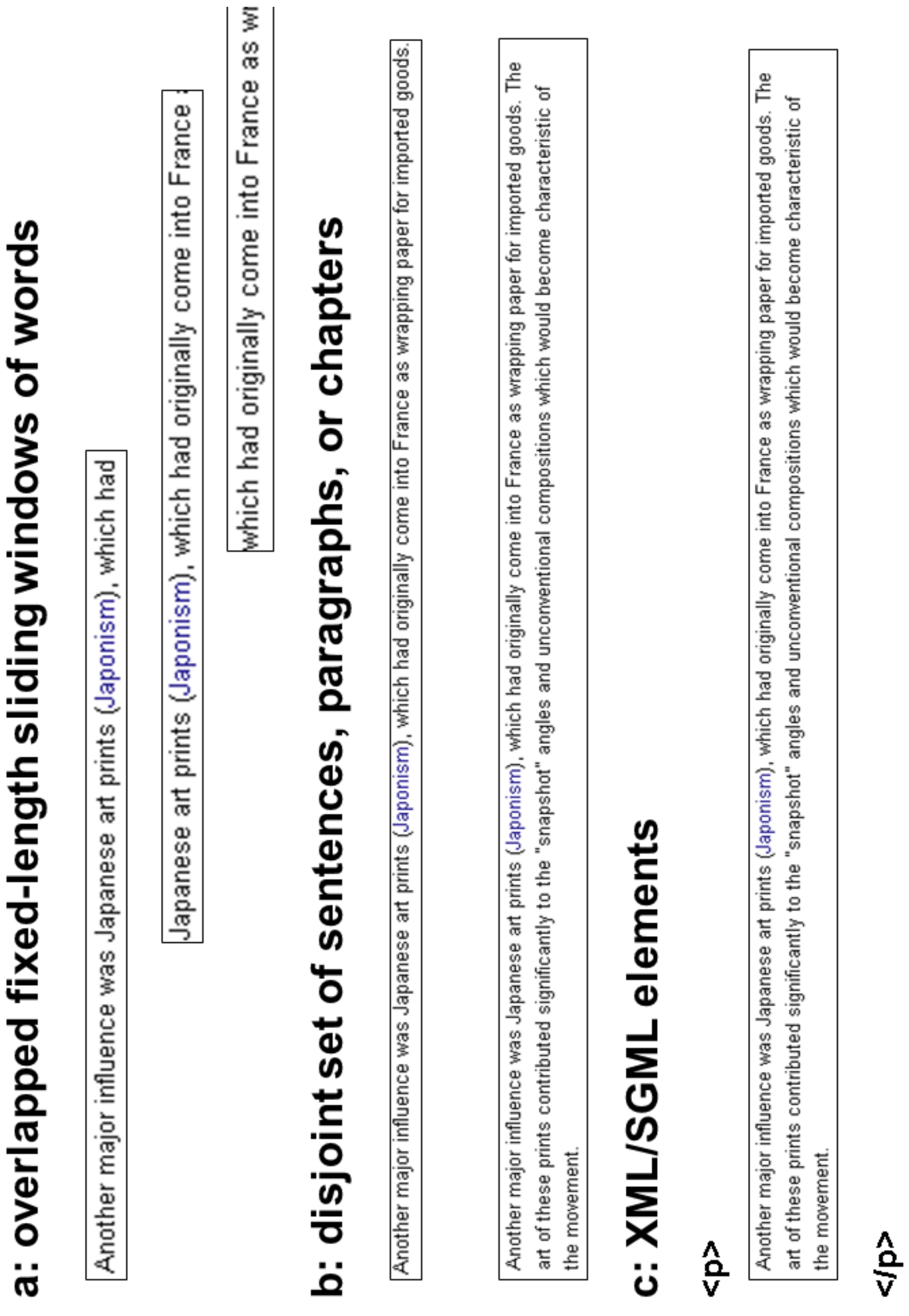
The goal of focused retrieval is to return to a user document parts that balances specificity with exhaustivity. A whole Wikipedia page about Impressionism is too long for a busy user to skim through to find information about how Japanese arts influenced it. But a phrase “Japanese arts” in the same Wikipedia page is too short for a user to understand how it actually influenced the art movement. Chapter 1 proposed three problems inherent to focused retrieval, the unit of retrieval, usefulness of (XML) markup, and complexity. In each of the following sections, I discuss related work that involves these three issues.

2.2.1 The Best Passage Size

The majority of previous work on focused retrieval assumes a fixed unit of retrieval. For example, sliding windows of words give fixed-length passages. Simply scoring XML elements do not have fixed length, but fixed structure. We call this *predefined passage retrieval.* Figure 2.1 [1] displays different types of predefined passages from the Wikipedia article on "Impressionism". The unit can be overlapped fixed-length sliding windows of words [8,81,91,103] (Figure 2.1-a), or sentences [1], which is very common in question answering tasks. The unit can be also a disjoint set of sentences, paragraphs, or chapters [8,22,55,56,85,86,111,118,143] (Figure 2.1-b), based on document markup such as XML [3,126] (Figure 2.1-c)and SGML [110,143].

In general, the more flexible the unit is, the better the performance [8,81,91,103]. Using a sliding window of words in Fig 2.1-a would perform better than using paragraphs in Fig 2.1-b as a unit if the window length was small because sliding window of words normally produces more number of passages than the number of paragraphs. Callan [8] experiments on retrieval of passages based on paragraphs and compare it against sliding windows of words and find that sliding windows of words perform better than paragraphs. Liu and Croft [91] compares the performance of half-overlapped sliding windows of words against a simulated arbitrary passage retrieval, in which they set the length of passage size to be in a certain length and compute a passage starting at every 25 words. Though the simulated arbitrary passage retrieval is still a variation of sliding windows, it produces more varieties of passages than the half-overalpped

[1]Excerpt from `http://en.wikipedia.org/wiki/Impressionism` taken on July 7, 2010


a: overlapped fixed-length sliding windows of words

Another major influence was Japanese art prints (Japonism), which had

Japanese art prints (Japonism), which had originally come into France

which had originally come into France as w

b: disjoint set of sentences, paragraphs, or chapters

Another major influence was Japanese art prints (Japonism), which had originally come into France as wrapping paper for imported goods.

Another major influence was Japanese art prints (Japonism), which had originally come into France as wrapping paper for imported goods. The art of these prints contributed significantly to the "snapshot" angles and unconventional compositions which would become characteristic of the movement.

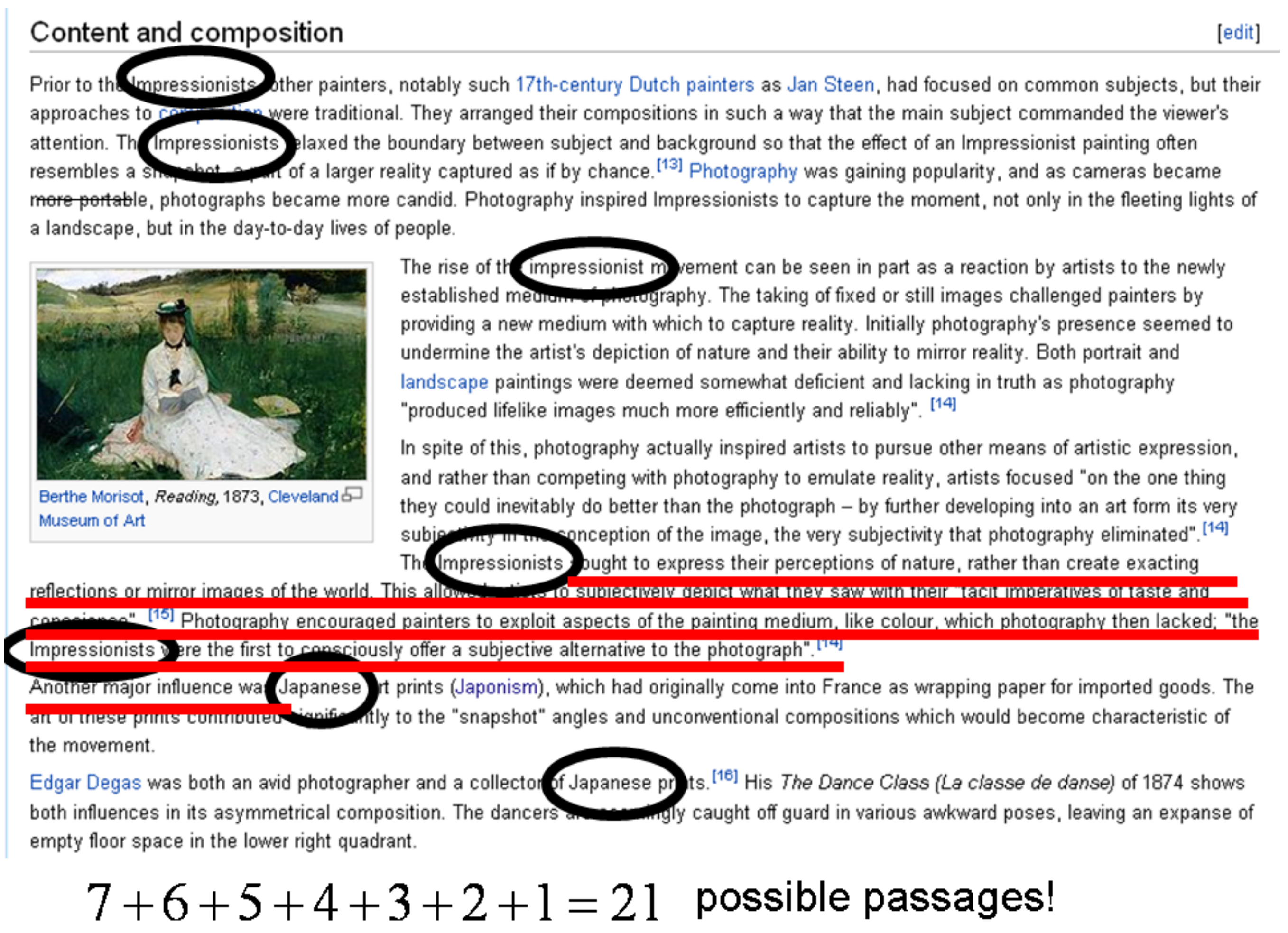
c: XML/SGML elements

<p>

Another major influence was Japanese art prints (Japonism), which had originally come into France as wrapping paper for imported goods. The art of these prints contributed significantly to the "snapshot" angles and unconventional compositions which would become characteristic of the movement.

</p>

Figure 2.1: Predefined Passage Retrieval



Content and composition

[edit]

Prior to the Impressionists, other painters, notably such 17th-century Dutch painters as Jan Steen, had focused on common subjects, but their approaches to composition were traditional. They arranged their compositions in such a way that the main subject commanded the viewer's attention. The Impressionists relaxed the boundary between subject and background so that the effect of an Impressionist painting often resembles a snapshot, a part of a larger reality captured as if by chance.[13] Photography was gaining popularity, and as cameras became ~~more portable~~, photographs became more candid. Photography inspired Impressionists to capture the moment, not only in the fleeting lights of a landscape, but in the day-to-day lives of people.

Berthe Morisot, *Reading*, 1873, Cleveland Museum of Art

The rise of the impressionist movement can be seen in part as a reaction by artists to the newly established medium of photography. The taking of fixed or still images challenged painters by providing a new medium with which to capture reality. Initially photography's presence seemed to undermine the artist's depiction of nature and their ability to mirror reality. Both portrait and landscape paintings were deemed somewhat deficient and lacking in truth as photography "produced lifelike images much more efficiently and reliably". [14]

In spite of this, photography actually inspired artists to pursue other means of artistic expression, and rather than competing with photography to emulate reality, artists focused "on the one thing they could inevitably do better than the photograph – by further developing into an art form its very subjectivity in the conception of the image, the very subjectivity that photography eliminated".[14] The Impressionists sought to express their perceptions of nature, rather than create exacting reflections or mirror images of the world. This allowed artists to subjectively depict what they saw with their "tacit imperatives of taste and conscience".[15] Photography encouraged painters to exploit aspects of the painting medium, like colour, which photography then lacked; "the Impressionists were the first to consciously offer a subjective alternative to the photograph".[14]

Another major influence was Japanese art prints (Japonism), which had originally come into France as wrapping paper for imported goods. The art of these prints contributed significantly to the "snapshot" angles and unconventional compositions which would become characteristic of the movement.

Edgar Degas was both an avid photographer and a collector of Japanese prints.[16] His *The Dance Class (La classe de danse)* of 1874 shows both influences in its asymmetrical composition. The dancers are seemingly caught off guard in various awkward poses, leaving an expanse of empty floor space in the lower right quadrant.

$7+6+5+4+3+2+1=21$ possible passages!

Figure 2.2: Arbitrary Passage Retrieval

windows, giving superior performance.

The above observation is especially useful when the query size is small, as in the case of our running example, “Impressionist and Japanese art”. Melucci [103] studies the effectiveness of retrieval of overlapping passages against disjoint passages and shows that overlapping passages are superior with short query terms and for relatively small passage size of 50 to 100 words. The finding is supported by Kise et al. [81], who compares document retrieval based on document scoring against document retrieval using sliding windows of passages to find that for very small queries sliding windows works better.

The advantage of flexible unit of retrieval in document retrieval suggests that we should move away from predefined passage retrieval towards *arbitrary passage retrieval*, that allows variable passage length. Figure 2.2, shows an example of an arbitrary passage underlined. The passage starts from the fourth incidence of search terms, “Impressionists” and ends at the sixth incidence of search terms, “Japanese”, containing three incidences of search terms in total.

Few previous work on arbitrary passage retrieval exist. Kaszkiel and Zobel [76] approximate arbitrary passage retrieval by having a fixed-word-sized sliding windows of 12 different sizes to make computation feasible. Because some of their results skew towards shorter lengths, they use pivoted length normalization to normalize the passage lengths. The results on the performance of document retrieval are that there is no single ideal size of a passage and that not using document markup such as sentence boundaries provides a more effective arbitrary passage retrieval system. Liu and Croft [91] approximate arbitrary passages similarly to Kaszkiel and Zobel. They also support that the passage length can be of any length. Mittendorf and Schäuble [107], and Knaus et al. [83] show how arbitrary passage retrieval is possible using Hidden Markov Models . However, the lack of a passage retrieval evaluation set forces them to evaluate a document returned that contains the passages returned in a document collection. Clarke et al. [16] show an arbitrary passage retrieval algorithm that works well for one to three query terms, but does not consider term frequency. The score of a passage depends on the number and the lengths of the shortest sub-passage that starts and ends with any query terms.

The decision of ideal passage size to return becomes more complex in the case of arbitrary passage retrieval. Suppose that we define an arbitrary passage to start and end with an instance of query terms, as in Section3.2, Definition 8. Then, in Fig 2.2, we have 21 possible passages of size one to size seven. A one-term passage seems too short, but the 21-term passage may be too long. The previous work on predefined passage retrieval suggests that the ideal size of passage to return is not too long, and not too short. The ten-word sliding window of Fig 2.1-a may be too short, but the entire article may be too long, but the paragraph in Fig 2.1-b/c may be about right.

Zobel et al. [143] compare the effectiveness of retrieving documents based on scoring of entire documents, pages, and sections according to SGML markup. They find that scoring of pages is superior to document scoring, which in turn is superior to section scoring. They suggest this may be due to the large length of documents and small length of sections. Huang et al. [56] on the other hand, show that retrieval based on chapters is better than that of a document. The disparities between Zobel et al. [143] and Huang et al. [56] may be due to the fact that chapters are significantly longer than sections, suggesting that markup can be helpful if we only score those that are long enough. This idea is explored by Myaeing et al. [110], who show that judicious weighting of SGML elements improves performance for document retrieval over ignoring SGML elements. Although the above results examines the performance of passage retrieval in document retrieval systems, it also holds true for question answering systems that use passage retrieval. Abney et al. [1] compare the performance of 50 and 250 bytes passage retrieval approach with entity recognition in question answering against retrieval of single sentences using tf-idf. They find that their 250-byte passage retrieval approach works better than their single-sentence retrieval approach, which in turn is better than 50-byte passage retrieval. An interesting approach to decide on the length of a passage maybe to consider semantics. O'Connor [111] looks for connector words such as “and” to combine two sentences into a passage. Lee et al. [85] used lexical chains to connect passages.

2.2.2 Scoring Function

Different ways to score passages exist. One of the simplest ways to do so is to not score passages, but instead select all passages that contain all query terms, resulting in an AND query [111]. This type of query as well as other forms of boolean query is not popular in IR literature, because it misses many relevant documents. Instead, a ranked query is normally used [98]. Instead of retrieving precise matches to the query as in a boolean query, the search engines ranks text according to the relevance to the query. A simple way to use a ranked query to score passages is to count the number of query terms in the sentence. If one sentence has higher term frequencies than the other, we choose that sentence. A variant of this method combined with linguistic analysis seems fairly common among question answering tasks. Light et al. [86] call this approach *word overlap*. They show that the size of overlap relative to other sentences indicate the relevance to a question, even though the size may be small. A similar, but more complex method than simply counting term frequencies, is to use weighted term frequencies to score passages. The VSM uses an inverted document frequency of a term to determine term weights. Barbay et al. [3] adopt this in the passage retrieval of XML documents. Their scoring function is the sum of weighted term frequencies. Abney et al. [1] also score sentences using weighted term frequencies, but since their unit of passage was three-sentence long, they discount the score of the first and the last sentences against the score of the middle sentence. A slightly more advanced method of scoring passages than weighted term frequency, is a variation of density. Kise et al. [81] locate the densest part of a document according to tf-idf measure to aid in the retrieval of documents. For passage retrieval application, however, it lacks length normalization. Tellex et al. show that the density-*based* scoring work well [131] on pre-defined passage retrieval on the performance of question answering combined with document retrieval. BM25 develops these density-based functions, but add length-normalization.

BM25 [122] along with cosine measures [124] and language modeling [114] are widely used to score passages [1,8,91,118,126,143]. For example, Ittycheriah et al. combine a BM25-variant score of a sentence with a score based on linguistic analysis [70]. They look in WordNet hypernyms and hyponyms of a word that occurs after a question term. The additional sentence score then depends on the distance of such words and hyper/hyponyms in the WordNet. Pivoted length normalization with cosine measure performs well for passage retrieval [76].

The language modeling belongs to a group of approaches that use Bayesian techniques. Melucci presents a way to score passages using Bayesian probabilities [103]. He computes a posterior probability of term occurrences for a given passage, and judges the passage to be relevant if it is less than the mean. Language modeling can be thought of as a Hidden Markov Model. At a document level, a basic Hidden Markov Model approach proposed by Miller et al. [105] works well. He showed that a series of query terms can move between two states, a document, and a "general English", each with a transition probability. Then a probability of a document being relevant given a query becomes a linear combination of term frequencies over the document and over the entire corpus, assuming a uniform probability of query occurrence and document relevance. The linear combination then resembles the basic language model used for document or passage retrieval [54, 91, 114]. At a passage level, Mittendorf and Schäuble [83, 107] use a Hidden Markov Model to retrieve passages. Each term in a document is mapped to either the relevant or the non-relevant state. The transition probabilities and the background model for each state determines the next state. At each sate, a similarity value between the term and the query is produced as an observation. In order to find the best passage, we first compute the sequence of similarity values for a given document and a query. We then use Vitarbi algorithm to find the path that maximizes the probability of producing the sequence of similarity values. Makhoul et al. use Hidden Markov Model in audio stream retrieval [94]. They construct two models, one from the audio, and the other from the audio transcript to perform speech recognition. Myaeng et al. [110] use a Bayesian net to represent the hierarchical SGML structure, in which leaf nodes point to terms they contain. The degree of belief of an element containing a query term is then obtained by computing the conditional probabilities from the top down.

Outside of the variants of the standard tf-idf and probabilistic model-based scoring functions, there exist various schemes using hand-crafted matching rules and information theoretic scoring scheme. Hovy et al. [55] and Lee et al. [85] create hand crafted matching rules to score sentences. Hovy et al.'s score depends on how the query terms matched. For example, if the query term matched with proper capitalization, the score was changed. Lee et al.'s rules gave scores according to the distribution of matched terms, in addition to the weights and number of matched terms. Although all the above scoring functions treated query terms as a bag of words, there are some that take into account the order of query terms. When the query is in the form of a natural language statement or question, choosing a passage whose occurrences of the query terms are in similar order as in the query sentence reduces the amount of false positives. Specifically, Cui et al. [22] use a fuzzy matching technique to find passages in which the query terms have the same grammatical dependencies as in the query. Clarke et al. [15] use an information theoretic scoring function. They score a sentence by how rare the occurrence of a sentence is by self-information of a probability that the sentence contains all different query terms. They then look for an answer to a question by looking at the vicinity of the high scoring sentences.

Is one function better than the other? Fang and Zhai [30] suggested that scoring function is a composition of "primitive weighting function", "document growth function", and "query growth function". The primitive weighting function defines how a score is awarded when a single-term document contains a single instance of a query term. The document growth function and the query growth function defines how the primitive weighting function can be extended as the number of terms in the document and the query increases. They decomposed some of the common document scoring functions into these functions and showed that these functions do not quite fit the list of constraints that make a good scoring function. They then generated a new set of scoring functions using the three-function model that follows the constraints and experimentally showed that the new scoring functions give better performance and less vulnerable to parameter tuning.

2.2.3 Complexity

Another factor that makes arbitrary passage retrieval challenging is its computational overload. In Figure 2.2, using a predefined passage retrieval on paragraphs, we only need to execute a scoring function five times. In arbitrary passage retrieval, we need to score 21 times. In general, if there are n words in a document, in the brute force approach, we need to score $1+2+3+\ldots+(n-1)+n = n(n-1)/2 = O\left(n^2\right)$ passages. Since computing two overlapping passages seems redundant, is it possible to close the gap?

Computational complexity of information retrieval system is rarely studied and most studies that do focus on document retrieval. Moffat et al. [109] show an $O\left(n \log n\right)$ time algorithm to rank all n documents with efficient memory usage to approximate document lengths. Moffat and Zobel [108] give a fast document retrieval algorithm by skipping parts of inverted lists. Dong and Watters [26] show an algorithmic improvement in making similarity computation and document ranking. The aforementioned work of Barbay et al. [3] is the only work that I know that provides an algorithm (linear time) for passage retrieval.

Computational complexity of arbitrary "passage" retrieval is widely studied in the field of bioinformatics; finding segments of a DNA sequence satisfying a certain condition. Huang [62] show how to identify certain biological compositional criteria of DNA sequence. Ruzzo and Tompa [123] describe a linear time algorithm to find a set of disjoint segments with a unit length that maximizes the total sum of weights. Chen and Chao [9] show how to find the shortest and the longest segments of unit length with minimum weight constraint. Lin et al. [88] show how to find the densest path in a *tree* with length constraint. Goldwasser et al. [48] and Chung and

Lu [13] present a linear time algorithm to find the densest segment, where a unit segment has an arbitrary length. Chen et al. [11] describe a polynomial time algorithm to find a k disjoined segments that maximizes the sums of the densities. These bioinformatics algorithms may be applicable in deriving an efficient passage retrieval algorithm. However, it requires reducing the more complex IR scoring functions into simpler forms.

2.2.4 Element Retrieval

Element retrieval involves retrieval of a portions of structured documents such as XML and SGML documents (Figure 2.1-c). The biggest question is, "Does markup help at all in the retrieval of passages?"

Larsen et al. [84] show that compared to reading the whole documents, users prefer reading XML elements. So users do prefer reading parts of documents over the whole one, but also a simple fixed-length passage retrieval of Figure 2.1-b may do the job. Kamps and Koolen [74] show that passages that the users highlighted to be relevant matches XML structure. Giving credit to retrieval of XML elements over fixed-length passage retrieval.

One advantage of retrieving XML elements over a passage is that a user can give structural hints such as types of elements preferred. For example, a user may specify that a section containing a paragraph describing "Impressionism" is preferred. However, the usefulness of structural hints seems limited so far. Kamps et al. [75] suggest that majority of user's queries do not require structural hints, and even when the structural hints are required, it only acts as precision enhancement. Trotman and Lalmas [134] suggest that XML structural hints are not useful because users do not give proper structural hints. Specifically, the users may specify they want an article element by `//article[about(., Impressionists)]//sec[about(., Japanese)]`, where when assessing, they prefer sections and other elements.

Another advantage of retrieving XML elements is that we can ignore minor elements such as `<italicized>`. Pharo [113] supports the findings of Trotman and Lalmas that users prefer reasonably long elements such as sections, but too long elements such as articles were not popular. This implies we could simply ignore short elements. Kamps et al. [72] says not so. They showed that although most XML elements are very short, the length of relevant XML elements are spread evenly over every length. Therefore, instead of simply ignoring short elements, we should perform length normalization to give more bias to longer elements. Ramirez et al. [117] shows that although retrieving small elements themselves are not desirable, linking the relevance information of the small elements to related elements increases the effectiveness of retrieval.

Though XML is capable of annotating semantics of its elements, e.g., "¡north america¿Canada¡/north america¿", the problem arises when a foreign system has no knowledge of what each of the tag means. This may not be a problem within a group of people who use the same standard, e.g., a group of online retailors, or a group of libraries. But a third-party web developers who want to integrate the information about a book in the library network and in the retailer network would have difficulty mapping each group's lingo into each other. The purpose of the *Semantic Web* is to take down the barrier between different schemas to enable free movement of data available on the web. A *Resource Description Framework (RDF)* was created to facilitate this process [24]. An RDF entity consists of a triplet, subject, predicate, and an object. In the above example, the subject would be "Canada", predicate would be "belongs to", and object would be "North America". In order to define what subject, predicate, and object means, a *uniform resource identifier (URI)* must be used. For objects, a literal string is also allowed. An example application is Creative Common's [2] music licensing information.

[2]`http://creativecommons.org/`

Year	Corpus	Tasks	Query Types	Assessment	Evaluation
2002	500 MB IEEE	unspecified	CO, CAS	individual	spec. v.s. exhaust.
2003		unspecified	CO, SCAS, VCAS	individual	spec. v.s. exhaust.
2004		unspecified	CO, VCAS	individual	spec. v.s. exhaust.
2005		Focused	CO+S, CAS's	highlighted	spec. v.s. exhaust.
2006	6 GB Wikipedia [25]	Focused	CO+S	highlighted	specificity only
2007		Focused	CO+CAS	highlighted	number of characters
2008		Focused	CO+CAS	highlighted	number of characters
2009	50 GB Wikipedia [125]	Focused	CO+CAS	highlighted	number of characters

Table 2.1: Ad Hoc Track Over the Years

2.3 INEX Ad Hoc Track Focused Task

In Section 2.3, I summarize the official top runs submitted to INEX Ad-Hoc Track focused task. Though unofficial runs are reported in the proceedings [32, 34–38, 46, 47] the best scores reported by two runs only perform as good as the top runs [63, 99].

I divide INEX years into two eras. The first four years from 2002 to 2005 represent the exploratory era, using the small 500MB IEEE corpus with changing specifications. The small size and the simplicity of the document structure in the IEEE corpus reduced the overhead cost of participation and allowed the INEX community to grow. The next four years from 2006 to 2009 represent "maturity years", where the task specifications and evaluation procedures remained mostly the same. This allowed participants to compare their approaches across the years. To make comparisons of top runs across the years, for each era, I first explain the differences in task specifications, followed by the descriptions of the top runs for each year in the era. The specifications and rankings are obtained from INEX proceedings [33, 39, 45, 51, 73, 95–97] and on official sites [3].

2.3.1 INEX 2002 - 2005

Tasks

From 2002 to 2004, INEX held a single track and task. In 2005, it introduced the Focused task, Thorough task, and FetchBrowse task. The Focused task prohibited overlaps in the results, where the Thorough task permitted overlaps. The FetchBrowse task asked the systems to return relevant documents first and then relevant elements in these documents. Thus the unspecified task from 2002 to 2004 can be considered as a Thorough task. There are roughly two types of topics, content-only (CO) topics and content-and-structure (CAS) topics. The CO topics have topic titles that are series of query terms. The CAS topics have, in addition to query terms, structural conditions enforced by a structure-specification language, NEXI [135]. In 2003, the CAS topics were divided into strict CAS (SCAS) and vague CAS (VCAS), where the structural condition should be interpreted vaguely in VCAS. In 2005, a single set of topics are created for CO+S query, that is a combination of CO and CAS query. Thus it became easier to compare the effect of using structure for the same set of topics. Until 2004, the assessments involved assigning numeric values for specificity and exhaustivity for a pool of elements returned by the systems. In 2005, instead of assessing each element, users highlighted relevant passages first, and then gave exhaustivity values. The evaluation metrics I report are average precision from 2002 to 2004, and nxCG [78, 79] for 2005.

[3] For 2003-2007, `http://inex.is.informatik.uni-duisburg.de:` followed by a year (e.g., 2003). For 2008/2009: `http://www.inex.otago.ac.nz/`.

Metrics

The normalized extended cumulative gain (nxCG) metric [78, 79] is computed as follows. First, each element in the assessment pool is assigned a pair of exhaustivity (0, 1, or 2) and specificity (0 or 1) values. For Thorough task, the pool corresponds to the recall-base. For focused task, the ideal recall-base is constructed by the removal of overlap. Overlap is removed by prioritizing higher-path elements and elements with higher exhaustivity/specificity value. For example, if a section contains a paragraph whose exhaustivity value is higher than itself, then the paragraph is chosen. On the other hand, if both elements have the same exhaustivity values, the parent node, the section is chosen.

To compute nxCG, we create two vectors, xG from the result set, and xI from the (ideal) recall base. We next fill in the vectors with a gain, $xG[i]$ of ith element in the result set and $xI[i]$ for the ith element in the (ideal) recall base as follows. For the Thorough task, if an element in the result set is not found in the recall base, the gain is 0. Otherwise, a gain of an element in the result set is the gain of an element in the recall base, which is usually the product of exhaustivity and specificity values. For the Focused task, a gain of 0 is given to an element in the result set that overlaps with a higher ranked element in the result set. Otherwise the gain is calculated as in the Thorough task. Sometimes it is possible to be lenient on overlap by assigning discounted values to overlapping results. Moreover, for the Focused task the gain in the result set is normalized so that sum of the gain of an element's descendants does not exceed the gain of the element. Once we obtain xG and xI, we create cumulative gain vectors, xCG and xCI by entering the cumulative gain values up to each rank. We finally obtain normalized xCG, or nxCG by dividing each entry of xCG with the corresponding entry of xCI.

Another metric used in this thesis is mean average effort-precision (MAep) and MAep at rank i denoted as iMAep. They are obtained by computing effort-precisions and gain-recalls as follows. An effort precision at a gain-recall point r is computed by the rank of a submitted run that reaches the given gain r, divided by the corresponding value from the (ideal) recall-base. Gain-recall at rank i is computed by submitted run's $xCG[i]$ divided by the $xCI[n]$ of an (ideal) recall-base. Then MAep is obtained by for each query, computing effort precisions at every rank, taking the average of the effort precisions at all rank, and taking the average of average effort precision across the queries. The value of iMAep is obtained by plotting an effort-precision-gain-recall curve.

Best Runs

With these variations in specifications across the years, I report evaluation results from 2002 to 2005 that use specifications similar to the years 2007 to 2009, when I participated, while maintaining consistency. Thus the rankings I base on are CO and CAS results, unless the combined CO+S rankings are available. For CAS results, VCAS results are reported over (S)CAS results whenever possible. Generalized quantization is used because it was used throughout the years. For 2005, focused results disallowing overlapping of the results are reported, otherwise the results permit overlaps.

2002 The general theme behind the top runs in 2002 is to compute a variant of tf-idf score for each element and then adjust the scores according to the structural requirement. However, CSIRO's unofficial run that only scores article surpassed their best-performing run that ranked first in CAS topics.

In CO topics, Universität Dortmund/Universität Duisburg-Essen tops the first two ranks with the highest score of 0.07 for the run `Epros03`. Their idea is to compute tf-idf for each node in the XML tree. The higher-level nodes receive discounted term frequencies for the terms contained in the descendant nodes [50]. The second best ranked group is Royal School of Library and Information Science with the score of 0.06 for the run `bag-of-words` [51] [4].

[4]Their approach is unknown as they did not submit a paper to the proceeding.

For CAS topics, the run `manual` by CSIRO Mathematical and the run `Merge` from Information Sciences and IBM Haifa labs occupies the first two ranks with the score of 0.28 and 0.27 respectively. The CSIRO run scores selected elements by the combination of BM25 and the term frequencies. It is then filtered by the manually constructed structural requirements, where a term in an element only maps to the smallest container element. This run, however, is surpassed by their unofficial run that simply retrieves documents [136]. IBM Heifa translates INEX queries into their own query forms that define structure and terms. They score elements by how much the structure defined in the query matches the structure of a given element. It is combined with a term similarity score in a variant of vector space model that takes structural resemblance in mind [102].

2003 The general theme behind the top runs in 2003 seems to be combining the element scores with document scores.

For CO topics, University of Amsterdam submitted the best performing run, `UAmsI03-CO-lamda=0.20` with score of 0.10. The group submitted two other runs that rank the second and the third. An element score is computed using a language model and depends on the text inside the element as well as the text surrounding the element. All the top three runs used different smoothing factors to bias the element size to very large to very small. The best performing run has a bias towards middle-sized elements such as sections [126]. IBM Haifa submitted the next best performing run, `CO-TDK-With-No-Clustering` with score of 0.10. They apply VSM at component level, scoring each element with component-wide term statistics [100].

For SCAS topics, University of Amsterdam occupies the first three ranks with the run `UAmsI03-SCAS-MixedScore` giving the score of 0.30. It extracts `<article>` elements from an element retrieval run, from these, extracts all elements that satisfy structural constraints. The final element score is the combination of the element score and the document and other related elements' scores [126]. The best runs are followed by the run `SCAS-TDK-With-No-Clustering` from IBM Haifa with the score of 0.24. They first score documents according to the query term specified in the article component of the topic. If components smaller than an article were also specified in the topic, each of the components (as well as components similar to it) is queried accordingly.

2004 [5] In 2004, various approaches produced top ranking runs. The Amsterdam's technique of combining the element and the document scores from 2003 still ranks high, and ranks first in VCAS task, where they ignored the structural requirements.

For CO topics, IBM Haifa tops the first two runs with the score of 0.13 for the best run, `CO-0.5-LAREFIENMENT`. When scoring individual elements they use pivoted length normalization at element level to normalize the differences in element-wide statistics such as average element lengths. The first set of results are then used for query expansion [101]. The third ranked run `UAms-CO-T-FBack` was submitted by University of Amsterdam achieving the score of 0.12[6]. They use a language modeling approach that combines the score of elements and their surroundings, along with blind feedback. [127]. For VCAS task, University of Amsterdam submitted the top performing run, `UAms-CAS-T-FBack` with the score of 0.12, followed by the run `VCAS_PS_stop50K_099_049` from Queensland University of Technology (QUT) with the score of 0.12[7]. The Amsterdam run is implemented exactly the same as their best performing CO run, only using VCAS topics without structural constraints [127]. The QUT runs create a tree consisting of elements that contain at least one query term, assign a heuristic tf-idf-esque score to each leaf node and propagate the scores upwards the tree using heuristic functions. The structural constraints are then applied to filter out the results [43].

[5]Two assessments sets exist to accommodate duplicate assessors for a number of topics. Here, the first Assessments set is used.

[6]`http://inex.is.informatik.uni-duisburg.de:2004/internal/results/CO.html`

[7]`http://inex.is.informatik.uni-duisburg.de:2004/internal/results/VCAS.html`

2005 In 2005, both of the top runs seem to use score propagation along the XML document tree. It can be thought of a variant of combining a document (background) and element scores.

In the CO+S task, University of Kaiserslautern tops the first two ranks with the best scoring run `COS_Pattern_Focussed` having nxCG[10]=0.29. They guess the element types by such patterns as the length of its parent and child node. Once the element types are determined, terms that appear in the elements have an effect on the scores of its neighboring nodes [27]. The best run is followed by the run `COS_3_Focused_highest_VVCAS` from Queensland University of Technology with the score of nxCG[10]=0.28. The QUT run performs an overlap elimination on the last year's task. Instead of filtering a ranked list of overlapping elements top down, they create a set of non-overlapping elements first by taking the highest scoring elements in each path, and then sorted the elements according to the scores [44].

2.3.2 INEX 2006 - 2009

Tasks

From 2006 to 2009, Ad Hoc Track always held a Focused task, in addition to a Best-in-Context task, Relevant-in-Context task, and the occasional Thorough task. The CO and CAS topics are unified [8] that enable ranking of runs using either run together. The assessments are conducted by highlighting relevant passages and except for 2006, the evaluation metric is a precision and recall respect to the highlighted passages. For 2006, later result gives a ranking using 2007 metrics which I report here. Thus I report two runs from the first two best performing groups in Focused

[8]Confusingly, in 2006, the unified topics are called CO+S topics. From 2007, the unified CO+S topics are called CO topics and CAS topics, whereas prior to 2007, a separate CO(+S) and CAS topics exist.

task, CO or CAS topics, and with the official metric of interpolated precision at a recall of 0.01 (iP[0.01]) with respect to highlighted passages.

Metrics

The interpolated precision is computed by first drawing a recall-precision curve, and then at each specific recall point, say at 0.01, choosing the highest precision obtained up to the chosen recall point. Another metric used in this thesis, MAP roughly corresponds to the area under the interpolated recall-precision curve. To obtain the specific MAP value, for each query we compute an average precision. At each rank we compute a precision of a set consisting of results up to the rank. We accumulate these precisions at all ranks and then divide this by the number of results returned to obtain average precision. In order to obtain the mean value of average precision, we average the average precisions across all queries.

Top Runs

2006 In 2006, the best run continues to use the approach of combining the background score with an element score.

The best run was submitted by LIP6, `OkTg-Lineaire-RkSym-100cent...-NoTrait` [9] with iP[0.01]=0.45. It first uses BM25 to score elements, and then uses machine learning to learn how to combine the scores of the element and its parents [137]. The second best run `FOCU-BM25-cutoff400` was submitted by City University London with iP[0.01]=0.44. They limit the element lengths to 400 characters to score them by BM25. Though they also cut off the number of documents from which elements were to be retrieved, they give no improvement [93].

[9] The actual run name is: OkTg-Lineaire-RkSym-100cent-Oki-2-7-0.75-DocDoxParent-VectTagFamClass-2006etiq-50it-2006-SansOvl-NoTrait

2007 In 2007, though the top run continues to use the mixture of element and the background score, my run that only used the element score ranks behind the top two runs.

Dalian University of Technology submitted the best performing run with iP[0.01]=0.53. They combine document scores with element scores. They check CAS titles to see if it contains an image path, separated the topics with image path to process differently. [89]. Ecoles des Mines de Saint-Etienne ranks the second with iP[0.01]=0.52. They create a proximity function that computes the proximity of each term in the document to each query terms. The proximity scores for each term is summed for each section-like elements. The structure is taken into account by giving maximum proximity scores to those terms appearing in the title-like elements [4]. My run ranks the third with iP[0.01]=0.51 [33]. My run scores selected XML element types by BM25.

2008 In 2008, top runs gave higher weights to the terms appearing at a certain location in the elements.

My run `FOERStep` ranks the best with iP[0.01]=0.69. It modifies the basic focused retrieval approach from 2007 by adding extra weights to the terms appearing at the beginning of elements. It gives a very modest improvement over the baseline run that ranks the second. My runs are followed by a run `TopX-CO-Proximity-articleOnly` from Max-Planck-Institut für Informatik with iP[0.01]=0.68. The run only retrieves article-level elements combining the BM25 scores with proximity scores. Proximity scores are introduced to give more weights to terms appearing together in the same element over terms that appear across an element boundary. Adding proximity scores improves over the baseline BM25-only article run, but decreases the performance at official metric of iP[0.01] when applied for element retrieval [7].

2009 In 2009, the best run combined the background and the element scores as often done by the previous years' best perfoming runs. However, the second ranked run only scores documents.

My run `UWatFERBM25F` ranks the best with iP[0.01]=0.63. It builds on the baseline run from

2007 by combining the element scores with the scores of all titles the element belongs to. The second best run `I09LIP6Okapi` from LIP6 had iP[0.01]=0.61 [23]. It is a baseline run retrieving only documents.

2.3.3 Meta-analysis

Table 2.2 shows that a good XML retrieval system can be made by scoring either single elements or articles by a variant of VSM, BM25, and language modeling. Arbitrary passage runs were submitted by a few participants since 2007, but none topped the top 2 nor dominated the top 10. A list of XML element types to score is extracted by observing previous years' assessments. Structural information may be used to propagate an element score across the XML tree, or to filter out irrelevant element-types. Document level statistics can be combined with the element statistics to produce the relevance score. However, considering LIP6's second ranked run in 2009 that only scores articles with BM25, the big question at INEX, the utility of XML structure in the retrieval of text fragment is still left unanswered.

2.4 State of the Art for Focused Search Engines

In Section 2.4, I present the current state [10] of focused, heterogeneous search on major search engines, Google [11], Yahoo [12], Bing [13], and A9.com [14]. Google has the largest search engine share in the US of 64.4% as of April 2010, followed by Yahoo!, and Microsoft search sites including Bing [18]. Although Amazon's search engine, A9.com may not have a large share, it offers both book search and heterogeneous search.

[10] As of June 10, 2010 and July 7, 2010

[11] `http://www.google.com/`

[12] `http://www.yahoo.com/`

[13] `http://www.bing.com/`

[14] `http://a9.com/`

Year	Group	Scoring	Structure	Size	Note
2002	U Dortmund	VSM	no	single elements	
2002	RSLIS	-	-	-	no final report
2002	CSIRO	tf, BM25	no	article-only	unofficial run
2002	IBM Haifa	extended VSM	yes	single elements	CAS
2003	U Amsterdam	language model	no	single elements	
2003	IBM Haifa	VSM	no	single elements	
2003	U Amsterdam	language model	yes	single elements	SCAS
2003	IBM Haifa	VSM	yes	single elements	SCAS
2004	IBM Haifa	SVM	no	single elements	
2004	U Amsterdam	language model	no	single elements	
2004	U Amsterdam	language model	no	single elements	VCAS
2004	QUT	VSM	yes	single elements	VCAS
2005	U. Kaiserslautern	Lucene	yes	single elements	Lucene uses VSM
2005	QUT	VSM	yes	single elements	
2006	LIP6	BM25	yes	single elements	learning to rank
2006	City U. London	BM25	no	single elements	
2007	Dalian	unknown	yes	single elements	CAS
2007	Ecole Mines de S.-E.	proximity	yes	single elements	self-made function
2007	U. Waterloo	BM25	no	single elements	
2008	U. Waterloo	biased BM25	no	single elements	
2008	MPI	BM25	yes	article-only	
2009	U. Waterloo	BM25F	yes	single elements	
2009	LIP6	BM25	no	article	

Table 2.2: Best Performing Focused Retrieval Strategies

2.4.1 Google

Google has the most comprehensive search features regarding focused snippets and heterogeneous search results of all the search engines studied in Section 2.4.

Figure 2.3 [15] shows an example search using “Impressionist and Japanese art”. We see that the snippets returned resemble what a focused search should return. In addition, for certain results, it gives links to jump into the beginning of sections inside the result pages. The drawback, however, is that since Google does not perform pure-focused search, when clicking on the page results, it brings directly to the beginning of the page, as opposed to where the snippet starts.

Google performs heterogeneous search, or “Universal Search” as they call it [49], over web pages, books, images, video, news and others. The bottom result in Fig 2.3 shows an example of a book search result blended in with web search results. When clicking the book result, it takes into the pages where the search phrases appear. Thus for book search, Google implements focused heterogeneous search.

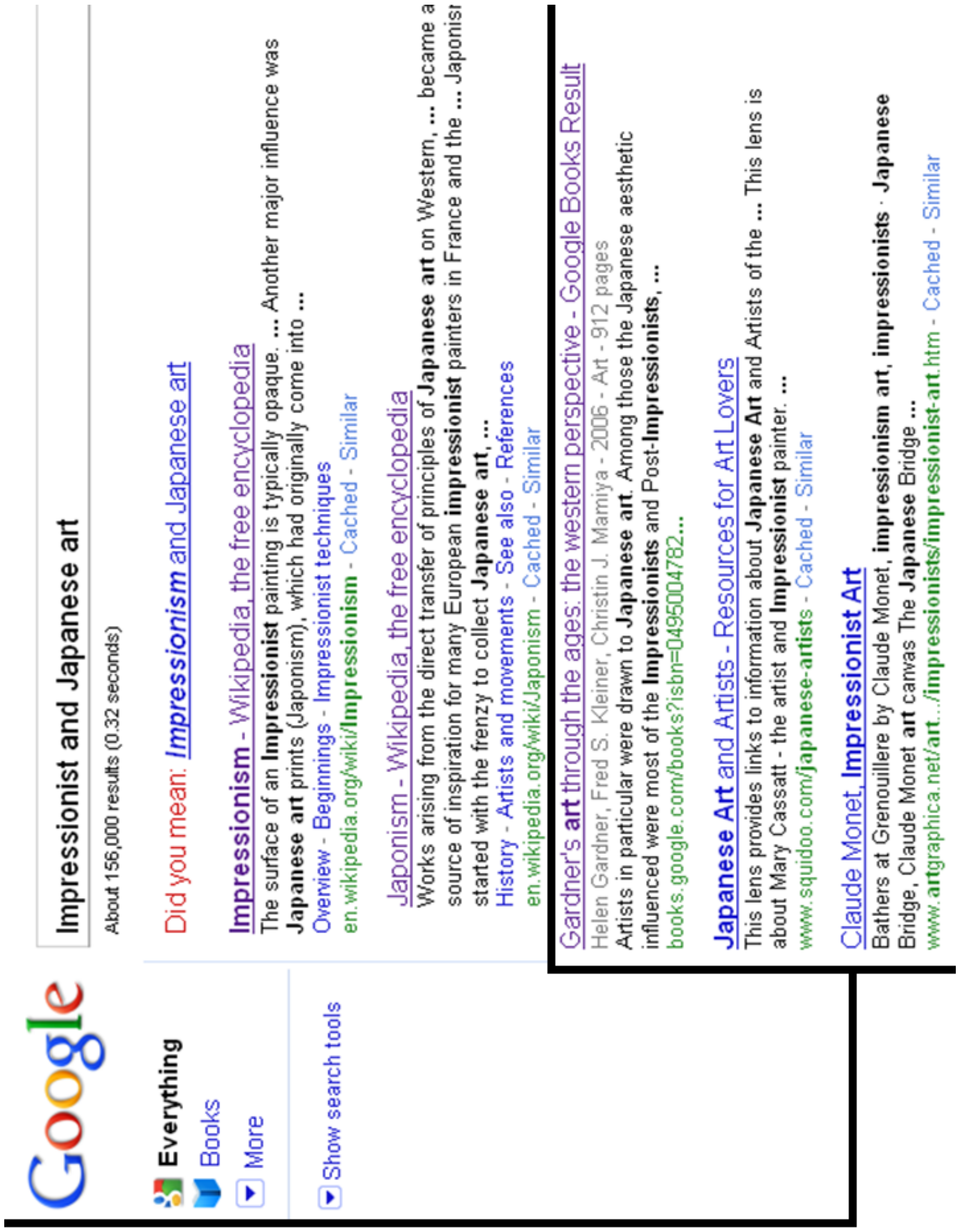
2.4.2 Yahoo! and Bing

Fig 2.4 [15] shows the search results using Yahoo! and Bing for a query “Impressionist and Japanese art”. At first glance, the search results of Yahoo! and Bing seem similar to that of Google. All have snippets and links to jump into the beginning of the sections in a search result page. However, neither of them returns heterogeneous results. For example, they lack book search results, beyond searching Amazon.com site.

The more concrete query phrase, “Katsushika Hokusai”, however, changes the landscape. Yahoo! now includes images tagged as such as their search results similar to Google results [16]. Bing, in addition to including image results, includes an *entity card*, a summary of the entity searched,

[15]Snapshots taken on June 10, 2010.

[16]As of July 7, 2010



Impressionist and Japanese art

About 156,000 results (0.32 seconds)

Did you mean: *Impressionism* and Japanese art

Impressionism - Wikipedia, the free encyclopedia
The surface of an **Impressionist** painting is typically opaque. **...** Another major influence was **Japanese art** prints (Japonism), which had originally come into **...**
Overview - Beginnings - Impressionist techniques
en.wikipedia.org/wiki/**Impressionism** - Cached - Similar

Japonism - Wikipedia, the free encyclopedia
Works arising from the direct transfer of principles of **Japanese art** on Western, **...** became a source of inspiration for many European **impressionist** painters in France and the **...** Japonisr started with the frenzy to collect **Japanese art**, **...**
History - Artists and movements - See also - References
en.wikipedia.org/wiki/Japonism - Cached - Similar

Gardner's **art** through the ages: the western perspective - Google Books Result
Helen Gardner, Fred S. Kleiner, Christin J. Mamiya - 2006 - Art - 912 pages
Artists in particular were drawn to **Japanese art**. Among those the Japanese aesthetic influenced were most of the **Impressionists** and Post-**Impressionists**, **...**
books.google.com/books?isbn=0495004782...

Japanese Art and Artists - Resources for Art Lovers
This lens provides links to information about **Japanese Art** and Artists of the **...** This lens is about Mary Cassatt - the artist and **Impressionist** painter. **...**
www.squidoo.com/**japanese-artists** - Cached - Similar

Claude Monet, **Impressionist Art**
Bathers at Grenouillere by Claude Monet, **impressionism art**, **impressionists** · **Japanese** Bridge, Claude Monet **art** canvas The **Japanese** Bridge **...**
www.**art**graphica.net/**art**.../**impressionists/impressionist-art**.htm - Cached - Similar

Figure 2.3: Example Google Search Result

extracted from multiple search results. Figure 2.5 [17] shows that for the Japanese printmaker, it returns his vital statistics as his date of birth. For a non-living entity, “Impressionism”, it returns the focused result that summarizes the entity. Google, on the other hand, instead of returning a summary of any entity, creates a timeline of a historical entity from the search results.

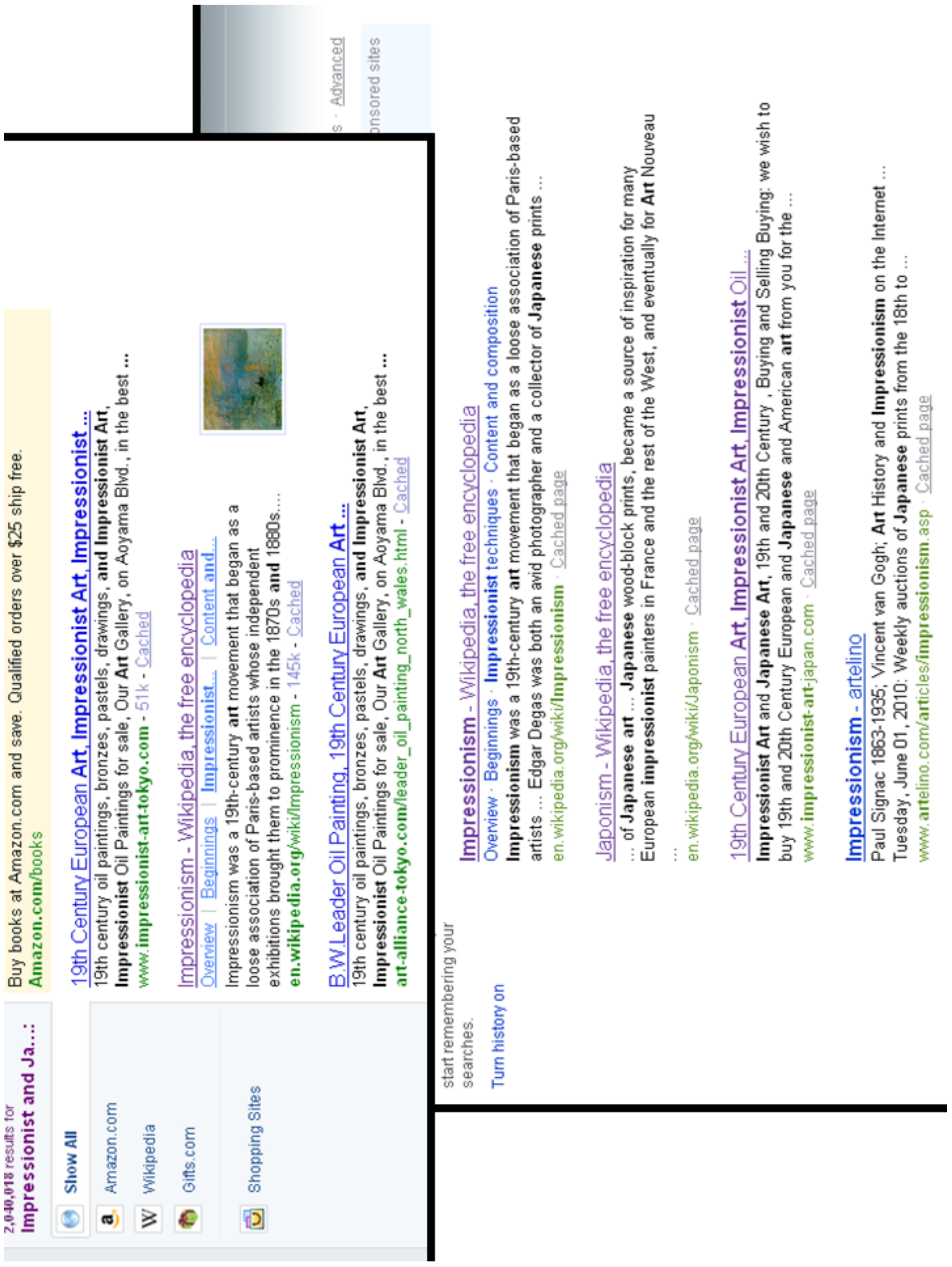
2.4.3 A9.com

An online retailer Amazon.com subsidizes a search engine company, A9.com. Technologies developed at A9.com are used, among others for searching books and products on Amazon.com.

Given a query term, the book search engine matches the query terms against the book titles, authors, and the text. For those books that copyrights allow Amazon.com to display contents of the books, the search results show snippets and page numbers in which the query terms appear. Once users select a book, users can also search inside the books. Although it appears that the book search engine performs focused search, the page search results, listing the results in ascending page numbers, seem to be a simple matching of query terms, rather than a ranked list based on relevance. The snippets results accordingly, seem to list the first incidence of a query term match in the book, rather than a focused result.

The product search lets user search for any products sold at Amazon.com. For example, with the query “Impressionist and Japanese art”, the first result is a wall screen of a Japanese art, the second result is a book, the third result a DVD, and the lower ranked results are occupied by poster prints.

[17]Snapshots taken on July 7, 2010.



Buy books at Amazon.com and save. Qualified orders over $25 ship free.
Amazon.com/books

19th Century European **Art, Impressionist Art, Impressionist ...**
19th century oil paintings, bronzes, pastels, drawings, **and Impressionist Art, Impressionist** Oil Paintings for sale, Our **Art** Gallery, on Aoyama Blvd., in the best **...**
www.**impressionist-art-tokyo.com** - 51k - Cached

Impressionism - Wikipedia, the free encyclopedia
Overview | Beginnings | **Impressionist...** | Content and...
Impressionism was a 19th-century **art** movement that began as a loose association of Paris-based artists whose independent exhibitions brought them to prominence in the 1870s **and** 1880s....
en.wikipedia.org/wiki/Impressionism - 145k - Cached

B.W.Leader Oil Painting, 19th Century European **Art ...**
19th century oil paintings, bronzes, pastels, drawings, **and Impressionist Art, Impressionist** Oil Paintings for sale, Our **Art** Gallery, on Aoyama Blvd., in the best **...**
art-alliance-tokyo.com/leader_oil_painting_north_wales.html - Cached

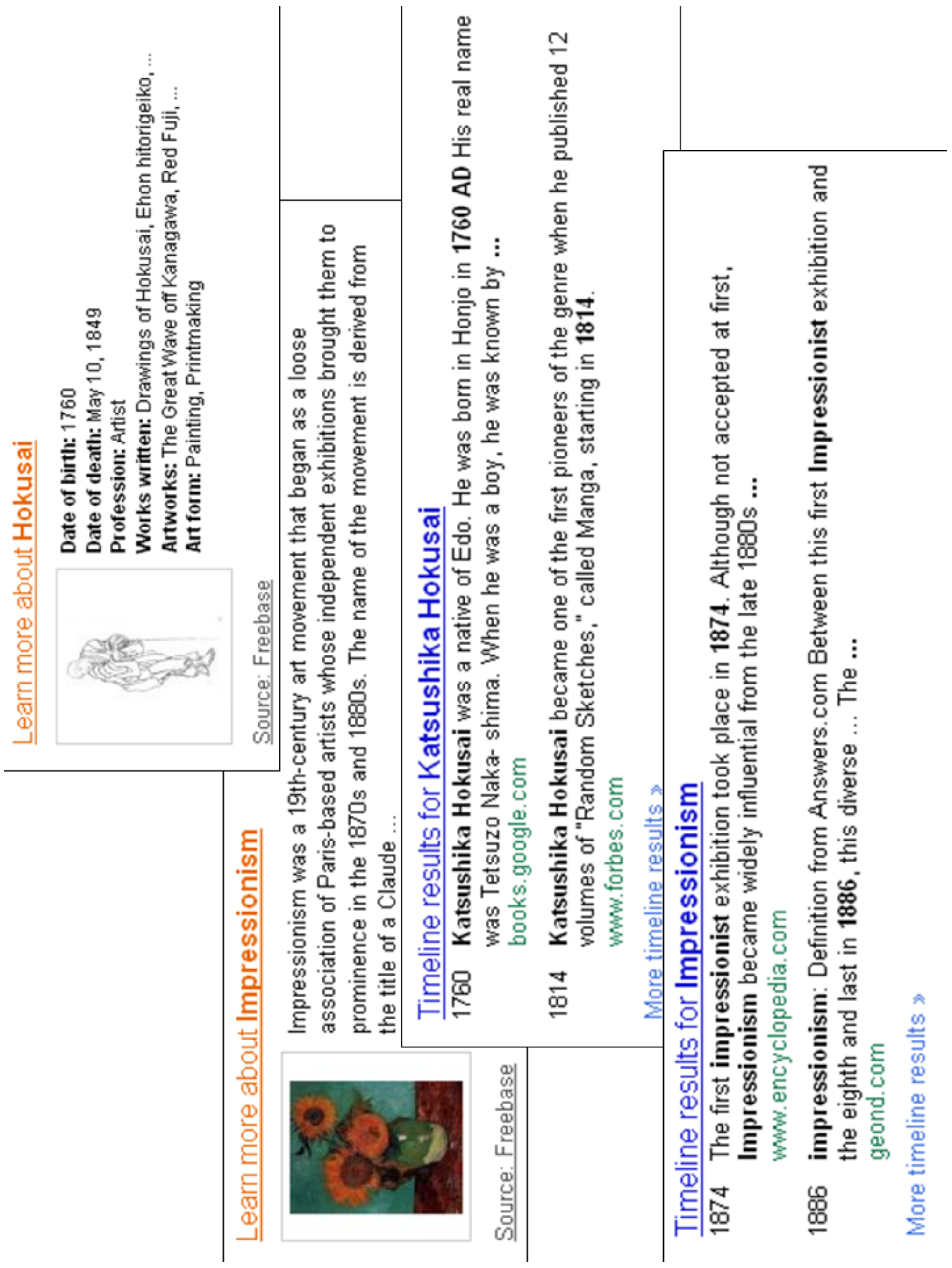
Impressionism - Wikipedia, the free encyclopedia
Overview · Beginnings · **Impressionist** techniques · Content and composition
Impressionism was a 19th-century **art** movement that began as a loose association of Paris-based artists ... Edgar Degas was both an avid photographer and a collector of **Japanese** prints ...
en.wikipedia.org/wiki/**Impressionism** · Cached page

Japonism - Wikipedia, the free encyclopedia
... of **Japanese art** ... **Japanese** wood-block prints, became a source of inspiration for many European **impressionist** painters in France and the rest of the West, and eventually for **Art** Nouveau ...
en.wikipedia.org/wiki/Japonism · Cached page

19th Century European **Art, Impressionist Art, Impressionist** Oil ...
Impressionist Art and **Japanese Art**, 19th and 20th Century , Buying and Selling Buying: we wish to buy 19th and 20th Century European and **Japanese** and American **art** from you for the ...
www.**impressionist-art**-japan.com · Cached page

Impressionism - artelino
Paul Signac 1863-1935; Vincent van Gogh; **Art** History and **Impressionism** on the Internet ...
Tuesday, June 01, 2010: Weekly auctions of **Japanese** prints from the 18th to ...
www.**art**elino.com/**art**icles/**impressionism**.asp · Cached page

Figure 2.4: Example Yahoo! (top) and Bing (bottom) Search Results



Learn more about **Hokusai**

Date of birth: 1760
Date of death: May 10, 1849
Profession: Artist
Works written: Drawings of Hokusai, Ehon hitorigeiko, ...
Artworks: The Great Wave off Kanagawa, Red Fuji, ...
Art form: Painting, Printmaking

Source: Freebase

Learn more about **Impressionism**

Impressionism was a 19th-century art movement that began as a loose association of Paris-based artists whose independent exhibitions brought them to prominence in the 1870s and 1880s. The name of the movement is derived from the title of a Claude ...

Source: Freebase

Timeline results for **Katsushika Hokusai**

1760 **Katsushika Hokusai** was a native of Edo. He was born in Honjo in **1760 AD** His real name was Tetsuzo Naka- shima. When he was a boy, he was known by **...**
books.google.com

1814 **Katsushika Hokusai** became one of the first pioneers of the genre when he published 12 volumes of "Random Sketches," called Manga, starting in **1814**.
www.forbes.com

More timeline results »

Timeline results for **Impressionism**

1874 The first **impressionist** exhibition took place in **1874**. Although not accepted at first, **Impressionism** became widely influential from the late 1880s **...**
www.encyclopedia.com

1886 **impressionism**: Definition from Answers.com Between this first **Impressionist** exhibition and the eighth and last in **1886**, this diverse ... The **...**
geond.com

More timeline results »

Figure 2.5: Entity Cards from Bing (top) and Timelines from Google (bottom)

2.5 Conclusions

In Chapter 2, I list previous work related to focused or heterogeneous search. For focused search, even though there is a good evidence that the finer the granularity of search, the better the results, nobody has come up with a way to efficiently implement an arbitrary passage retrieval approach. I next describe efforts at INEX. Since 2002, it has been unclear whether there is a way to take advantage of the XML structure to improve retrieval performance significantly. Passage retrieval was implemented by some participants, but although they perform comparable to other approaches, they did not occupy the highest ranks among the submitted runs. Finally, I survey the current state of the popular web search engines. Google performs reasonably well in regards to focused heterogeneous search, though the algorithms of theirs and other search engines are unknown. Amazon.com's product search presents heterogeneous search well, whereas the book search, even though they take into account the content of the books in addition to the titles and the authors when ranking books, when searching inside a book, the search results seem a simple matching of query terms as opposed to focused results. Therefore, there is a need to investigate the following three agenda.

- **Efficiency:** Is there a way to efficiently implement focused retrieval ?
- **Size:** What is the best granularity of search results, sentence, paragraphs or XML elements?
- **Heterogenity:** How do we score objects of different type?

In the next three chapters, I investigate these three issues individually.

Chapter 3

Foundation of Passage Retrieval

This chapter defines the problem of passage retrieval. It also tries to reduce the running time of the brute-force passage retrieval approach quadratically, to present a solution to the first problem of focused retrieval, complexity, proposed in Chapter 1. The next Chapter 4 then presents how the problem of passage retrieval defined in this Chapter can be implemented. The later Chapters 5 and 6 use the implementation to solve the problem of passage retrieval, in order to answer two other questions of focused search, the unit of retrieval and taking advantage of XML structure. Additionally, Chapter 3 reduces one-term BM25 and language scores in the focused retrieval setting to a proposed function I used to analyze the complexity.

Section 3.1 introduces to the problem of passage retrieval. Section 3.2 defines a formal model of passage retrieval. Section 3.4, reduces two popular document scoring functions into one canonical form. Section 3.4, discusses the consequence of the canonical form. Section 3.5 concludes Chapter 3 with future possibilities.

3.1 Introduction

Suppose there is a sequence of n points on a line with positive values W_i associated with each of them. Moreover, each pair of points W_i and W_{i+1}, called an *extent* denoted $[i, i+1]$ has a length l_i. Given a search phrase and the relevance scoring function $S[u, v]$, our goal is to find a set of extents $[u, v]$ in decreasing order of scores such that the extents do not overlap with each other, that contains all n points.

In the domain of passage retrieval, the points map to occurrences of search terms such as "Impressionist" and "Japanese", l_i maps to the number of terms between the points i and $i+1$, and $S[u, v]$ maps to a scoring function such as BM25 that measures the relevance to the query. Though my focus in this thesis is focused text retrieval, I believe that the problem can be generalizable in other areas of research such as audio retrieval.

The motivation comes from INEX Ad Hoc track Focused task, where overlaps are disallowed in the result set. One of the ways to remove overlaps, that I employ throughout the thesis and described in Chapter 4, is to score all XML elements, sort them by scores, and then examine each elements top-down to see if there is any overlap in the result set. If there is no overlap in the result set, the element in question joins the result set. For example, if a section contains a paragraph whose score is higher than itself, then I choose the paragraph over the section. The problem with this approach is that it involves scoring overlapped parts multiple times. In the case of my example, it involves scoring the section once and the paragraph once, even though the text that contains the search terms in the paragraph is scored twice, once as part of the section, and once as part of the paragraphs. It seems that if there is a way to examine each part of text only once, it would reduce the processing time quadratically.

3.2 Formal Model of Passage Retrieval

Section 3.2 constructs a mathematical model of passage retrieval. Because my goal is to make passage retrieval general, so that it can be applied to not only text but also audio streams and other data, I first define a set of symbols that is a building block of information entity, an entity we wish to retrieve information from.

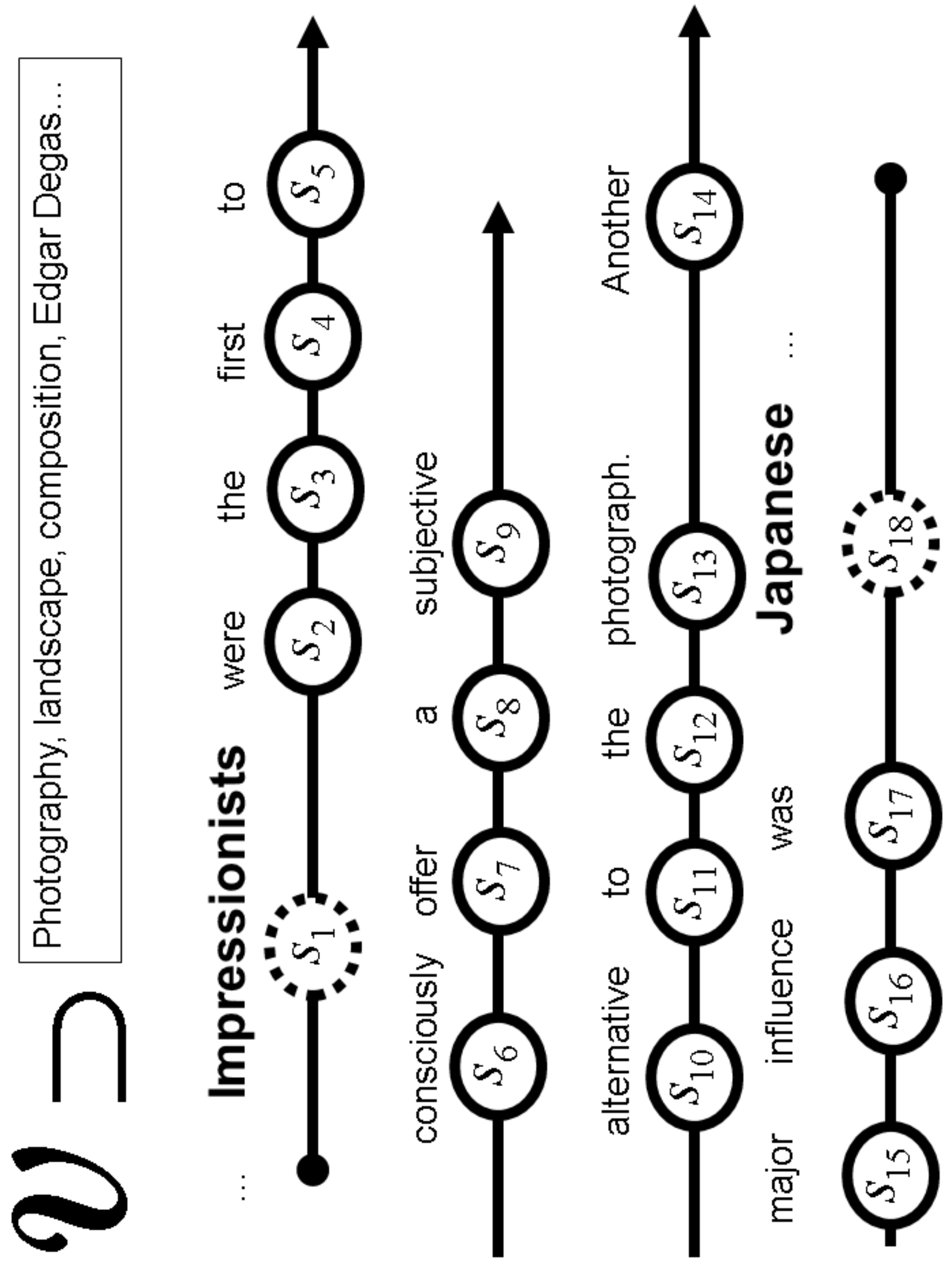
Definition 1 (Vocabulary) *A set of symbols $\mathcal{V}$ is called the* vocabulary.

A subset of vocabulary is used to represent an information entity. Figure 3.1 lists a subset of vocabulary used in the Wikipedia entry on Impressionism [1]. In general, in an English text, the vocabulary is English words, and in a French text, the vocabulary is French words. The information entity can also be XML files that contain data, or an audio stream. In an audio stream, a vocabulary could be a set of phonemes, but it could also be a set of words or both, depending on how we choose to represent it. It is also possible to define vocabulary as a set of letters that represent DNA sequences or a set of numbers or discrete signals that constitute a sequence of data stream. Next, I define a collection of entity we wish to retrieve information from in terms of symbols.

Definition 2 (Corpus) *A* corpus *$\mathcal{C}$ is a sequence of symbols $s_1, s_2, \ldots, s_N$, where $s_i \in \mathcal{V}$, for $1 \leq i \leq N$.*

A corpus can be a single book or a single section of an audio stream, or a collection, be it homogeneous or heterogeneous. In the case of collections, it is possible to ignore the collection boundaries such as book chapters and audio sections when retrieving. In my thesis, I set the collection boundaries as each book or article. The Wikipedia article in Figure 1.1, can be considered a corpus of length $L = 4441$ by MS Word word count.

[1]August 14, 2010 version of `http://en.wikipedia.org/wiki/Impressionism`



$v \supset$
Photography, landscape, composition, Edgar Degas...
... Impressionists were the first to
consciously offer a subjective
alternative to the photograph. Another
major influence was Japanese ...
s_1 s_2 s_3 s_4 s_5
s_6 s_7 s_8 s_9
s_{10} s_{11} s_{12} s_{13} s_{14}
s_{15} s_{16} s_{17} s_{18}

Figure 3.1: Vocabulary, Symbols, and Extents

I define a passage of a corpus $\mathcal{C}$ as follows:

Definition 3 (Extent) *An* extent *is a pair* $[u, v]$ *with length* l *representing an interval of corpus from* u *to* v*,* $s_u, s_{u+1}, \ldots, s_v$*.*

A passage in Figure 3.1 excerpted from Figure 1.1 [2] contains an extent $[s_1, s_{18}]$ spans the subsentence beginning with "Impressionists" and ending with "Japanese".

Given a corpus, a user can now issue a query.

Definition 4 (Query) *A* query *is a vector of symbols* $\vec{q} = \langle t_1, t_2, \ldots, t_m \rangle$ *where* $t_i \in \mathcal{V}$ *for* $1 \leq i \leq m$*.*

Therefore, I consider a query to be a disjunction of an unordered set of symbols, instead of a natural language query. For example, a Wikipedia query "Impressionist and Japanese art", will be represented as, after stop words removal,

$$\vec{q} = \langle \text{``impressionist''}, \text{``japanese''}, \text{``art''} \rangle .$$

Under this model, it is possible to add a scaler weight on each symbol in a query, which scales *match* function involving the symbol accordingly. For example, in the case of "**+impressionism** and **Japanese** art", we could represent the query vector as,

$$\vec{q} = \langle \mathbf{100\,\text{``impressionist''}}, \mathbf{100\,\text{``japanese''}}, \cdot\,\text{``art''} \rangle .$$

The score of *match* function will be scaled 100 times, which will be reflected in higher term frequency and score as we define below. In the rest of Chapter 3, I assume that $\vec{q}$ is arbitrary but fixed.

Given a query $\vec{q}$, I would like to match each $t \in \vec{q}$ against the symbols in $\mathcal{C}$. I define what makes two symbols match with each other. The following function represents the symbol similarity.

[2] `http://en.wikipedia.org/wiki/Impressionism`

Definition 5 (Symbol Match) *Given two symbols* $w_1, w_2 \in \mathcal{V}$, *define a function* match *so that* $0 \leq match(w_1, w_2)$. *In a simplest case,* $match(w_1, w_2)$ *is either* 0 *or* 1. *For* $\bigvee w \in \mathcal{V}$, $match(w, w) = 1$.

For example, if I set *match* to be a function of edit distance,

$$\text{match}\left(\text{“japanese”}, \text{“japanese”}\right) = 1,$$

$$\text{match}\left(\text{“japanese”}, \text{“japan”}\right) = 0.99,$$

and

$$\text{match}\left(\text{“japanese”}, \text{“camera”}\right) = 0.01.$$

But I can also set *match* to be a synonymity function, which returns 1 if the symbols are synonyms and 0 otherwise. In this case, the first two examples would return 1, and the second 0.

Match values are used in computing term frequencies.

Definition 6 (Term Frequency) *A* term frequency *of a single* $q \in \mathcal{V}$ *in interval* $[u, v]$ *is* $f_q[u, v] = \sum_{i=u}^{v} match(q, t_i)$. *From here, define a* term frequency vector $\vec{f} = \langle f_1, f_2, \ldots, f_m \rangle$ *of dimension* m *of a query vector* $\vec{q}$ *in the interval as* $f_i = f_{q_i}[u, v]$, *for* $0 \leq i \leq m$.

Using the query vector $\vec{q}$ and the match functions above, $f_{impressionist}[s_1, s_{18}] = 0.99$, $f_{japanese}[s_1, s_{18}] = 1$, and $f_{art}[s_1, s_{18}] = 0.01$. Thus $\vec{f} = \langle 0.99, 1, 0.01 \rangle$.

Using a vector corresponding to each extent with respect to a given query, define a score of an extent that reflects the relevance of the extent to the query.

Definition 7 (Score) *A* score $S[u, v] \geq 0$ *of an interval* $[u, v]$ *depends on a term frequency vector* $\vec{f}$, *and the length of the interval and can also be written as* $S\left(\vec{f}, l\right)$.

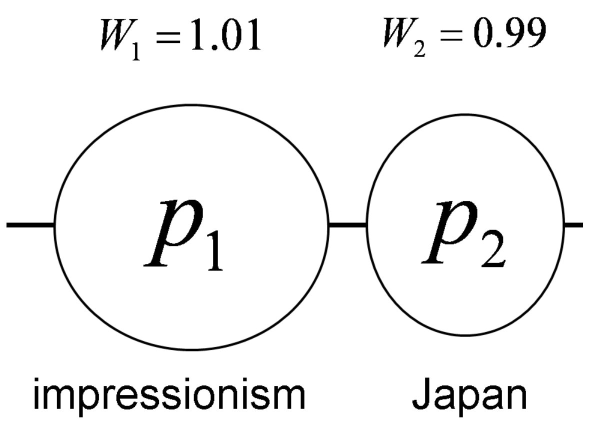


Figure 3.2: Symbols with Positive Matches

When scoring an extent, I am only interested in those symbols in $\mathcal{C}$ that have positive *match* scores with $\vec{q}$. Otherwise, the score of an extent $[u, v]$ that starts and ends with non-positive *match* scores is always lower than the score of an extent it contains that starts and ends with positive match scores.

Definition 8 *Given a query term t, let $n_t = \sum_{1 \le i \le N} (match(s_i, t) > 0 : 1 : 0)$, and let $n = \sum_{t \in \vec{q}} n_t$. Thus n_t is the number of symbols in $\mathcal{C}$ that has a positive match with t, and n is the number of symbols in $\mathcal{C}$ that have positive match with the given query terms. Consequently, let $(p_1, p_2, \ldots, p_n)$ be a subsequence of $(s_1, s_2, \ldots, s_N)$ for which $match(s_i, t) > 0$ for all i. Each p_i has its weight W_i associated with it, where $W_i = \sum_{t \in \vec{q}} match(s_i, t)$.*

In Figure 3.2, with the same $\vec{q}$ and match functions as before, $W_1 = 0.99$ because p_1 almost matches one of the terms, "impressionist".

There are many ways to select extents out of $O(n^2)$ possible extents. When a user is presented with the extents in decreasing order of scores, it is a reasonable assumption that the user do not want to read the same part of extents over and over. The redundancy is considered an *overlap*.

Definition 9 (Overlap) *An extent $[u', v']$ overlaps another extent $[u, v]$ when either $u \leq u' \leq v$ or $u \leq v' \leq v$.*

Accordingly, define the problem of passage retrieval.

Definition 10 (Disjoint Passage Retrieval) *Given a corpus C, find a set $\mathcal{D}$ of intervals, $[u_1, v_1], \ldots [u_k, v_k]$ such that*

1. $S[u_1, v_1] > S[u_{2,2}] > \ldots > S[u_k, v_k]$.

2. $\nexists u', v' \in [p_1, p_2, \ldots, p_n]$ such that

(a) $S[u_i, v_i] < S[u', v']$,

(b) $[u', v']$ overlaps $[u_i, v_i]$, and

(c) Covers all corpus.

Therefore, $\mathcal{D}$ is a set of disjoined extents that have highest scores among all those that overlap them. In Figure 3.2, the only two possible $\mathcal{D}$ is either $[p_1, p_1]$ and $[p_2, p_2]$, or $[p_1, p_2]$.

Although there are $O\left(n^2\right)$ extents in $\mathcal{C}$, there are at most n extents in $\mathcal{D}$.

Finding $\mathcal{D}$ takes $O\left(n^2\right)$ time using a brute force approach by computing scores for each possible $O\left(n^2\right)$ extents. The lower bound for this problem is $\Omega\left(n\right)$ as we need to take a look at all vocabularies in $\mathcal{C}$ to compute term frequencies. Therefore, an algorithm that finds sets $\mathcal{D}$ in linear time may be optimal.

Assume the scoring function to possess the following properties that are typical of IR scoring functions:

1. **For a fixed length, the higher term frequencies, the higher the score.**
   $S\left(\langle \ldots, f_i, \ldots \rangle, l\right) < S\left(\langle \ldots, f_i + \epsilon, \ldots \rangle, l\right)$, for $\forall \epsilon \geq 0$.

2. **For a fixed term frequencies, the longer the interval, the lower the score.**
   $S\left(\vec{f}, l\right) \geq S\left(\vec{f}, l+\epsilon\right)$, for $\forall \epsilon \geq 0$.

3. **Computing a score takes a linear time to the size of query.**
   In this Chapter, the size of query is $O\left(m\right)$, therefore, computing a score must take $O\left(m\right)$ time.

There are three well-known *document* scoring functions that satisfy the above properties; pivoted length normalization [128], BM25 [122], and language modeling [114]. I decided to adopt pivoted length normalization into scoring *extents* because of its versatility. Later, in Section 3.4, I reduce BM25 and language modeling into the same form as the score using pivoted length normalization.

Using the formula in Section 2.1, define a passage score as

$$S[u,v] = \frac{\sum_{t \in \vec{q}} f_t \cdot w_t}{s \times l + \Delta} \quad , \tag{3.1}$$

where w_t is a weight of a symbol t, in this case set to an IDF of t and $\Delta = (1.0 - s) \times p$ is a constant for a pivot p and a slope s. From Equation 2.1, set $f\left(\cdot\right)$ and $g\left(\cdot\right)$ as the identity function, use an extent length as `old_normalization`, and train for s and p. Therefore, a score of an extent is the sum of all weighted symbols per extent length with Δ acting as a length normalization factor. Using Definition 8, Equation 3.1 becomes

$$\sum_{p_i \in [u,v]} \frac{W_i}{l + \Delta} \quad . \tag{3.2}$$

3.3 Reduction to Density Problem

In this section, I use my proposed scoring function Equation 3.2 to try to come up with an algorithm that solves the problem 10 in almost linear time. I first make the problem simpler by finding the highest scoring extent only.

An algorithm $theHighest(u, v)$ describes the general framework for finding the single highest scoring extent within an extent $[u, v]$. It depends on having a function that finds the highest scoring extent that includes the middle point $[\lceil (v-u+1)/2 \rceil, \lceil (v-u+1)/2 \rceil]$. I call $theHighest(1, n)$ to cover the entire corpus.

Algorithm 1 $theHighest\,(u, v)$

```
if $v = u$ then
    return $[u, u]$.
end if
if $v = u + 1$ then
    return $\operatorname{argmax}_{[a,b] \in \{[u,u],[u,v],[v,v]\}} S[a, b]$
end if
$M \leftarrow \lceil (v-u+1)/2 \rceil$.
$[l_S, l_E] \leftarrow theHighest\,(u, M)$
$[r_S, r_E] \leftarrow theHighest\,(M, v)$
Find the highest scoring extent $[m_S, m_E]$ that includes $[M, M]$, where $m_S \neq M$ and $m_E neq M$.

return $\operatorname{argmax}_{[a,b] \in \{[l_S,l_E],[r_S,r_E],[m_S,m_E]\}} S[a, b]$
```

Theorem 1 *An algorithm $theHighest(u, v)$ correctly returns the highest scoring extent in time $O(T \log n)$, where T is the time it takes to find the highest scoring extent $[m_S, m_E]$ that includes the middle point, $[\lceil (u+v)/2 \rceil]$.*

Proof 1 *The highest scoring extent must lie either within the left or the right half, or on both halves. The first two cases are handled by recursions and finding the highest scoring extent that includes the midpoint $[m_S, m_E]$ takes $O(T)$. Thus $theHighest(u, v)$ runs correctly in time $O(T \log n)$.*

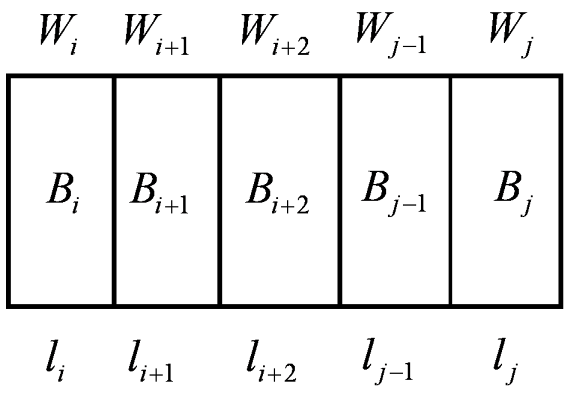


Figure 3.3: A Sequence of Blocks (B_i, B_j)

Next, I reduce the problem of finding an extent $[m_S, m_E]$ that includes p_M into a density problem. First, define elements that constitutes the density problem.

Definition 11 (Block) *A block B_i has a single weight W_i and length l_i associated with it. A sequence of blocks $(B_i, B_{i+1}, \ldots, B_j)$ is denoted (B_i, B_j) and corresponds to Figure 3.3*

Definition 12 (Density) *A density of a sequence of blocks (B_i, B_j) denoted $d(B_i, B_j)$ is a sum of weights associated with the blocks divided by the sum of lengths associated with the blocks. In Figure 3.3,*

$$d(B_i, B_j) = \frac{W_i + W_{i+1} + \ldots + W_j}{l_i + l_{i+1} + \ldots + l_j} \quad .$$

Figure 3.4 describes the mapping of extents so that the densest sequence of blocks that includes p_M will correspond to the highest scoring extent that includes p_M. For each p_i for $i < M$, we associate a block B_i with a weight W_i and a length $p_{i+1} - p_i + 1$. We assign a block B_M for p_M with a length Δ. To each p_i for $M < i$, we associate a block B_i with weight W_i and length $p_i - p_{i-1} + 1$.

Theorem 2 *Finding the highest scoring extent $[m_S, m_E]$, that includes p_M is equivalent to*

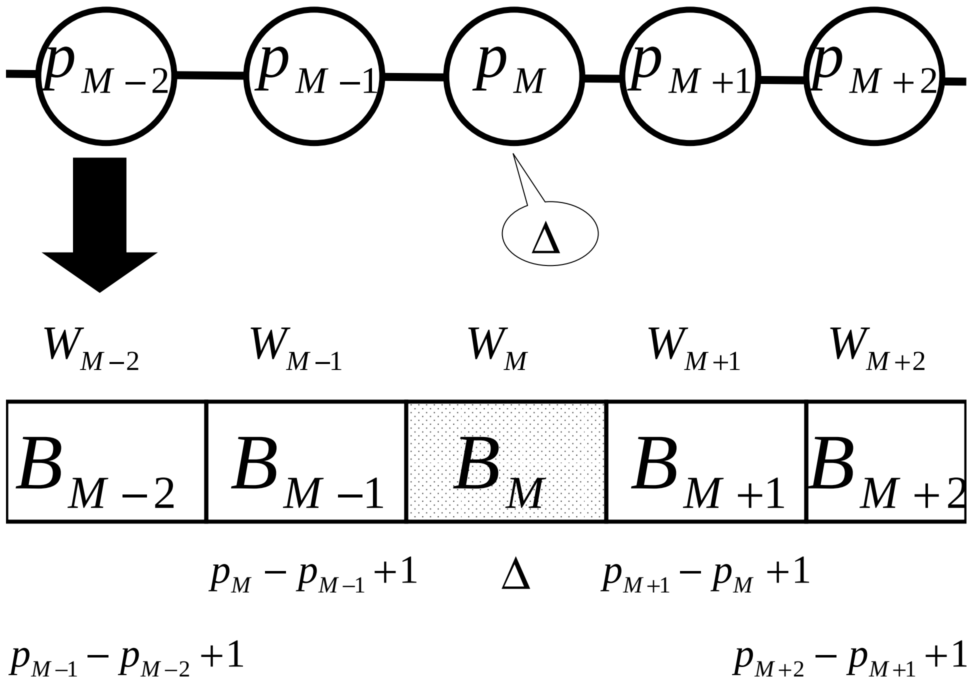


Figure 3.4: Constructing Blocks from Points

finding the densest sequence of blocks that includes p_M, where the length of the block B_M is augmented by a constant $\Delta > 0$.

Proof 2 *To find $[m_S, m_E]$, we need to add the normalization constant Δ to the length of p_M as in Figure 3.4. This way, every extent that includes p_M have Δ added to its length. Then we only need to find the densest sequence of blocks among those that include p_M to find the highest scoring extent that includes p_M.*

Therefore, an algorithm to find the highest scoring extent with respect to Equation 3.2 takes as much time as finding the densest extent that contains the middle block, times a $\log n$ factor.

3.4 Canonical Passage Scoring Function

This Section transforms the single-term BM25 and language model scoring functions into a form similar to my proposed function in Equation 3.2. This way, it may be possible in the future to derive a sub-quadratic algorithm to solve Problem 10 in a manner similar to my proposed scoring function using a real-world scoring function.

3.4.1 BM25

From BM25 scoring function for documents and using notations in Section 3.2, define a score of an extent using BM25 as,

$$S[u,v] \equiv \sum_{t \in \vec{q}} w_t \frac{f_t[u,v]\,(k_1+1)}{f_t[u,v] + k_1 \left(1 - b + b\frac{l}{avgl}\right)} \quad , \tag{3.3}$$

where w_t is a logarithm of the relative number of extents containing a symbol t in a collection and $avgl$ is an average extent length in the collection. Parameters $k_1 > 0$ and $b = \{b : 0 < b < 1\}$ are tuned to optimize the scoring scheme.

Theorem 3 shows how we can reduce the problem of comparing one-term BM25 scores of two extents into that of comparing densities of these two extents.

Theorem 3 *An extent $[u,v]$ has a higher BM25 score than an extent $[u',v']$ if its density computed by adding $\Delta = avgl\,(1-b)\,/b$ to its length is larger than that of the other extent. That is, for two extents $[u,v]$ with length l_1, and $[u',v']$ with length l_2,*

$$S\,[u,v] \geq S\,[u',v'] \tag{3.4}$$

$$\Leftrightarrow \quad \tfrac{\vec{f}}{l_1+\Delta} \geq \tfrac{\vec{g}}{l_2+\Delta} \quad . \tag{3.5}$$

Proof 3 *Let $t_1 = \vec{f}$ $t_2 = \vec{g}$, $A = 1/\,(k+1)$, $B = kb/\,((k+1)\,avgl)$, and $C = k\,(1-b)\,/\,(k+1)$.*

Then,

$$S\left[u,v\right] \geq S\left[u',v'\right] \tag{3.6}$$

$$\Leftrightarrow \quad \tfrac{t_1}{At_1+Bl_1+C} \geq \tfrac{t_2}{At_2+Bl_2+c} \tag{3.7}$$

$$\Leftrightarrow \quad \tfrac{t_1(At_2+Bl_2+C)-t_2(At_1+Bl_1+C)}{(At_1+Bl_1+C)(At_2+Bl_2+C)} \geq 0 \tag{3.8}$$

$$\Leftrightarrow \quad At_1t_2 + Bt_1l_2 + Ct_1 - At_1t_2 - Bt_2l_1 - t_2C \geq 0 \tag{3.9}$$

$$\Leftrightarrow \quad t_1l_2 - t_2l_1 \geq \tfrac{C}{B}\left(t_2 - t_1\right) \tag{3.10}$$

$$\Leftrightarrow \quad t_1\left(l_2 + \tfrac{C}{B}\right) \geq t_2\left(l_1 + \tfrac{C}{B}\right) \tag{3.11}$$

$$\Leftrightarrow \quad \tfrac{t_1}{l_1+\frac{C}{B}} \geq \tfrac{t_2}{l_2+\frac{C}{B}} \tag{3.12}$$

$$\Leftrightarrow \quad \tfrac{t_1}{l_1+\Delta} \geq \tfrac{t_2}{l_2+\Delta} \quad . \tag{3.13}$$

Equation (3.8) is obtained by expanding (3.7). I canceled the denominator when moving from (3.8) to (3.9) because $C = k\left(1-b\right)/\left(k+1\right)$ *is positive because of our assumption that* $k > 0$ *and* $0 < b < 1$*. From (3.12) to (3.13) I used the fact that* $C/B = avgl\left(1-b\right)/b = \Delta$ *because,*

$$\frac{C}{B} \quad = \tfrac{k(1-b)}{k+1}\tfrac{(k+1)avgl}{kb} \quad = \frac{avgl\left(1-b\right)}{b} \quad . \tag{3.14}$$

3.4.2 Language Modeling

Following notations in Section 3.2, a score of an extent using language modeling with Dirichlet smoothing is defined as [141]

$$S[u,v] \equiv \sum_{t\in\vec{q}} \log\left(1 + \frac{f_t[u,v]}{\mu}\frac{l_{\mathcal{C}}}{n_t}\right) - m\log\left(1 + \frac{l}{\mu}\right) \quad , \tag{3.15}$$

where μ is a parameter, $l_{\mathcal{C}}$ is the length of the corpus $\mathcal{C}$, n_t is frequency of t in the entire corpus $\mathcal{C}$, and m is the number of terms in the query.

For a single term, (3.15) becomes

$$\log\frac{\left(1 + \frac{f_t[u,v]}{\mu}\frac{l_{\mathcal{C}}}{n_t}\right)}{\left(1 + \frac{l}{\mu}\right)} \quad .$$

Letting $t_1 = \vec{f}\ t_2 = \vec{g}$, $\alpha = (l_{\mathcal{C}}) / (n_t \mu)$ and $\beta = 1/\mu$, we get

$$S[u,v] \geq S[u',v'] \tag{3.16}$$

$$\Leftrightarrow \quad \tfrac{t_1\alpha+1}{\beta l_1+1} \geq \tfrac{t_2\alpha+1}{\beta l_2+1} \quad . \tag{3.17}$$

Since both α and β are constants, I can use theorem 2 by setting $\Delta = 1$, scaling all lengths by β, scaling all term frequencies by α, and adding 1 to the term frequency of B_M.

3.4.3 Discussion

It is interesting to note that the language model score is derived from quite a different perspective than BM25, that is a variation of tf-idf. Language model score, however, is derived from a Markov model. Both models, at least at the single-term level, reduce similar to my proposed function, Equation 3.2 that is a very simple form of pivoted length normalization. For all three scoring functions, what I show was that, when comparing the two overlapping extents, we only need to compare the density of the extents. The additional Δ does not necessarily need to be inserted in the middle. A much longer extent whose adjusted density (including Δ in the overlapped point) is higher than a shorter extent would be selected over the shorter extent.

Following the pattern of how the three scoring functions are formed in the setting of passage retrieval, I could generalize a single-term scoring function as follows.

$$S[u,v] = \frac{t\alpha + \gamma}{l\beta + \Delta} \quad , \tag{3.18}$$

where α, β, γ, and Δ are either constants or combinations of parameters to train. From the generalized single-term scoring function, it may be possible to derive a generalized multiple-term scoring function that is more versatile than BM25 and language model (it may require significant tunings), akin to the work of Fang and Zhai [30], but in the passage retrieval setting.

3.5 Conclusions

Chapter 3 establishes a formal model of focused retrieval. I define the goal of focused retrieval is the retrieval of passages that cover the corpus in the decreasing order of scores in which no overlapping extent whose score is lower than the others appear. I partially solve the first problem of focused retrieval in Chapter 1, to reduce the running time, by showing that a linear time algorithm that finds the densest extent that contains the mid point leads to the log-linear time to find the highest scoring extent according to my proposed scoring function that depends on pivoted length normalization. I finally show that the comparing the single-term BM25 and language model scores is similar to that of comparing my proposed function.

Chapter 4

Basic Retrieval Strategy

Chapter 1 and Chapter 2 introduce the importance of building a heterogeneous focused search system. Chapter 3 defines the problem of passage retrieval. This chapter takes the theory established in Chapter 3 into practice. It describes the implementation details of the basic search system I implement that is capable of performing focused Wikipedia and book search. The next two Chapters 5 and 6 then builds upon this focused search system to solve the two main questions in this thesis; what is the best unit of retrieval, and how can we take advantage of structure?

Section 4.1 introduces the basic concept of my search system. Section 4.2 introduces the step-by-step implementation details. Section 4.3 concludes Chapter 4.

4.1 Introduction

Probably the major difference between my Ad Hoc search engine and those of others used in INEX is the ability to process XML element retrieval and arbitrary passage retrieval on the same ground. This enables me to compare the utility of applying XML markup over text.

Below are the basic steps taken in my focused search system. Slight variations in details and implementation exist across the years. All code is implemented using Perl v5.8.8 (most of the code) or C++ (to score passages and finding XPaths) and executed on 2,814MHz AMD Opteron Processor 154 with 1,880MB of memory. In addition, Java is used to run evaluation software provided by INEX. The steps are:

1. Pre-process the corpus.
2. Locate XML tags using Wumpus.
3. For each XML element indicated by the Wumpus term-index, build a lookup table containing XPath.
4. Locate the query terms in a corpus using Wumpus.
5. Compute Inverse Document Frequency (IDF).
6. Compute passage scores.
7. Remove overlap.
8. Return the top results as indicated by INEX specification.
9. Train

4.2 Indexing

4.2.1 Step 1: Pre-Processing

The 6GB Wikipedia corpus [25] contains comments that I remove prior to indexing as much as possible. However, the wild nature of Wikipedia left some characters, notably angular brackets in equation, unescaped, limiting comment removal. For earlier experiments on the 6GB Wikipedia corpus and other corpora, comments are not removed.

The book corpus [40, 77] is originally 400GB in size. In 2008, I partitioned parts of the corpus across thirteen computers. In 2009, I strip all tags except `<document>`,`<page>`, `<region>`, `<section>`, and `<line>`, and replaced these tags to shorter versions, shrinking the original size into 37GB, smaller than 50GB Wikipedia corpus of that year.

4.2.2 Step 2: XML Tag Indexing

Instead of using an off-the-shelf XML parser, I use the text search engine Wumpus [1]. The main reason to use this text search engine over an XML parser is to compare arbitrary passage retrieval against XML element retrieval using the same approach. The Wumpus search engine indexes files term by term, thus for a given query it returns the term positions such as 1 for the first term in the first file it indexed.

The Wumpus search engine has limited XML capacity. The search engine has difficulty retrieving XML tags because it ignores non-alphanumeric characters, including angular brackets, even though search is performed verbatim. For example, `<article>` is considered identical to `<article>,` , but different from `</article>` and `<article>>`. It also cuts a term containing non-alphanumeric letters into pieces, which is problematic when identifying an entire tag. Thus my search engine implementation have followup lines to handle the Wumpus problems, but it may not be complete. For example, to make sure I do not misidentify the opening tags with closing tags or some other terms, I parse through the markup trees to check if all opening tags are closed appropriately. For the book collection, I replaced the original tags with simpler tags that do not interfere with Wumpus. In the future, I plan to also pre-process the Wikipedia corpus.

[1] `www.wumpus-search.org/`

4.2.3 Step 3: Pre-compute XPath

At the indexing time, I create a table containing a list of XML tags in the order of appearance. Initially, the specification of XPath for an element involved parsing the table top down, however, doing so every time we encounter the same element during training takes time. Instead, I pre-computed XPaths for all elements in the corpus to act as a lookup table. For the 50GB Wikipedia corpus [125], the number of XML tags is too large to map efficiently, so only the elements of interest are mapped to abbreviated XPath (because I did not query the position of all distinct 32311 tags that exist in the 50GB Wikipedia corpus). INEX provides participants with a program, `XML2FOL.jar` [2], to convert XPaths into a different format. In doing so, the program lists all XPaths for XML elements of a minimum size. Thus I use abbreviated XPaths to match the full XPaths in the list to return.

Since 2009, INEX requires passage results to be in file-offset-length (FOL) format [45]. The format assumes a plain text version of the corpus. Both the offset and length are counted as numbers of utf-8 characters. Since Wumpus does not handle character-counts, I retrieve the actual passages to return from Wumpus, remove XML tags, and count the number of utf-8 characters. The retrieval of actual passages takes about eight hours for the first run. The time is dramatically shrunken during training, by recording the Wumpus-term-position to a FOL map.

4.2.4 Step 4: Obtain Query Term Positions

I process queries by converting each topic title into a disjunction of query terms without negative query terms. Positive terms are treated as regular terms. Stop words are removed, the terms are lower-cased, and the stemming operator, indicated by '\$' is applied on Wumpus query.

For example, the topic in Figure 1.3 is converted into,

[2]Downloadable from `http://www.inex.otago.ac.nz/data/software.asp`

```
"$impressionist","$japanese","$art"
```

Topic 607 in INEX 2008 Ad Hoc track,

```
law legislation act +nuclear -family
```

is converted into

```
"$law","$legislation","$act","$nuclear".
```

4.2.5 Step 5: Compute IDF

Corpus-level IDF values are computed for each query term. Specifically, an IDF of a term t is

$$\mathrm{IDF}\,(\mathrm{t}) = \frac{\text{the number of documents in corpus}}{\text{the number of times } t \text{ appears in corpus}}.$$

At this time I also compute an average document length.

4.2.6 Step 6: Score Passages

In scoring XML elements, I extract following tags to score:

IEEE Collection `<abs>`, `<app>`, `<article>`, `<bb>`, `<bdy>`, `<bm>`, `<fig>`, `<fm>`, `<ip1>`, `<li>`, `<p>`, `<sec>`, `<ss1>`, `<ss2>`, `<vt>`.

6GB Wikipedia Collection `<p>`, `<section>`, `<normallist>`, `<article>`, `<body>`, `<td>`, `<numberlist>`, `<tr>`, `<table>`, `<definitionlist>`, `<th>` ,`<blockquote>`, `<div>`, `<li>`, `<u>`.

50GB Wikipedia Collection `<p>`, `<sec>`, `<list>`, `<article>`, `<bdy>`, `<column>`, `<row>`, `<table>`, `<entry>`, `<indent>`,`<ss1>`, `<ss2>`,`<ss3>`,`<ss4>`,`<ss5>`.

2008 Book Corpus `<document>`,`<page>`, `<region>`, `<section>`.

2009 Book Corpus `<document>`, `<page>`.

The choice of tags in IEEE and Wikipedia collections base on the original selection of Clarke [14] for IEEE collection. Clarke hand picks the tags based on INEX 2003 test set. The IEEE collection on my experiments use the same set of tag as Clarke. For the Wikipedia collections, I choose the corresponding tags. For the book collection, from the list of tags used in tag-abbreviated version of the corpus downloadable from the Book Search site [3], I select tags that are larger than `<section>`, which is the smallest unit of retrieval in 2008. In 2009, only the page element, in addition to the document element is used because the evaluation in 2008 was based on page-level.

I ignore any elements or passages that are smaller than a certain threshold, mostly 25.

I use a version of BM25 [121] to score passages.

$$s(E) \equiv \sum_{t \in Q} W_t \frac{f_{E,t}(k+1)}{f_{E,t} + k(1 - b + b\frac{pl_E}{avgdl})} \quad , \tag{4.1}$$

where Q is a set of query terms, W_t is an IDF value of the term t in the collection, $f_{E,t}$ is the sum of term frequencies in an element E, pl_E is an element length of E, and $avgdl$ is an average document length in a collection to act as a length normalization factor.

When scoring an XML element, instead of just scoring the body of the element, I consider the textual context inside the XML element tag. For example, the `<a href>` tags contains the URL of the page to which an anchor phrase link to. Oftentimes the URL address contains the phrase that is also an anchor phrase. This may help increase the accuracy of retrieval. Finally, I include XML tags as part of length calculation, but they should be excluded for future work.

[3]`http://www.booksearch.org.uk/`

4.2.7 Step 7: Remove Overlap

All possible elements/passages are ranked according to the score. I eliminate overlap by going down the list and see if the current result in the list overlaps with previously retrieved list. This may not necessarily be the ideal way to remove overlap. In Section 5.3 I discuss how it could cause a problem.

4.2.8 Step 8: Return Results

Results returned are then evaluated using assessments provided by INEX.

4.2.9 Step 9: Parameter Tuning

I change the value of one parameter incrementally while having other parameters fixed. For my earlier runs, I only perform one iteration. In the later runs I perform multiple iterations until the parameters stabilizes or cannot increase the score for 10 iterations. It turns out for two-parameter tuning, a single iteration is sufficient because either the parameters stabilize or the best training scores increases only slightly after the first iteration. Some of the runs used average document length as a parameter, but later experiments in Chapter 5 and Chapter 6 show that it seems to lead to overtraining; during training it produces slightly better results, but during testing, the score is slightly lower.

4.3 Conclusion

The basic retrieval strategy scores all XML elements of interest using BM25. Overlaps are removed top down that seems to be effective on single XML element retrieval but may not be so in retrieval of passages of different sizes.

Chapter 5

What to Retrieve?

Chapter 3 defines the problem of passage retrieval. Based on the model, Chapter 4 lays out the implementation details. This chapter uses the same approaches to conduct two sets of experiments in order to answer the second question of focused search defined in Chapter 1: *What is the ideal unit of retrieval*?

The first experiments try to find a way to convert best passages into single elements. I show that scoring single elements is better than trying to turn passages into elements. The second experiments directly compares the performance of focused retrieval based on five different granularity of retrieval; single element, range-of-elements, semi-arbitrary passage, arbitrary passage, and article. I show that even in this scenario, single element seems to suffice as a good unit of retrieval.

The experiments use BM25 [122] as a scoring function. The choice of BM25 among others is simply because of familiarity and popularity. Pivoted length normalization [128] is not widely used. Language modeling [114] is equally popular, but because the canonical form implies that the basic version of both BM25 and Language Modeling shares the same philosophy, I choose to work on BM25. Finally, most of my INEX runs are based on BM25, so we can see the results judged by an objective third-party.

Section 5.1 introduces the problem of passage size. Section 5.2 suggests returning single XML elements are not good enough. Section 5.3 shows that for using BM25, single XML elements produces results good enough. Section 5.4 concludes Chapter 5.

5.1 Introduction

Using the notations in Chapter 3, suppose a corpus contains N documents, and there are n incidents of a positive match with a query term vector, $\vec{q}$. In document retrieval, there are only N possible documents to return, thus the unit of retrieval is a document. In passage retrieval, the unit of retrieval can range from a single match occurrence to as long as the entire sequence of n match occurrences. It is conceivable to score all $O(n^2)$ possible passages, but can we do better? One way to decide on the passage size to return is to use XML markup on text documents.

5.2 Turning Passage into XML Elements

Originally in INEX Ad Hoc Track, the unit of retrieval is single XML elements [39, 51, 95–97]. The most intuitive way therefore is to score single XML elements. Trotman and Geva [133], however, show that when human assessors perform relevance judgement by highlighting any parts of documents, be it a single character, a paragraph, or an entire article, the highlighted passages do not necessarily correspond to XML elements. Thus it seems that obtaining relevant passages first and then converting them into single XML elements may be more appropriate. To transition into passage based XML retrieval, Trotman and Geva propose two ways to convert passages into XML elements for the focused retrieval task;

TG+ Return the smallest XML element that fully contains the relevant passage.

TG- Return the largest XML element that is fully contained in the relevant passage.

In Section 5.2, I implement TG+ algorithm on INEX 2005 test set and compare its performance against simply scoring single XML elements. I show that TG+ algorithm has less precision than single XML element retrieval. This suggests that direct comparison of arbitrary passage retrieval against single XML elements may be more appropriate than trying to convert high scoring passages into single XML elements.

5.2.1 Experiments

As a test collection, I use INEX 2005 IEEE collection and the query topics provided for the INEX 2005 Ad Hoc Track. The corpus contains 16,819 files from various IEEE journals from 1995 to 2004. There are 39 topics, each containing a set of query terms. In INEX 2005, the results are assessed using the nxCG metric [78]. I only consider the generalized quantization because this is the only metric with published ranking in both INEX 2005 and 2006. I implement three algorithms:

Arbitrary Passage Retrieval (AR) An arbitrary passage starts and ends at a query term. I use this as a gold standard.

Single Element Retrieval (SR) Scores and return only single XML elements.

TG+ Retrieval (TG+)

My implementation of TG+ algorithm is as follows. I first score all arbitrary passages as in Arbitrary Passage Retrieval. Next, for each passage, I assign the smallest element that fully contains the passage. The score of an element is then the score of the corresponding passage. When more than one passage is assigned to the same element, I assign to the element the best

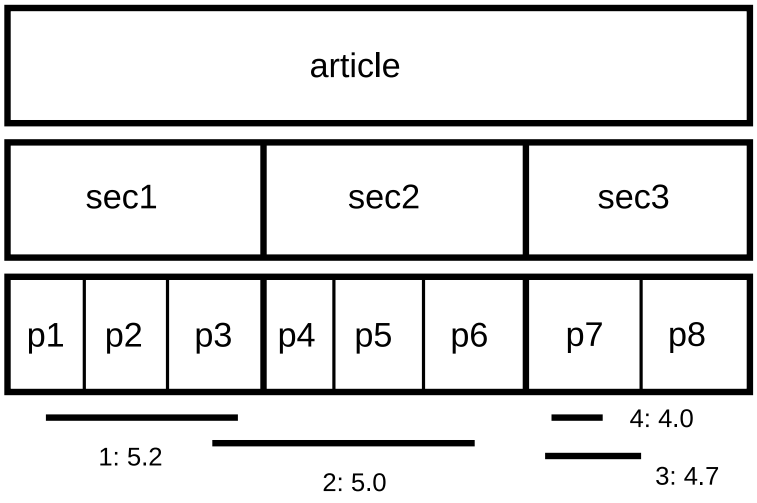


Figure 5.1: Assigning Scores and Nesting Elimination in TG+ Retrieval

passage score. Now that we only have a set of single elements, I remove overlap as in single element retrieval.

Figure 5.1 illustrates TG+. Suppose there are four passages with scores; passage 1 with a score of 5.2 to passage 4 with a score of 4.0. Passage 1 spans through paragraph 1, 2, and 3, which are under section 1 and an article. Similarly for other passages. After computing scores for these passages, I assign scores to corresponding XML elements. The score of 5.2 is assigned to `sec[1]`, the smallest element containing passage 1. Similarly, the score of 5.0 is assigned to `article`. Because passage 3 has a higher score than passage 4, I assign the score of passage 3, 4.7 to `p[7]`. Next, I remove nested elements while taking the top scored elements. I first take the element with the highest score, `sec[1]` and assign a rank of 1. The element with the second highest score, `article` is eliminated because it causes a nesting of `sec[1]` within it. I can safely take the element with the next highest score, `p[7]` because it does not cause a nesting with `sec[1]`, and assign a rank of 2.

5.2.2 Evaluation and Analysis

Performance Against INEX 2005

I train both SR and TG+ retrieval over the INEX 2005 corpus and the query set. Figure 5.2 and Figure 5.3 show the scores for different values of k_1 with $b = 0.8$ using the nxCG metric, mean average precisions (MAep) and interpolated MAep (iMAep) in the CO.Focused Task. I then choose $k_1 = 10$ for TG+ retrieval and $k_1 = 4$ for element retrieval for training b as in Figure 5.4 and Figure 5.5. The results of nxCG and MAep/iMAep metrics seem correlated as both have similar curves and give maximum values at the same parameters. The values of both k_1 and b need to be quite large for both algorithms to perform well. Clarke [14] also points out

	nxCG[10]	nxCG[25]	nxCG[50]	MAep	iMAep
TG+	0.1856	0.1774	0.1633	0.0612	0.0387
Element	0.2586	0.2323	0.217	0.0929	0.0715

Table 5.1: TG+ and SR at $k_1 = 10$ and $b = 0.9$ in INEX 2005 CO-focused

this phenomenon for his BM25-based passage retrieval algorithm that is quite different from these two. Having a large k_1, therefore, seems to be necessary for using BM25. Section 5.2.3 examines this phenomenon further.

These figures suggest that the simple element retrieval performs much better than TG+ retrieval. To see why, I compare the results of both algorithms at $k_1 = 10$ and $b = 0.9$, which are optimal parameters for TG+ retrieval. Table 5.1 shows that even if at an optimal setting, TG+ retrieval performs significantly worse than element retrieval.

I analyze the submission files of TG+ and element retrieval and realized that in the TG+ algorithm, the majority of the elements returned have a large granularity; many of them are either `/article` or `/article/bdy`. In the element retrieval, however, the returned elements have much finer granularity such as paragraphs and sections.

For example, as we see in Figure 5.6, the first element returned for the first topic in both retrieval methods is from the same file, `ex/2001/x1026` that spans through positions 27040000 and 27046835. In TG+ retrieval, the element returned is `/article/bdy`, corresponding to positions 27040246 through 27045776 with a score of 42.53. The passage corresponding to this element that gives the score of 42.53 spans through positions 27042389 and 27043619. In element retrieval, the element returned is `/article/bdy/sec[4]` corresponding to positions 27042506 through 27043559 with a score of 40.90. We see that `/article/bdy`, the smallest element that contains the highest scoring passage, is much longer than the passage. However, /article/bdy/sec[4], the element returned in element retrieval contains much of the passage without much excessive text.

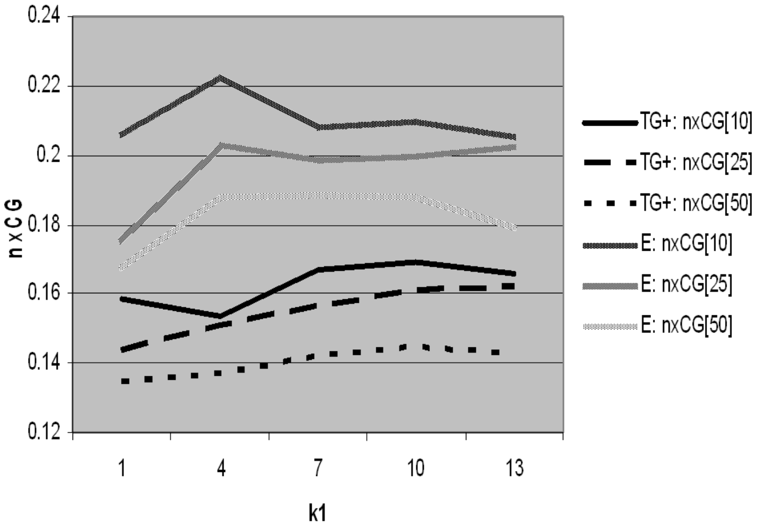


Figure 5.2: nxCG: Training on different k_1 with $b = 0.8$ in INEX 2005 CO-Focused

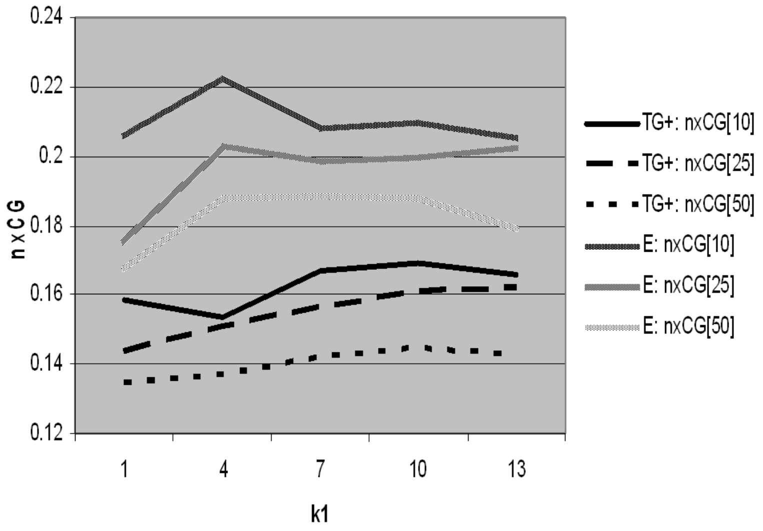


Figure 5.3: MAep/iMAep: Training on different k_1 with $b = 0.8$ in INEX 2005 CO-Focused

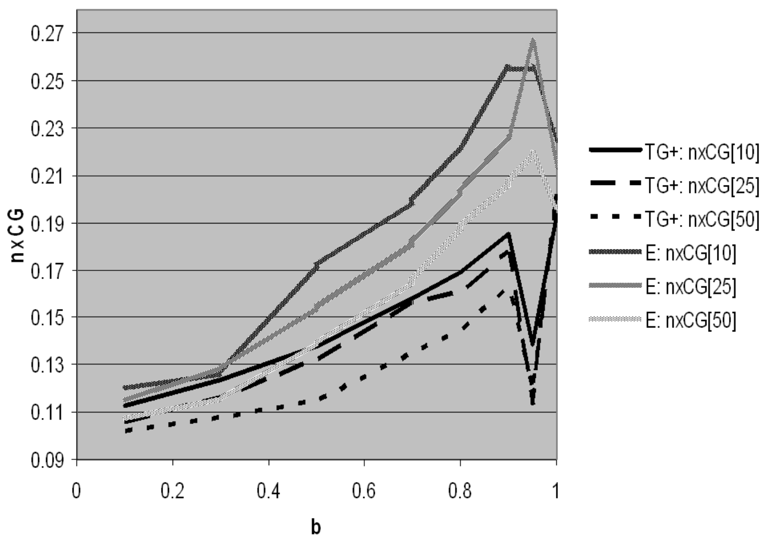


Figure 5.4: nxCG: Training on different b in INEX 2005 CO-Focused

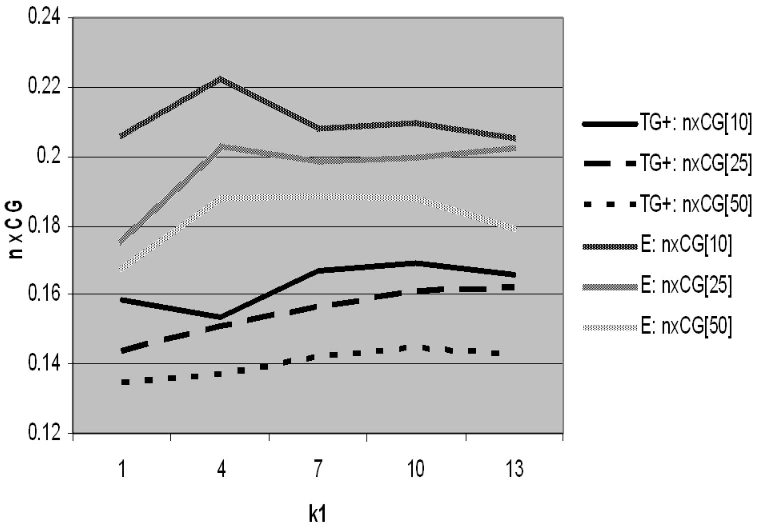


Figure 5.5: MAep/iMAep: Training on different b in INEX 2005 CO-Focused

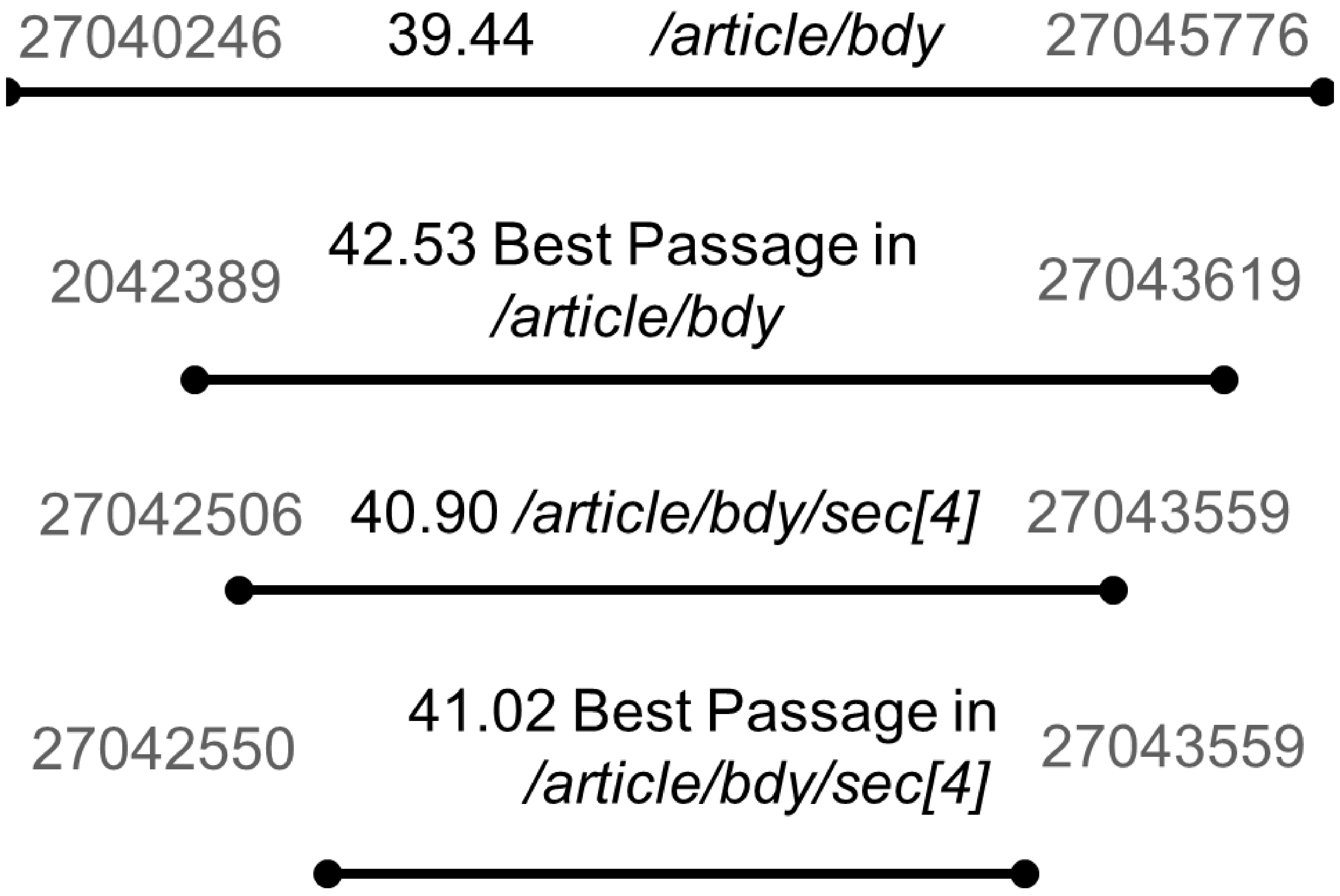

27040246
39.44
/article/bdy
27045776
2042389
42.53 Best Passage in /article/bdy
27043619
27042506
40.90 /article/bdy/sec[4]
27043559
27042550
41.02 Best Passage in /article/bdy/sec[4]
27043559

Figure 5.6: Elements and Passages Relating to Rank One Result

The element, `/article/bdy/sec[4]`, returned by element retrieval is not returned by TG+ retrieval because the highest scoring passage within the element that spans through positions 27042550 and 27043534 only scored 41.02, lower than the highest passage contained in `/article/bdy`. On the other hand, element retrieval does not return `/article/bdy` because its score, 39.44 is lower than the score of `/article/bdy/sec[4]`, 40.90.

The above observation suggests that the TG+ algorithm is not a good approach for converting passages into an XML element because it returns a lot of excessive text. The TG- algorithm, taking the largest XML element contained in a passage, would likely perform well for the same reason element retrieval performs well; the returned elements would be unlikely to contain too much excessive text. However, both element retrieval and TG- retrieval may miss too much text to reduce the overall performance.

Performance Against the Gold Standard

In Section 5.2.2, I observe that TG+ retrieval tends to return excessive text. I also speculate that element retrieval may be missing too much text. To measure how much text is missing or in excess for both algorithms compared to what users find relevant, I create a gold standard to which I compare the elements returned by TG+ and single element retrieval [1].

Gold Standard Because human assessors see passages when making relevance judgement, the gold standard may be passages. The best XML elements are those that cover the gold-standard passages sufficiently, but not much more. I create the "gold standard" using passage retrieval with the same parameters as TG+ retrieval, $k_1 = 10$ and $b = 0.9$. This is because as Table 5.1 indicate, even at the optimal parameter TG+ retrieval performance is

[1]INEX 2005 relevance assessments are at element level. From INEX 2006, passage level relevance assessments are available.

worse than that of element retrieval. Thus the choice of parameters may highlight what is wrong with TG+ retrieval.

Percentage Lack/Excess I compare a set of passages/XML elements returned by both TG+ and element retrieval against the gold standard at each rank up to 1500. The percent *lack* at rank r is defined as the percentage of the gold standard up to rank r that is not covered by the returned elements up to rank r. The figures are averaged over the topics.

Figure 5.7 and 5.8 show that SR misses more gold standard than TG+, but TG+ returns more irrelevant text than SR.

Perhaps the reason that element retrieval performs better on the nxCG, MAep, and iMAep metric for the focused task is because having excessive text is punished more than missing text. In the INEX Focused task, specificity is preferred to exhaustivity [95]. In the same manner, the TG- algorithm, that takes the largest element contained in the passage, will likely score high in the focused task. The TG+ algorithm would score high in other tasks that place preference on exhaustivity over specificity.

5.2.3 Discussion

In this section, I discuss the behavior of parameters k_1 and b in passage retrieval. I use an average document length for *avgdl* in computing a BM25 score for a passage in Equation 4.1. Because most passages are less than a size of a document, $|P|/avgdl$ is less than 1 most of the time. The result is that in the denominator, we multiply k_1 with something very small. Then if k_1 is also small, the denominator approaches to $f_{P,t}$, turning Equation 4.1 into

$$score(P) \equiv \sum_{t \in Q} W_t 1. \tag{5.1}$$

Therefore, one reason for a large k_1 value is to prevent Equation 4.1 from degenerating into Equation 5.1 when the length of P is smaller than the average document length. Similarly, a

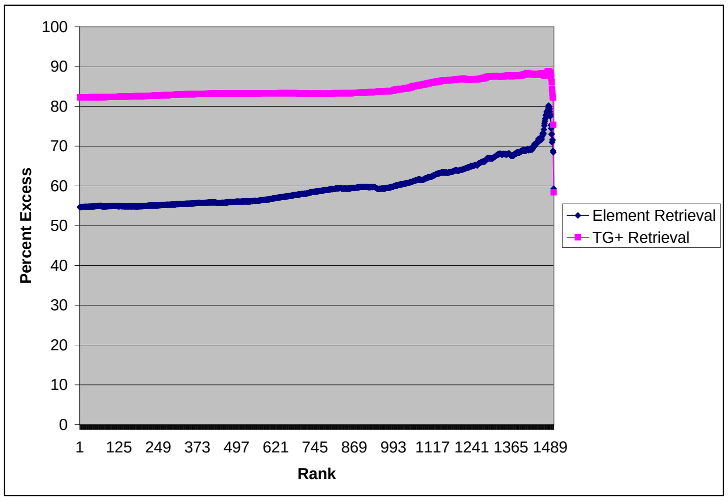


Figure 5.7: Percent Excess of the Gold Standard

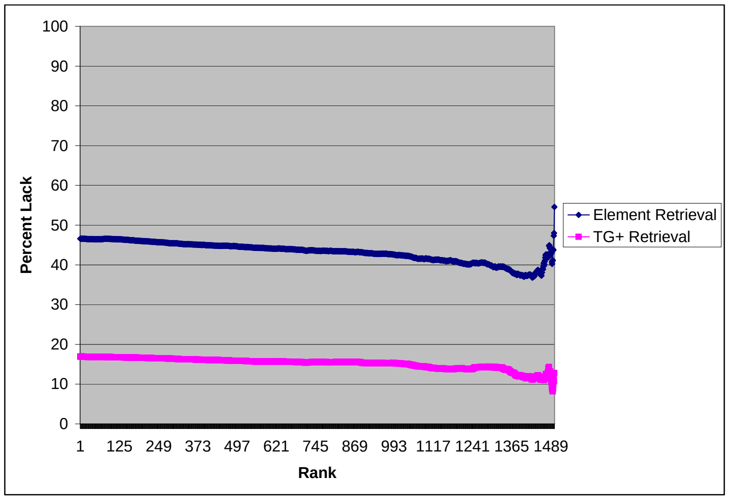


Figure 5.8: Percent Lack of the Gold Standard

	Final	Intermediate	Nested
Passage	1760.56	N/A	2424.85
Element	750.87	N/A	1354.69
TG+	3322.06	770.72	874.56

Table 5.2: Average Length in Passage, Element, and TG+ Retrieval

large b augments the effect of length normalization in $b|P|/avgdl$ in the denominator, and this sets off the large gap between the average document length and a passage length. Therefore, it appears that the choice of the average document length is balanced by the choices of k_1 and b.

To see how small the passage lengths are compared to the average document length, I compute various passage lengths. Table 5.2 shows the average length of the top 1500 final results (Final), the passages that produced the top 1500 final results (Intermediate) (only applicable to TG+), and the top 1500 results before the elimination of nesting (Nested), for passage, element, and TG+ retrieval. From the average document length of 6147.97, I compute the average passage length to be 1538.47. Then both the TG+ and the element algorithms retrieve fairly small elements/passages, 770.72 for TG+ retrieval, and 750.87 for element retrieval, compared to the average passage length of 1538.47. However, the average length of the elements returned by TG+ retrieval is about four times as large (3322.06) as the average length of the corresponding passages (770.72). This shows that the TG+ algorithm is inherently ineffective for the following reason. The passages returned by TG+ algorithm is about the right size because this is about the same size as the elements returned by element retrieval that performs well. Therefore, it is the way we assign passage scores to elements, rather than the choice of *avgdl*, k_1, and b, that makes TG+ retrieval less effective than element retrieval.

Finally, it is interesting to note that the average length of the passages returned by the passage retrieval, 1760.56 is close to the average passage length of 1538.47. The difference between the average lengths of the passages returned by TG+ and passage retrieval may be accounted for by how nesting is eliminated. In passage retrieval, I eliminate nestings of the passages that eliminates more passages than nesting elimination of elements in TG+ retrieval. Furthermore, the fact that the average passage length returned by passage retrieval is close (in fact more) than the average passage length implies that TG- retrieval would likely be able to find the largest element within the passages. However, I believe that the best elements to return in XML retrieval is multiple consecutive elements that contain passages just enough.

5.2.4 Related Work

Huang et al. [61] implement their own version of the TG+ algorithm, where the method of identifying passages are different from ours. Instead of considering all possible passages as I do, they consider a fixed size passages obtained from sliding windows. Moreover, they first perform document retrieval, and then runs passage retrieval on the top scoring documents, whereas I directly retrieve high scoring passages. When scoring passages, they use a simple term frequency approach and two language modeling approaches, whereas I use a variant of BM25, which is used for document retrieval.

The effectiveness of the TG+ algorithms of Huang et al. and mine can be easily compared because both of us use the same test set, the INEX 2005 IEEE collection, run the same CO.Focused task, and use the nxCG generalized quantization to score the results. Huang et al. compared their passage based XML retrieval results against the INEX 2005 CO-Focused task submissions of IBM Haifa that ranks 4th and of University of Amsterdam that ranks 28th in the Mean Average Precision (MAep) ranking. They conclude that their algorithm ranks between these two. My results of the TG+ algorithm at $k = 10$ and $b = 0.9$ with MAep of 0.0612 also ranks between these two. On the other hand, my element retrieval algorithm with the same parameters that has a MAep of 0.0929 ranks first before the first result of IBM Haifa that ranked 1st with a mean average precision of 0.0917 [95].

Huang et al. conclude the paper by mentioning that a passage retrieval algorithm can produce effective element retrieval results because it ranks between the 4th and the 28th out of 44 submissions. However, the fact that both versions of TG+ retrieval, despite with very different implementations of passage retrieval, *only* rank between the 4th and the 28th implies that TG+ retrieval is not a good idea for turning passages into an element. The comparison of my TG+ retrieval against my gold standard along with the average length statistics in Section 5.2.2 also imply that it does not perform well because TG+ algorithm always return excessive text, if the best passages do not match the element boundaries.

5.2.5 Conclusions

I implement three algorithms to test the effectiveness of the first algorithm of Trotman and Geva [133], which converts the results of passage retrieval into XML elements. This TG+ algorithm takes the smallest XML elements that contain the passages. I show that the TG+ algorithm does not perform as well as the simple element retrieval, where I score all XML elements, not passages. By comparing the result sets of the TG+ and the element retrieval algorithm, I realize that the TG+ algorithm tends to return a lot of excessive text. Even though element retrieval performs well, I speculate that it may miss too much text.

To see how much of text is missing or excessive, I create the gold standard, a set of results obtained from passage retrieval using the same BM25 parameters. I then compute an average percent excess and an average percent lack of text for returned results over all topics. Although element retrieval does miss out a lot of text, it scores well on the focused task of INEX because the task has a preference of specificity over exhaustivity. Because the TG− algorithm that returns the largest XML elements contained in the passages will not have much excessive text as in the case of element retrieval, the performance of the TG− algorithm may be similar to the one obtained by element retrieval. Ultimately though, the best results to return should be a series of XML elements.

5.3 Arbitrary Passages Vs. XML Elements

5.3.1 Introduction

In the INEX 2007 Ad Hoc Track, the granularity of passage to retrieve is liberated from single elements to arbitrary passages [33]. This makes it possible to compare if an arbitrary passage retrieval that ignores document structure can perform better than a single element (or range of elements) retrieval that considers structure. In Section 5.3 I implement five versions of my subdocument retrieval strategies in Section 4.2 and compare their performance to each other.

5.3.2 Experiments

I use the 6GB Wikipedia collection [25] as a corpus and a collection of assessments called *qrels* in FOL format from INEX 2006, 2007, and 2008 Ad Hoc Track Focused Task to compare five different subdocument retrieval strategies in which only the unit of a passage to score is different.

- **Single Element Retrieval (SingleE):** Retrieves single XML elements as in Chapter 4.
  Example: `/article[1]/section[1]/p[2]`
- **Range of Element Retrieval (RangeE):** Retrieves ranges of XML elements; starting and ending at single element boundaries listed in Section 4.2.6.
  Example: `/article[1]/section[1]/p[2]-/article[1]/section[2]/p[4]`

- **Semi-arbitrary Passage Retrieval (SemiP):** Retrieves passages starting at element boundary, ending anywhere.

  Example: `/article[1]/section[1]/p[2]-/article[1]/section[2]/p[4].40`

  (40 utf-8 characters after the beginning of the fourth paragraph.)

- **Arbitrary Passage Retrieval (ArbP):** Retrieves arbitrary passages.

  Example: `/article[1]/section[1]/p[2].20-/article[1]/section[2]/p[4].40`

  or in official INEX FOL notation, `1000 20 100`.

- **Article Retrieval (Article):** Retrieves entire articles.

  Example: `/article[1]/`

I trained on 2007 test set using average document length as parameter, and tested on 2006 and 2008 test sets. All evaluations are performed by `inex_eval.jar` [2] on the 2007/2008 official metric of iP[0.01].

5.3.3 Results and Analysis

Table 5.3 shows the results of the experiments. The score reported is the official metric of interpolated precision at 0.01 recall point (iP[0.01]).

The results are puzzling. One would think that an ideal retrieval system that retrieves more flexible unit, such as arbitrary passage retrieval, would perform at least as well as an ideal retrieval system that works with more coarse granularity. This seems true during training on 2007 set for single element, range of element, semi-arbitrary passage, and article retrieval.

However, on both 2006 and 2008 test set, it seems that single element retrieval performs better than any other runs. It seems as though they are over-trained. A similar over-training effect is observed in Chapter 6.

[2]Available at `http://www.inex.otago.ac.nz/data/software.asp`

Strategy	2008 Test Score	2006 Test Score	Best Training Score	Parameter (k,b,avg)
SingleE	0.6782	0.5879	0.5027	3, 0.8, 1100
RangeE	0.6721	0.5655	0.5104	4, 0.8, 1000
SemiP	0.6501	0.5535	0.5286	5, 0.8, 1000
ArbP	0.6092	0.5655	0.5095	4, 0.8, 700
Aritcle-only	0.6749	0.5236	0.4925	3, 0.8, 1000

Table 5.3: Different Granularity of Retrieval, Trained on iP[0.01]

	SingleE	RangeE	SemiP	ArbP	Article
SingleE	-	0.6338	0.1071	0.6846	0.6269
RangeE	-	-	0.1228	0.9553	0.4473
SemiP	-	-	-	0.1815	0.1284
ArbP	-	-	-	-	0.5327

Table 5.4: P-values reported on paired t-tests on 2007 Training Set, on iP[0.01]

	SingleE	RangeE	SemiP	ArbP	Article
SingleE	-	0.1983	0.0649	0.2450	***0.0051**
RangeE	-	-	0.3631	0.9964	0.0805
SemiP	-	-	-	0.4241	0.1955
ArbP	-	-	-	-	0.1034

Table 5.5: P-values reported on paired t-tests on 2006 Test Set (*=significant), on iP[0.01]

	SingleE	RangeE	SemiP	ArbP	Article
SingleE	-	0.7739	0.1951	***0.0057**	0.9004
RangeE	-	-	0.0965	***0.0005**	0.9019
SemiP	-	-	-	***0.0015**	0.3429
ArbP	-	-	-	-	***0.0282**

Table 5.6: P-values reported on paired t-tests on 2008 Test Set (*=significant), on iP[0.01]

Furthermore, even during training, arbitrary passage retrieval only performs close to range of element retrieval. This also holds true on the 2006 test set. One could argue that it is because elements are natural boundaries. However, single elements still perform better than range-of-elements and arbitrary passages. In addition, on 2008 test set, the results of arbitrary passage retrieval is much worse than range-of-element retrieval. The p-values reported from paired t-tests on Table 5.5 and 5.6 support this. Article retrieval seems to perform as expected, scoring the lowest of all runs except for 2008.

One theory is that the official metric of iP[0.01] is not an appropriate measure. Table 5.7 reports the number of characters returned at recall point of 0.01 and the average number of characters in a passage, computed from qrels. The average length at 0.01 recall is smaller than an average passage length for all years. Since it is likely that the set of all highlighted passages contains at least one type of the passage I experiment, achieving high scores on iP[0.01] does not require the choice of passage type.

Figure 5.9, Figure 5.10, and Figure 5.11 instead shows the performance of each type of runs at all recall points. The article runs seem to outperform other runs over most recall points. In general, the less flexible the run is, the higher the curve. Training on iP[0.01] itself may have caused the problem.

Table 5.7 also suggests that the gap between average highlighted passage lengths of 2006 and

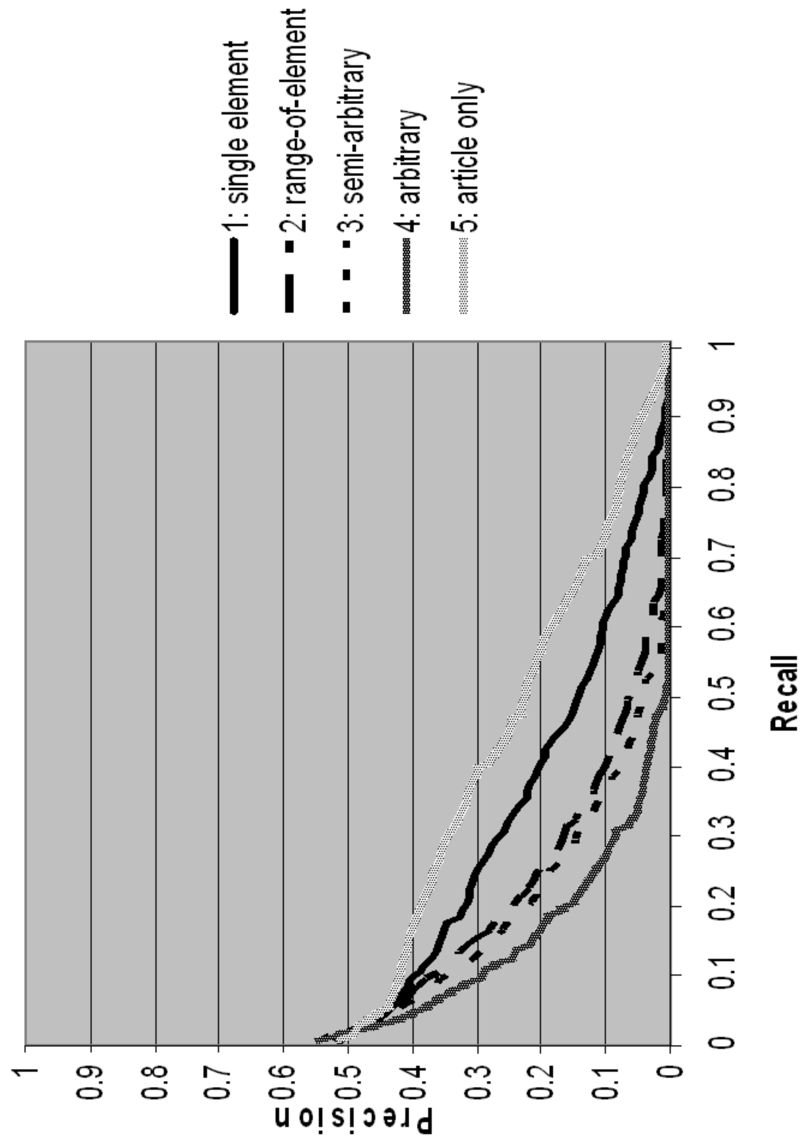


Figure 5.9: Precision Recall Curve on 2007 Training Set, Trained on iP[0.01]

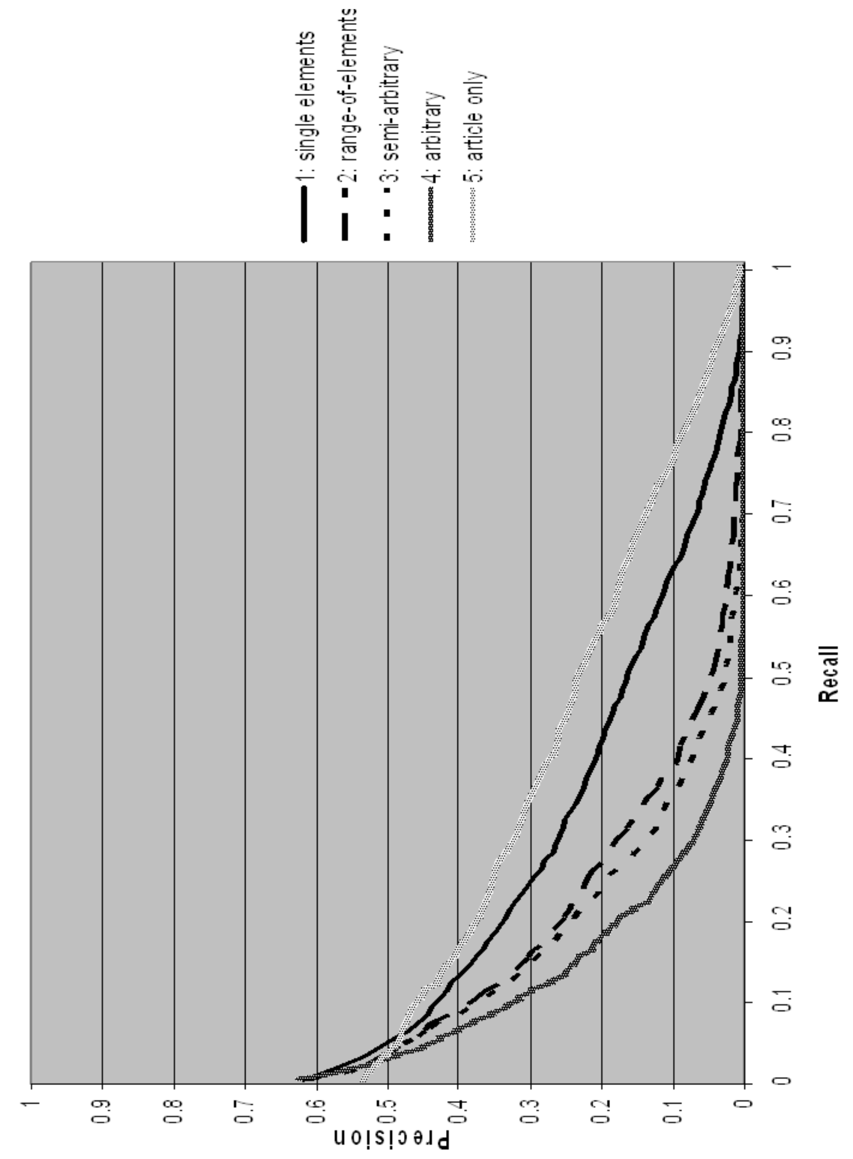


Figure 5.10: Precision Recall Curve on 2006 Test Set, Trained on iP[0.01]

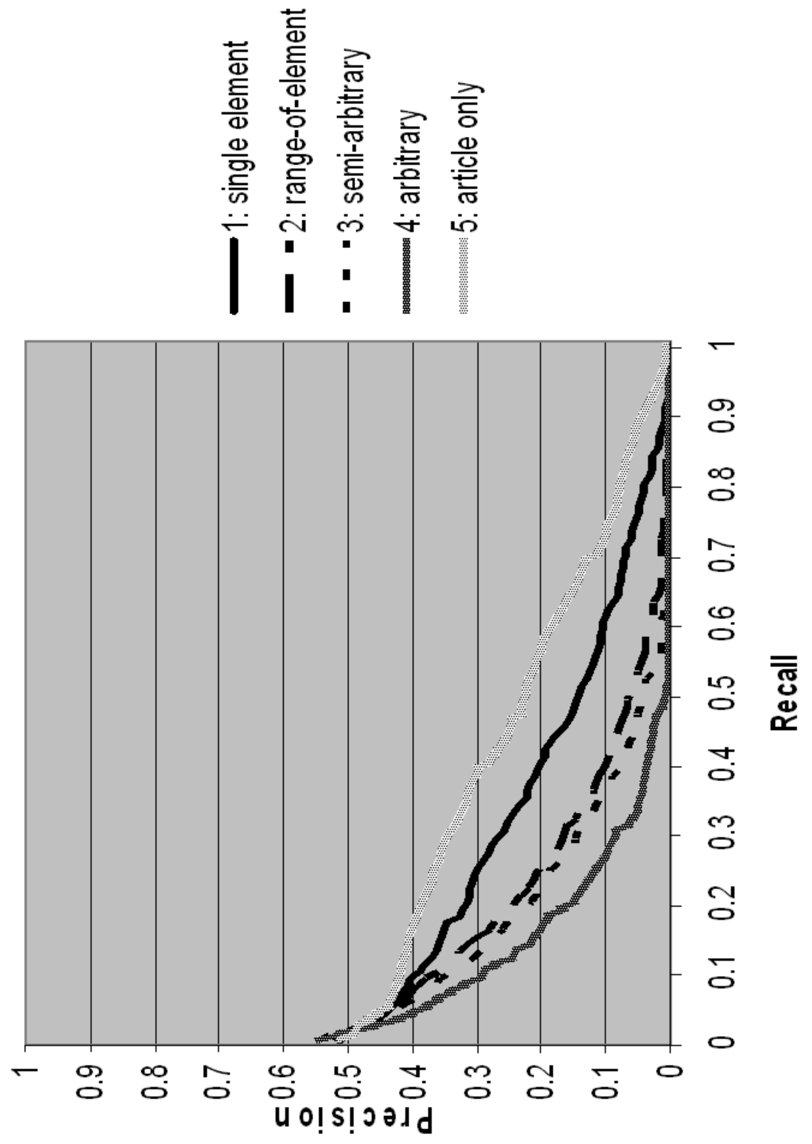


Figure 5.11: Precision Recall Curve on 2008 Test Set, Trained on iP[0.01]

Year	Average Length at 0.01 Recall	Average Highlighted Passage Length
2006	864.84	1745.92
2007	1009.32	1671.78
2008	1638.81	2347.38

Table 5.7: Number of Characters in 0.01 Recall Points and Average Number of Characters in Highlighted Passages

2007, and that of 2008 may also be a cause of misbehavior.

5.3.4 Training on MAP

With the observation from Section 5.3.3 in mind, I run the same five sets of retrieval strategies in the new setting:

- Fix average document length to prevent possible overtraining.
- Train on MAP to capture precisions across all recall points.

Table 5.8 shows the MAP scored I obtain for the five different retrieval strategies during 2007 training (reported best score achieved), and during 2006 and 2008 test sets. The results are even more surprising than when I train on iP[0.01]. Throughout the training and testing, article retrieval performs the best, followed by single-element retrieval, and this time, arbitrary passage retrieval performs the worst. It could be argued that the best unit of retrieval is an article!

Figures 5.12, 5.13, and 5.14 show the recall precision curves during the 2007 training set, and 2006/2008 test set. Indeed that the curves in each graph look quite similar to each other. Tables 5.9, 5.10, and 5.11 show the results of significant tests. It seems that during training on the

Strategy	2008 Test Score	2006 Test Score	Best Training Score	Parameter (k,b,avg)
SingleE	0.3023	0.2315	0.2177	3, 0.4, 1085.22556253005
RangeE	0.2588	0.2003	0.1828	3, 0.2, 1085.22556253005
SemiP	0.2615	0.2010	0.1868	3, 0.2, 1085.22556253005
ArbP	0.2565	0.1959	0.1801	3, 0.2, 1085.22556253005
Aritcle-only	0.3162	0.2445	0.2312	2, 0.8, 1085.22556253005

Table 5.8: Different Granularity of Retrieval, Trained on MAP

	SingleE	RangeE	SemiP	ArbP	Article
SingleE	-	***0.0116**	***0.0287**	***0.0390**	0.6378
RangeE	-	-	0.3091	0.2751	***0.0360**
SemiP	-	-	-	0.5237	0.0601
ArbP	-	-	-	-	0.0940

Table 5.9: P-values reported on paired t-tests on 2007 Training Set (*=significant), on MAP

2007 set, single element retrieval are significantly better than all strategies but article retrieval. The article retrieval itself is only more statistically significant than the range-of-element retrieval.

5.3.5 Conclusion

In Section 5.3, I run my basic retrieval approach in Chapter 4 using five different granularity of retrieval; single elements, range-of-elements, semi-arbitrary passages, arbitrary passages, and articles. Since the same corpus is used through 2006 to 2008, and passage-level qrels are available for each year, I train on 2007 set [3], and test on 2006 and 2008 test sets.

[3]Since qrels format changed from 2006 to 2007, I did not know the existence of new qrels for 2006, instead I trained on 2007.

	SingleE	RangeE	SemiP	ArbP	Article
SingleE	-	0.4778	0.3658	0.3755	0.6713
RangeE	-	-	0.5770	0.6255	0.3757
SemiP	-	-	-	0.8919	0.2940
ArbP	-	-	-	-	0.3041

Table 5.10: P-values reported on paired t-tests on 2006 Test Set, on MAP

	SingleE	RangeE	SemiP	ArbP	Article
SingleE	-	0.5234	0.5284	0.5494	0.4629
RangeE	-	-	0.8874	0.7500	0.2376
SemiP	-	-	-	0.3590	0.2343
ArbP	-	-	-	-	0.2429

Table 5.11: P-values reported on paired t-tests on 2007 Test Set, on MAP

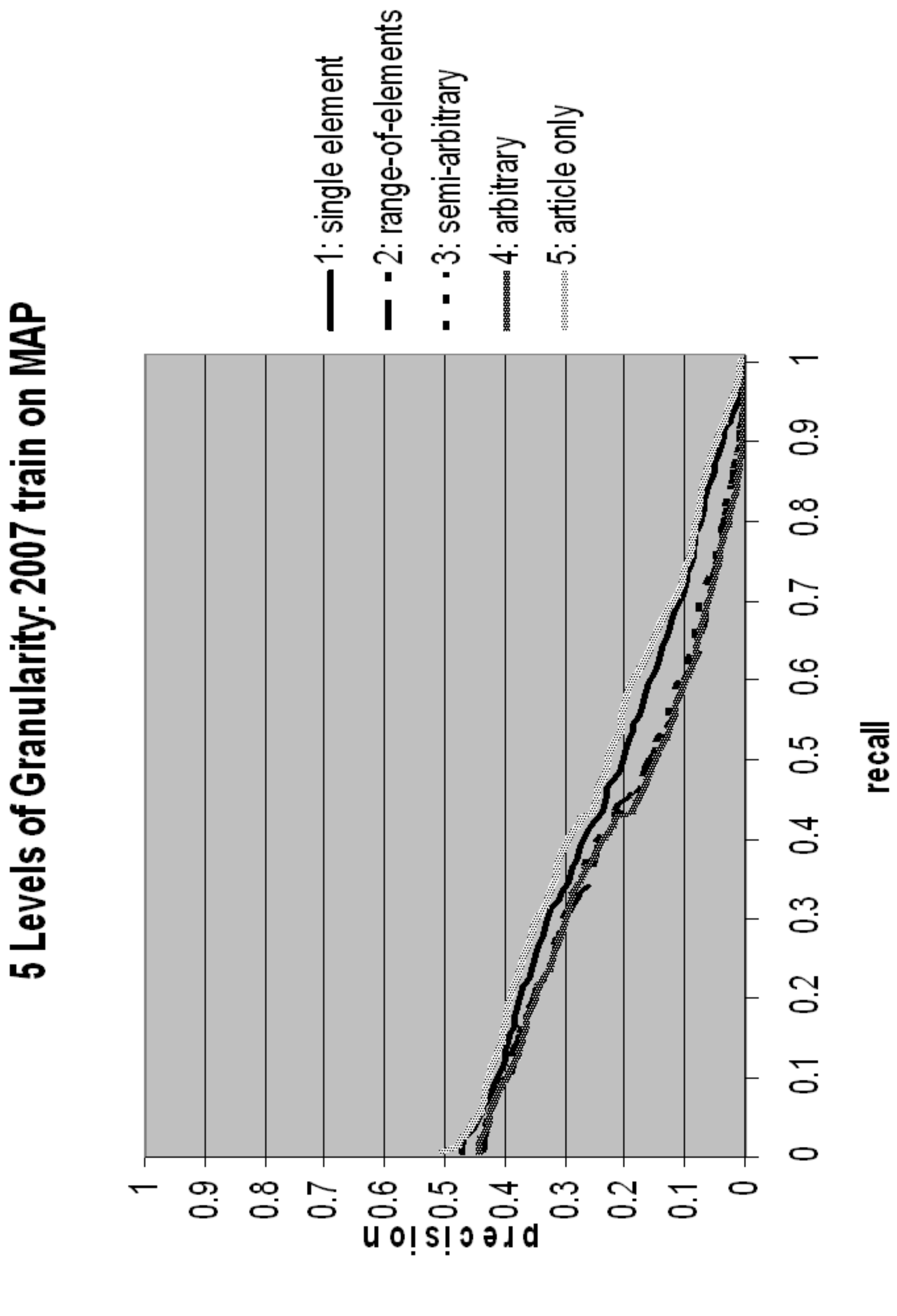


Figure 5.12: Precision Recall Curve on 2007 Training Set, Trained on MAP

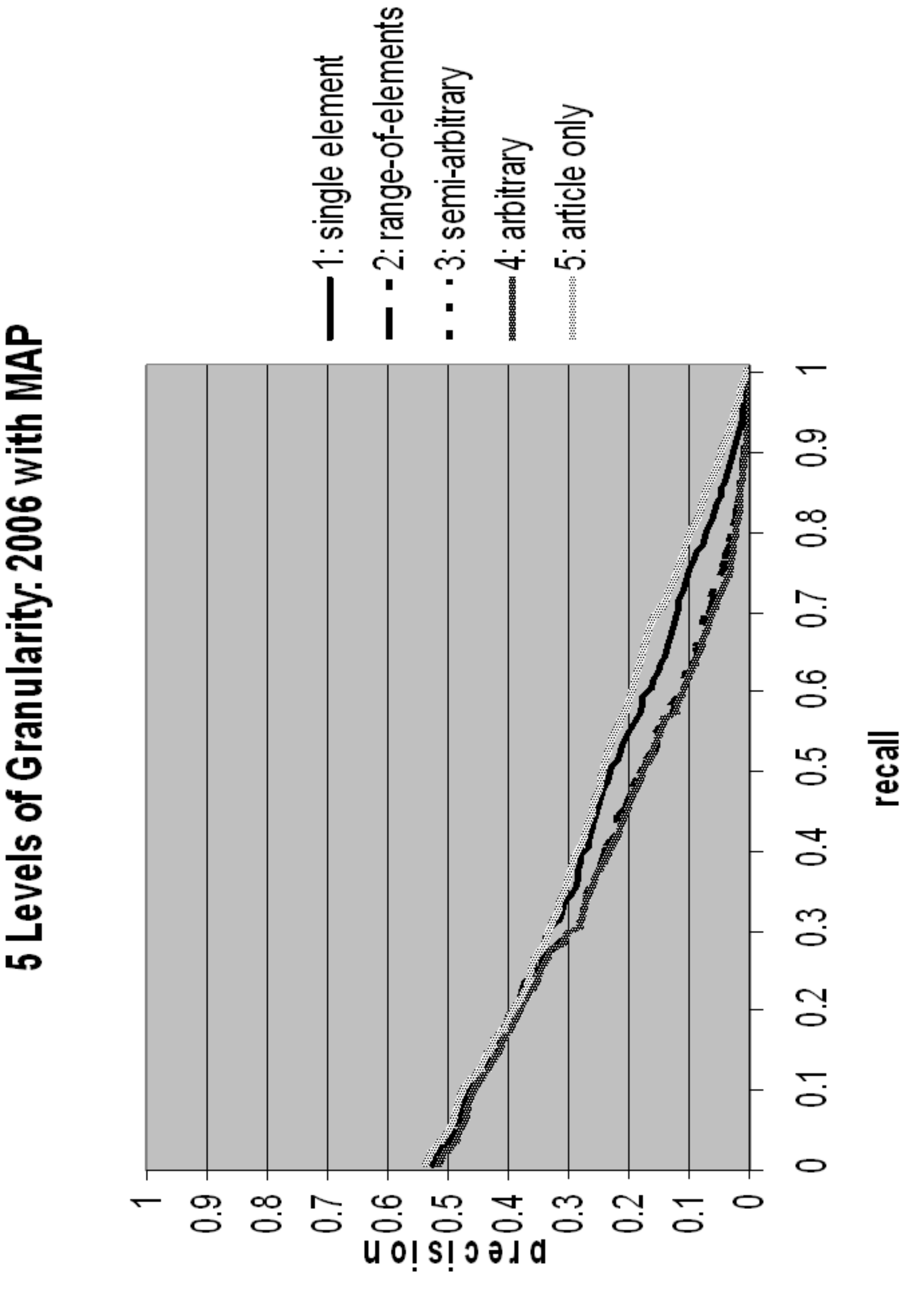


Figure 5.13: Precision Recall Curve on 2006 Test Set, Trained on MAP

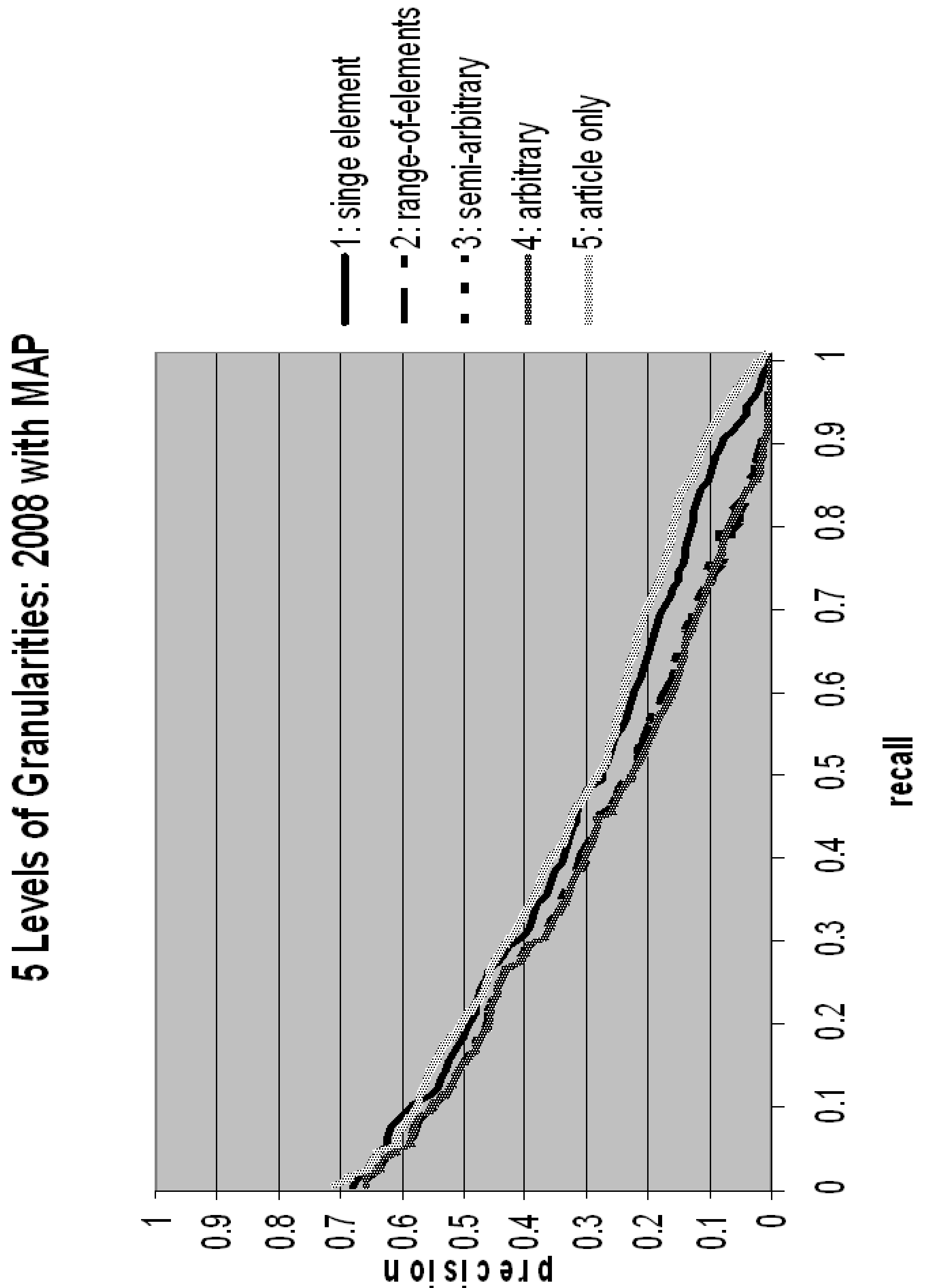


Figure 5.14: Precision Recall Curve on 2008 Test Set, Trained on MAP

The first set of results, trained on the official metric of iP[0.01] with variable average document length show that even though during training, the finer granularity performs better than the coarser one with the exception of arbitrary passages, during testing, single elements performed the best. Having average document length as a parameter may have contributed to overtraining.

The second set of results, in which I train on MAP with fixed average document length show that article retrieval performed the best. However, significant tests show that the performance of single element retrieval result is statistically more significant than others during training, except for article retrieval. During testing no approach is statistically better than any other. Since article retrieval is only more statically significant than one other retrieval strategy only during training, and that users would like a shorter document, single element retrieval may be the best retrieval approach after all. The second best run by LIP6 [23] in INEX 2009 only ranks articles. Thus the phenomenon may be system independent, but particular to the Wikipedia corpora. However, I do not have enough evidence to draw a strong conclusion.

5.4 Conclusions and Future Work

Chapter 5 investigates the issue of the passage size to return. Section 5.2 shows that turning a high scoring passage into a single element result by taking the smallest element containing the passage produces a lot of excess text compared to simply scoring all single elements. This suggests the need for an evaluation forum to directly compare single element retrieval results against arbitrary passage retrieval. Section 5.3 uses three years of test sets from INEX Ad Hoc Track Focused Task. It showed that for 6GB Wikipedia corpus [25] and for my subdocument retrieval strategy, single element retrieval performs well. Thus, together with Section 5.2, I show that single element is a sufficient (at least for the INEX tasks) unit of retrieval. Both of the results may result from the effectiveness of the pre-selection of element types and from the ineffectiveness of passage retrieval. The assessment tools may contribute to the bias towards single elements.

The future work includes studying the effect of different document structure and collection type, as well as inventing a passage retrieval specific strategy.

Chapter 6

Taking Advantage of XML Structure

Using the model of passage retrieval in Chapter 3, and the implementation details of Chapter 4, Chapter 5 comes to a conclusion that for the 6GB Wikipedia collection [25] and my subdocument retrieval strategy, single element retrieval is sufficient at least for the INEX collection. In this chapter, I adhere to the single element retrieval and study how I can take structure into account when scoring single elements.

Section 6.1 introduces the problem of structural retrieval. Section 6.2 introduces Robertson's et al.'s BM25F [21, 120], a structural version of BM25. Using BM25F, Section 6.3 proposes the idea of *characteristic field*. Section 6.4 performs experiments on INEX Ad Hoc and Book collection. Chapter 6 concludes by laying out the future map of research.

6.1 Introduction

In the INEX Ad Hoc Track focused task, while various participating groups have reported attempts to exploit XML structure in order to improve performance on this task, none of these efforts have consistently outperformed the simple approach of applying BM25 [121] to score individual XML elements and then filtering the resulting ranked list to remove overlap. Under this approach, each element is scored as if it were an independent document. The context of the element — such as information appearing in the elements that surround it — is ignored. Runs using this basic approach ranks third in 2004, third in 2007, and first in 2008 [33, 73, 96].

Robertson et al. [21, 120] describe BM25F, an extension of BM25 that exploits structural information. Under BM25F, terms contained in a document's title, for example, may be given more weight than terms contained in the document's body.

In this Chapter, I explore a framework for adapting BM25F to XML element retrieval. Under this framework, I construct two fields for each element. One field contains the contents of the element itself; the other field, called the *characteristic field*, contains contextual information associated with the element. This contextual information will vary from domain to domain, but might include the title of the overall document, titles from containing elements, and document metadata.

6.2 BM25F

6.2.1 The Original BM25F

The original BM25F proposed by Robertson et al. [120] simply assigns weights to different fields of a document. For example, a title field may have heavier weight than a section field because document titles may contain more important information than sections. A weight of each field is then used to multiply term frequencies and field length.

At INEX 2005, Robertson et al. implement BM25F for XML element retrieval, and show some improvement [92, 119]. They take the idea of anchor inheritance in BM25F web retrieval by making each element inheriting article titles. An element score is the weighted summation of the scores from three fields, one containing the body of the element, another containing the element title, and another containing all ancestral titles. However, the experiments are on a very small scale 705MB IEEE collection, and the training is performed to tune the scores on document level as opposed to element level. Furthermore, their evaluation is not on focused retrieval, so the actual effectiveness of their algorithm is unknown.

Around the same time, Trotman has studied the effectiveness of multiplying term frequencies on different parts of XML structures using a few standard document scoring function [132]. He shows that even with the optimal set of parameters, the improvement obtained by BM25F is less than 1% as computed by mean average precision.

6.2.2 BM25F with per-Field Length Normalization

One of the features of BM25 [121] in scoring a document is length normalization: The more the instances of query terms, the less significant a new instance of the query term will be. In the original BM25F [120], length normalization is applied only at a document level, whereas in reality, the weight of the first term in every section should be the same. An extended version of BM25F [21] corrects this and applies length normalization at every field. This is the version of BM25F I study in this chapter.

Using BM25F, an element's score is computed as follows:

$$BM25F(e) = \sum_{t \in q \cap e} \frac{\overline{x}_{e,t}}{K + \overline{x}_{e,t}} W_t \ \ ,$$

where q is a query term, $\overline{x}_{e,t}$ is a weighted normalized term frequency, K is a tunable parameter, and W_t is document-level IDF for a term t. To obtain $\overline{x}_{e,t}$, length normalization is first performed separately for each field f associated with an element e, producing field-specific normalized term frequencies. These normalized term frequencies $\overline{x}_{e,f,t}$ are multiplied by field-specific weights W_f

and summed to obtain $\overline{x}_{e,t}$

$$\overline{x}_{e,f,t} = \frac{x_{e,f,t}}{1 + B_f(\frac{l_{e,f}}{l_f} - 1)} \ , \overline{x}_{e,t} = \sum_f W_f \cdot \overline{x}_{e,f,t} \ ,$$

where $x_{e,f,t}$ is the term frequency of t in field f of element e, B_f is a tunable parameter, $l_{e,f}$ is the length of f in e, and l_f is the average field length of f. I report the results obtained by treating average document and field lengths as a constant, but later experiments that treated them as parameters seem to give no advantage.

6.2.3 BM25F in Book Search

In addition to web retrieval and XML retrieval, BM25F is also studied in the setting of book retrieval. Wu, Kazai, and Taylor implement BM25F with per-field length normalization on retrieval of books on INEX 2007 Book Track data set [140]. They use six fields, each corresponding to a different book part such as back of the book content and book title. Using BM25F, there is 9.09% improvement over the baseline BM25 on metric NDCG@1, but as rank goes down, the improvement decreases, 4.68% at NDCG@5, and 3.92% at NDCG@10.

Using the same INEX 2007 Book Track data set, Kazai and Milic-Frayling implement a book retrieval system incorporating external features such as the number of copies sold on Amazon.com as well as book content [80]. As part of scoring book content, they use BM25F with ten fields, some from the books and others from the MARC record that have information such as publisher name. Since they use BM25F as a baseline to compare the performance improvement from external features, I cannot compare BM25 against BM25F in this setting.

Both of the above papers use BM25F only for retrieval of books, as opposed to book pages, thus it is not focused retrieval. Furthermore, they do not perform inheritance of book titles to section as done by Lu et al. [92] and Robertson et al. [119].

6.3 The Characteristic Field

In this section, I establish a framework for retrieval of XML elements using BM25F. This framework applies to not only documents such as Wikipedia articles, but also to books and web pages. That is, using this framework, I wish to retrieve parts of Wikipedia articles, parts of Books, and parts of web pages more effectively than simply scoring all XML elements using BM25:

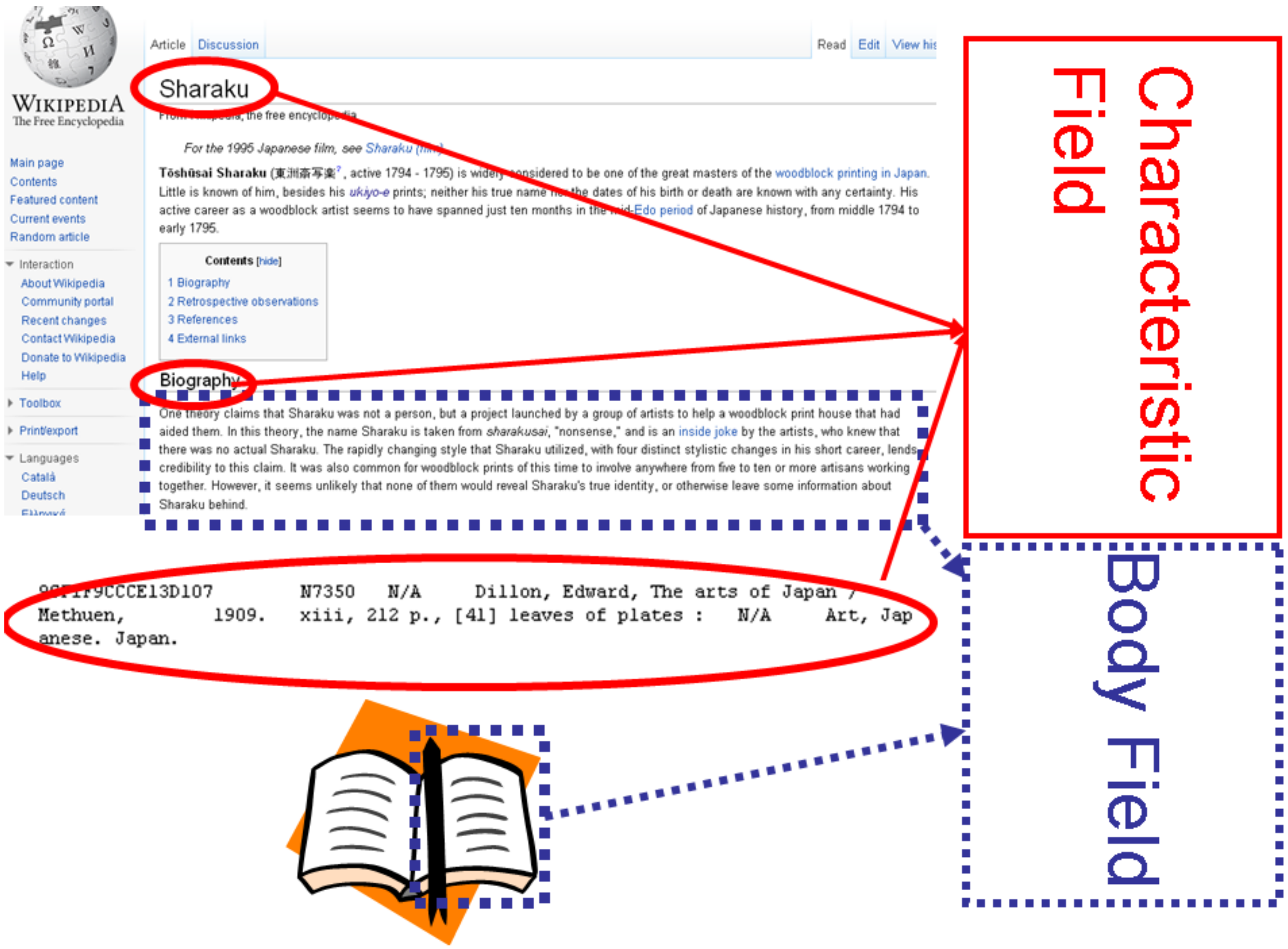
1. Decide on which XML elements are worth retrieving. For example, a paragraph element is probably more worthy of retrieval than a table column element. This step is standard in XML element retrieval.

2. Create two fields for each element listed previously. One of the field contains the entire body of an element, and the other field, I call the *characteristic field*, containing any meta data that characterizes the element.

3. Score all elements using BM25F with per-field length normalization. Since our interest in improving the performance of focused element retrieval, I remove overlapping elements.

In the case of Robertson et al. and Lu et al. [92, 119], they have two characteristic fields, one containing the current element title, and the other containing all the ancestral titles. I propose that this is an unnecessary step and concatenating all ancestral titles with the current title to fill in the single characteristic field is sufficient. For example, in Figure 6.1 [1], the title of the document is “Sharaku”, and the title of the current section is “Biography”, thus both titles go into the characteristic field. In the case of book page retrieval, in addition to book titles, I could add book categories, “ Art, Japanese. Japan.” into the characteristic field.

[1]Top: `http://en.wikipedia.org/wiki/Sharaku` snapshot taken on August 14th, 2010. Bottom: INEX 2008 Book Track MARC



Sharaku

From Wikipedia, the free encyclopedia

For the 1995 Japanese film, see Sharaku (film).

Tōshūsai Sharaku (東洲斎写楽?, active 1794 - 1795) is widely considered to be one of the great masters of the woodblock printing in Japan. Little is known of him, besides his *ukiyo-e* prints; neither his true name nor the dates of his birth or death are known with any certainty. His active career as a woodblock artist seems to have spanned just ten months in the mid-Edo period of Japanese history, from middle 1794 to early 1795.

Contents [hide]
1 Biography
2 Retrospective observations
3 References
4 External links

Biography

One theory claims that Sharaku was not a person, but a project launched by a group of artists to help a woodblock print house that had aided them. In this theory, the name Sharaku is taken from *sharakusai*, "nonsense," and is an inside joke by the artists, who knew that there was no actual Sharaku. The rapidly changing style that Sharaku utilized, with four distinct stylistic changes in his short career, lends credibility to this claim. It was also common for woodblock prints of this time to involve anywhere from five to ten or more artisans working together. However, it seems unlikely that none of them would reveal Sharaku's true identity, or otherwise leave some information about Sharaku behind.

```
9C7I79CCCE13D107        N7350   N/A     Dillon, Edward, The arts of Japan /
Methuen,        1909.   xiii, 212 p., [41] leaves of plates :   N/A     Art, Jap
anese. Japan.
```

Figure 6.1: Extracting Characteristics from Heterogeneous Source

Run	Training iP[0.01]	Test iP[0.01]	Test Rank
BM25F	0.7266 (0.7371)	0.6361 (0.6082)	1(6)
BM25	0.6818(0.6823)	0.5915 (0.5947)	15(11)

Table 6.1: Adhoc Focused Retrieval

6.4 Experiments and Results

6.4.1 Ad Hoc Retrieval

I first report the results of runs on INEX 2009 ad hoc task. I train on a 5.9GB INEX 2008 Wikipedia corpus [25] with 659,387 articles and 70 assessed topics and tested on a 50.7GB INEX 2009 Wikipedia corpus [125] with 2,666,190 articles and 68 assessed topics. The training optimizes the official metric of iP[0.01]. For these runs, I use a characteristic field formed from the titles of the article and the sections in which an element occurs.

Table 6.1 shows the best training results and ranks compared to INEX 2009 ranking [2]. The scores in brackets indicate the scores obtained by using average title/field length as additional parameters. The BM25F run that ranks first gives a 7.54% improvement over the BM25 run that ranks 15th. Because the training and the test corpora are different, the training corpus containing only section headings while the test corpus containing subsection headings, I only use up to section heading in the test corpus to create a run. The test result would likely improve if I considered subsection headings.

[2]The original ranks are 1st for BM25F and 12th for BM25, but here I report results of the runs after I remove some comments from the corpus.

6.4.2 Book Page Retrieval

I use INEX 2008 Book Track data [77] of 50239 books of size 37GB after pre-processing. Only 25 out of 70 topics has relevance judgements, thus I use 17 of them for training, and 8 for testing. The corpus comes with a file, in machine readable cataloging (MARC) format [3], that contains document ID, library of congress classification [4], ISBN, author, title, publisher, year, total pages, library of congress control number, and keywords. Among these, LC classification code and keywords seem to describe the content of the books well. The ISBN and LCCN, on the other hand do not pertain to the content of the books.

Table 6.4.2 shows MARC format for one book in the corpus. The LCC code for the book *9CF1F9CCCE13D107* is "N7350", which is categorized as

1. **N:** Visual Arts

The category of the book according to MARC is "Art, Japanese. Japan.". The title of the book is

> The arts of Japan

Our running topic title is "Impressionist and Japanese art", thus it seems reasonable to assume that adding LCC and category into the characteristic field along with the title would help increase the score for the pages from the book because of query term overlaps.

For simplicity purpose, I only use up to the second level of LCC classification. In the example above, it is "N". Adding all level of LCC classification into the characteristic field may help in the retrieval. When concatenating a title with keywords and LCC, the order of appearance in the characteristic field is LCC, categories, and title. Of 50305 available MARC records in the corpus, 24800 of them (49.30%) have LCC code, and 21497 (42.73%) have categories listed.

[3]`http://www.loc.gov/marc/`

[4]`http://www.loc.gov/catdir/cpso/lcco/`

The Book Track task requires to group the pages by the books and rank the books. Thus all of my runs does so and rank the books by the highest scoring page returned for the book. Training maximizes mean average precision.

Table 6.3 shows the results of my experiments. The runs with the plus signs indicate information used in the characteristic field. The scores in brackets indicate the scores obtained by using average title/field/document length as additional parameters. Along with Table 6.1, having additional average lengths as parameters marginally improves the best scores obtained during tuning, but it adds little improvement in testing if not overtrained. The table shows that using characteristic information gives up to 48.92% and 35.45% improvement over BM25 during training and testing respectively.

INEX 2009 Book Track

In this section, I report the results of INEX 2009 Book Track.

Unlike the experiments in Section 6.4.2, I use the brand new INEX 2009 topics for testing, use the entire MARC record in the characteristic field, and retrieve characteristic fields in addition to pages. Since the characteristic field should not have been retrieved, the actual score may be slightly higher than reported. Because of the limitation in the training/test set in my book experiments, the results for the effect of BM25F with characteristic field on book page retrieval should be considered together.

Table 6.4 shows the results obtained during training and testing at INEX 2009. For training, the improvement is 62.56%.

Doc Id	**LC Classification**	**ISBN**	**Author**	**Title**
9CF1F9CCCE13D107	N7350	N/A	Dillon, Edward	The arts of Japan
Publisher	**Year**	**Total Pages**	**LC Control Number**	**Keywords**
Methuen	1909	xiii, 212 p., [41] leaves of plates	N/A	Art, Japanese. Japan.

Table 6.2: Machine Readable Catalog (MARC) Format

Run	MAP (training)	MAP (test)
BM25	0.0278 (0.0280)	0.0110 (0.0092)
BM25F+title	0.0412 (0.0429)	0.0149 (0.0116)
BM25F+title+cat	0.0413 (0.0429)	0.0139 (0.0104)
BM25F+title+cat+LCC	0.0414 (0.0429)	0.0137 (0.0104)

Table 6.3: Book Page Retrieval

Run	MAP (training)	MAP (test)
BM25	0.0219	
BM25F+MARC	0.0356	

Table 6.4: Book Page Retrieval

6.5 Conclusions

I propose a framework for applying BM25F to XML element retrieval through the addition of the single characteristic field. This characteristic field merges contextual information from multiple sources, which may include inherited titles and metadata. The proposal is inspired by previous work, but aims to avoid the complexity of multiple fields and heterogenous structure by merging contextual information into this single field. Though it may be possible to have multiple fields and learn the type of information each field can contain from heterogeneous sources, the single characteristic field provides the baseline for heterogeneous retrieval.

The proposal is evaluated in the context of the INEX effort. While the results of the INEX 2009 Book Track have not yet been fully judged, they suggest that the benefits of field weights may be obtainable even in this simplified framework.

The proposal addresses the following questions: How do I order multiple sources of the characteristic field; should a title go before or after keywords? Is it necessary to perform length normalization within the characteristic field? Unlike the body field that contains a sequence of sentences, the characteristic field contains phrases and sentences that are in no order.

Future work includes experimenting my version of BM25F on heterogeneous collection and taking advantage of more detailed structural information available in the new INEX 2009 Wikipedia collection [125].

Chapter 7

Focused Links

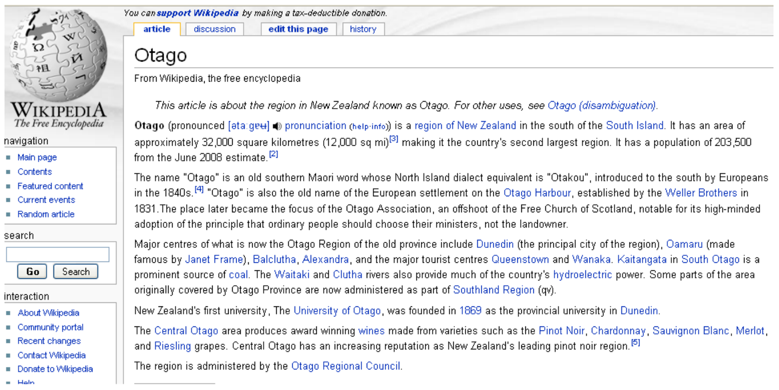
Chapters 3, 5, and 6 discuss focused search of subdocuments. This chapter discusses a special case of focused retrieval where the retrieval unit consists of a pair, an anchor phrase and a destination passage that the anchor phrase links to.

Section 7.1 introduces the problem of linking Wikipedia pages. Section 7.2 introduces related work. Section 7.3 introduces my work at INEX Link-the-Wiki Track. Section 7.5 builds on the PageRank algorithm [6] and its variant to create a new retrieval algorithm. Section 7.6 tests the effect of the new algorithm. Section 7.8 concludes Chapter 7.

7.1 Introduction

We click links in a hyperlinked document to obtain additional information. In the Wikipedia article of Figure 7.1, concerning the New Zealand region of Otago, we may click on links for Dunedin and Pinot Noir, to learn more about the capital of the region and the wines grown there.

Both "Dunedin" and "Pinot Noir" are proper nouns. As a result, editors of the article may



Otago

From Wikipedia, the free encyclopedia

This article is about the region in New Zealand known as Otago. For other uses, see Otago (disambiguation).

Otago (pronounced [ɐta:geʉ] pronunciation (help·info)) is a region of New Zealand in the south of the South Island. It has an area of approximately 32,000 square kilometres (12,000 sq mi)[3] making it the country's second largest region. It has a population of 203,500 from the June 2008 estimate.[2]

The name "Otago" is an old southern Maori word whose North Island dialect equivalent is "Otakou", introduced to the south by Europeans in the 1840s.[4] "Otago" is also the old name of the European settlement on the Otago Harbour, established by the Weller Brothers in 1831.The place later became the focus of the Otago Association, an offshoot of the Free Church of Scotland, notable for its high-minded adoption of the principle that ordinary people should choose their ministers, not the landowner.

Major centres of what is now the Otago Region of the old province include Dunedin (the principal city of the region), Oamaru (made famous by Janet Frame), Balclutha, Alexandra, and the major tourist centres Queenstown and Wanaka. Kaitangata in South Otago is a prominent source of coal. The Waitaki and Clutha rivers also provide much of the country's hydroelectric power. Some parts of the area originally covered by Otago Province are now administered as part of Southland Region (qv).

New Zealand's first university, The University of Otago, was founded in 1869 as the provincial university in Dunedin.

The Central Otago area produces award winning wines made from varieties such as the Pinot Noir, Chardonnay, Sauvignon Blanc, Merlot, and Riesling grapes. Central Otago has an increasing reputation as New Zealand's leading pinot noir region.[5]

The region is administered by the Otago Regional Council.

Figure 7.1: A sample Wikipedia page on "Otago"

be more likely to link to them than to link to the common noun "country". At the same time, Dunedin is arguably more closely related to the topic of the article than the wine, which is grown in many locations around the world.

The goal of research in this chapter is to better understand the link structure of the Wikipedia, in part as an aid for automatically suggesting links for new articles (or new links for existing articles). I derive my inspiration from the INEX Link-the-Wiki Track [57, 58, 60], whose purpose is to suggest incoming and outgoing links from a Wikipedia article stripped of links. The main focus is on suggesting *outgoing* links from an article, however, techniques used in suggesting outgoing links seem to help in identifying incoming links.

At INEX 2007, despite efforts from various participants, a simple statistical approach dominated the track. I call this approach the *Structural Threshold algorithm*, or just the *ST algorithm.* The next year, at INEX 2008, the ST algorithm was implemented by several participants, producing results that are comparable to the Wikipedia as the ground truth. However, when the runs were judged against separate manual assessments, it transpired that both the submitted runs and the Wikipedia ground truth (the links actually in the Wikipedia) differed substantially from the manual judgements. The following year in INEX 2009, the same phenomenon was observed.

To understand the difference between the Wikipedia ground truth and the manual judgements, a novel version of PageRank , which I call *KLD PageRank* or just KPR, was developed. The KPR algorithm combines the standard PageRank with a topic-oriented PageRank, using K-L divergence. Based on my experience with the INEX tracks and the KPR algorithm, I suggest that Wikipedia links may be characterized as being primarily *structural* or primarily *topical*.

Topical links are those strongly related to the topic of an article, such as “Dunedin” and “South Island” in our example. Structural links, such as “Pinot Noir” and “Merlot”, are those that have a weaker topical relationship to the article, but are frequently linked by articles in which they appear. Some links are both topical and structural to some extent: “New Zealand” is both topically related to “Otago” and also linked frequently by articles in which it appears. Current Wikipedia articles contain a fair proportion of structural links, some created automatically by bots such as Rambot [1].

In this chapter, I first describe the ST algorithm for automatically identifying links between Wikipedia articles, demonstrating that it reproduces the set of links in Wikipedia reasonably well, despite its lack of topical considerations. I follow this discussion with a summary of the INEX 2007, 2008, and 2009 Link-the-Wiki Tracks. I then compare the performance of the ST algorithm to the INEX 2008 manual assessments, highlighting the differences between them. I argue that these disparities stem from the difference in the underlying role of the links, many of which are not topical. To better demonstrate the topical nature of the manual assessments, I introduce the KPR algorithm, comparing it to the manual assessments, to the ST algorithm and to the Wikipedia ground truth. I finally deploy an inverted version of KPR algorithm to show

[1] `http://en.wikipedia.org/wiki/User:Rambot`

that it gives a good improvement over my simple baseline for identifying incoming links.

7.2 Previous Work

Wikis, such as the Wikipedia, have become important tools for gathering and sharing information. According to Kittur et al. [82], in the early days of the Wikipedia most articles were written by a small number of editors, but since then the number of editors has grown substantially. Although the Wikipedia is sometimes viewed as an unreliable source of information, Lih [87] identified 72 mainstream news sources, ranging from the Associated Press to AskMen.com [2], which cited or referred to the Wikipedia between January 2003 and March 2004. The Wikipedia itself is growing rapidly, with a projected growth of the English Wikipedia to as many as ten million articles by 2025.[3]

A substantial fraction of research related to the Wikipedia focuses on semantics, such as computing the semantic distance of concepts [41, 130]. Much of the prior work on automatic link discovery in the Wikipedia employs statistical approaches [104, 106] to compute the probability that a phrase constitutes an anchor. These approaches, as well as my own, can not only help improve the utility of the Wikipedia, but can also be applied to automatic document linking in more general contexts, such as news articles.

Similar to the prior work of Mihalcea and Csomai [104], and the work of Milne and Witten [106], the ST algorithm depends on simple statistics. Like the work of Mihalcea and Csomai, the ST algorithm ranks anchor phrases by link probability, but unlike that work, the ST algorithm considers more than just article titles as potential anchor phrases. Instead of performing word sense disambiguation separately, the ST algorithm assigns to an anchor phrase the most frequent destination in the Wikipedia corpus. Milne and Witten, on the other hand, build on

[2] `www.askmen.com`

[3] `en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth`

top of an ST-like algorithm by using the textual context of an article to disambiguate anchor phrases and destination files. In my KPR algorithm, I take topicality into account by using only the Wikipedia link graph, without the need for context parsing.

Gardner and Xiong [42] employ a sequence labeling approach in which a label indicates the start and part of the links. They use a conditional random field as a classifier and trained on both local and global data such as surrounding terms. Their results are comparable to Milne and Witten's.

All of Mihalcea and Csomai, and Milne and Witten, and Gardner and Xiong treat the links in the Wikipedia as *manually-crafted* ground truth, for comparison with their results. The ST algorithm's performance against this ground truth is better than that of Mihalcea and Csomai and somewhat less than the more complicated approach of Milne and Witten. However, I argue that the current Wikipedia links should not be strictly regarded as the one-and-only gold standard. Instead, when evaluating potential links, the topical links should be considered separately from the less-topical structural links. Otherwise, the structural links will tend to dominate the evaluation.

7.3 INEX Link-the-Wiki Track

Section 7.3 describes the background that lead to the ST algorithm that is discussed in Chapter 7.4.

7.3.1 INEX 2007 Link-the-Wiki Track

For the INEX 2007 Link-the-Wiki Track, 90 of the 659,387 documents in 6GB INEX Wikipedia corpus were nominated and selected as test *topic files* by track participants [25, 57]. These topics were stripped of all intra-article links that originally existed in the corpus. We call these stripped links *the Wikipedia ground truth.* In 2007, this ground truth represented the only method for evaluating the performance of submitted runs.

Participants then were asked to generate runs suggesting links to and from these test topic files using any information available in the stripped corpus. Use of link information related to the test topic files themselves was not permitted. Although participants were required to identify up to 250 anchor phrases per topic as well as the single best entry point (*BEP*) for each destination article, these anchor phrases and BEPs are ignored during the evaluation phase, reducing the problem to that of *file-to-file* link selection.

Four groups participated, submitting 13 runs in total. Although there were various approaches taken, the ST algorithm, my contribution, dominated the results [65]. None of the other runs used this approach.

7.3.2 INEX 2008 Link-the-Wiki Track

For 2008, the Link-the-Wiki Track introduced two tasks [58]: the File-to-File task and the Anchor-to-Best-Entry-Point task. The file-to-file task required participants to specify links at the file level, with 6600 randomly chosen topics. The Anchor-to-BEP task required participants to specify anchor phrases and up to five best entry points for a destination article for 50 topics nominated by the participants. For Anchor-to-BEP runs, the track organizers prepared manual assessments that required human assessors to judge about 600 links, an effort that was estimated to have taken 200-300 hours in total [59]. Since my goal is to compare Wikipedia ground truth against the manual assessments, my focus on the Anchor-to-BEP topics and runs. However, for the purposes of this evaluation, I treat them as if they were file level runs, ignoring the actual BEP.

Runs

The ten participating groups submitted 25 runs in total. Several other groups based their submissions on the ST algorithm, including Lycos Europe GmbH and the University of Otago. These groups in addition to my runs dominated the results. My runs [66] re-used previous year's implementation of the ST algorithm from INEX 2007. The University of Otago [71] implemented their own version of the ST algorithm, with additional weighting for capitalized phrases. Lycos Europe GmbH [28] built their algorithm on top of the ST algorithm, extending it to dynamically decide target files depending on the context of the topic, such as the set of possible links from the topic.

Other approaches that preformed reasonably well included that of the Graz University of Technology [52], who ranked destination files identified using Gazetteer by cosine similarity. The Queensland University of Technology [10] created a list of frequent phrases in the corpus, reducing the topic files into a set of frequent phrases from the list, and querying these phrases to obtain destination files. The University of Amsterdam [29] queried the corpus using the entire topic file as a query phrase to find destination files that have the same text segment as the topic file.

Evaluation and Results

The track evaluated the 2008 Anchor-to-BEP runs at the file level by ignoring the placements of anchor phrases and the best entry points, but considering all five possible destination files per anchor. The track evaluated the runs against both the Wikipedia ground truth, and against the manual assessments. For each of them, the track computed mean average precision, R-precision, and precision$n$ for various values of n. Figure 7.2 shows interpolated precision at various recall levels for the best performing runs from the top participants. The Wikipedia ground truth appears as a line at the top of the graph.

The runs based on the ST algorithm generally outperform runs based on other approaches.

However, when the track assessed the same set of runs, including the Wikipedia ground truth, against the manual assessments, the picture changes. Figure 7.3 presents the result. The performance of the Wikipedia ground truth is little better than that of the submitted runs.

7.3.3 INEX 2009 Link-the-Wiki Track

In 2009, the Link-the-Wiki Track [60] continued the same tasks as previous years, but with the new 50GB Wikipedia corpus with semantic annotations [125].

Six participants submitted runs for file-to-file level and anchor-to-BEP level tasks. Three other participants submitted their implementations of the ST algorithm. Of those, Queensland University of Technology and University of Otago occupied the highest ranks. The third participant only submitted to the file-to-file task and had the incorrect understanding of the ST algorithm. Figure 7.4 shows the results of the two participants at anchor-to-BEP level link discovery assessed against Wikipedia ground truth. Only the first destination is considered for evaluation.

In addition to my implementation of the ST algorithm, I also submitted the KPR algorithm described in Section 7.5. Figure 7.5 shows the results when assessed against manual judgements. My KPR run outperformed all other runs, including the ST algorithm and the Wikipedia as the ground truth.

7.4 Structural Threshold

7.4.1 Link Probability Estimate

The Structural Threshold (ST) algorithm is computed by creating a list of all potential anchor phrases in the corpus stripped of test topic files, and in the test topic files, by extracting phrases

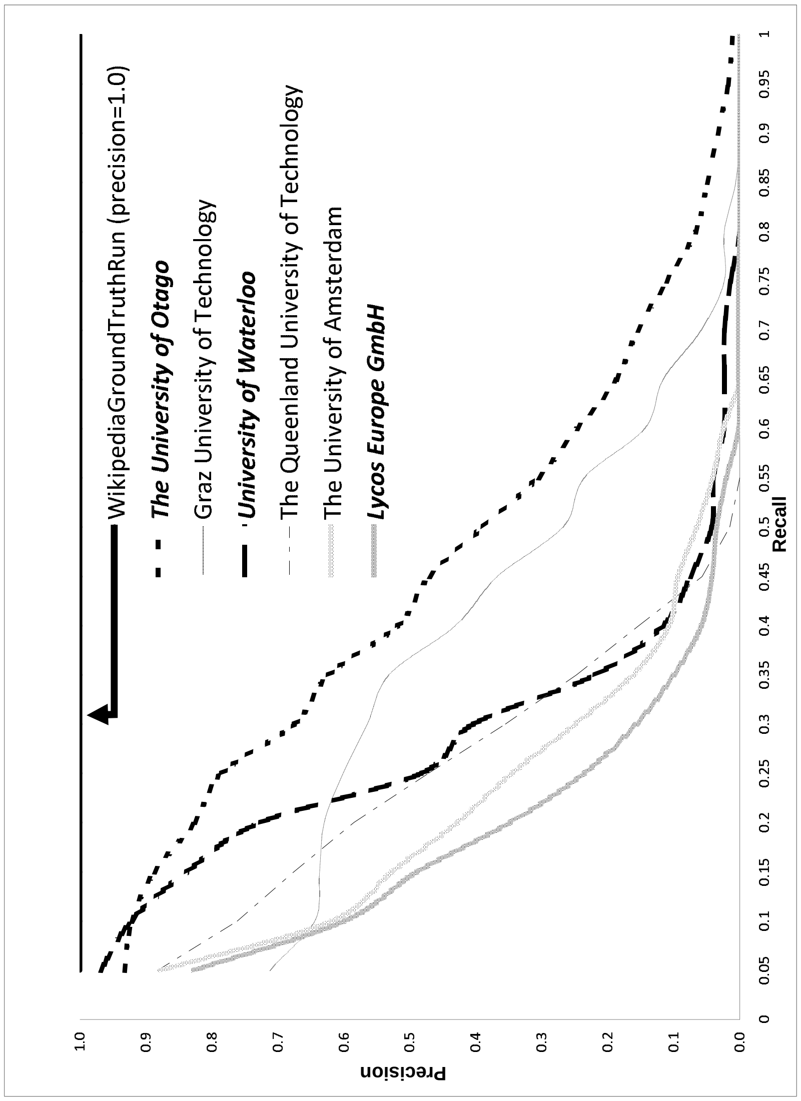


Figure 7.2: INEX 2008 Top 6 Anchor-to-BEP runs evaluated at file level against the Wikipedia ground truth (Bolded *runs* are based on the Structural Threshold) [58]

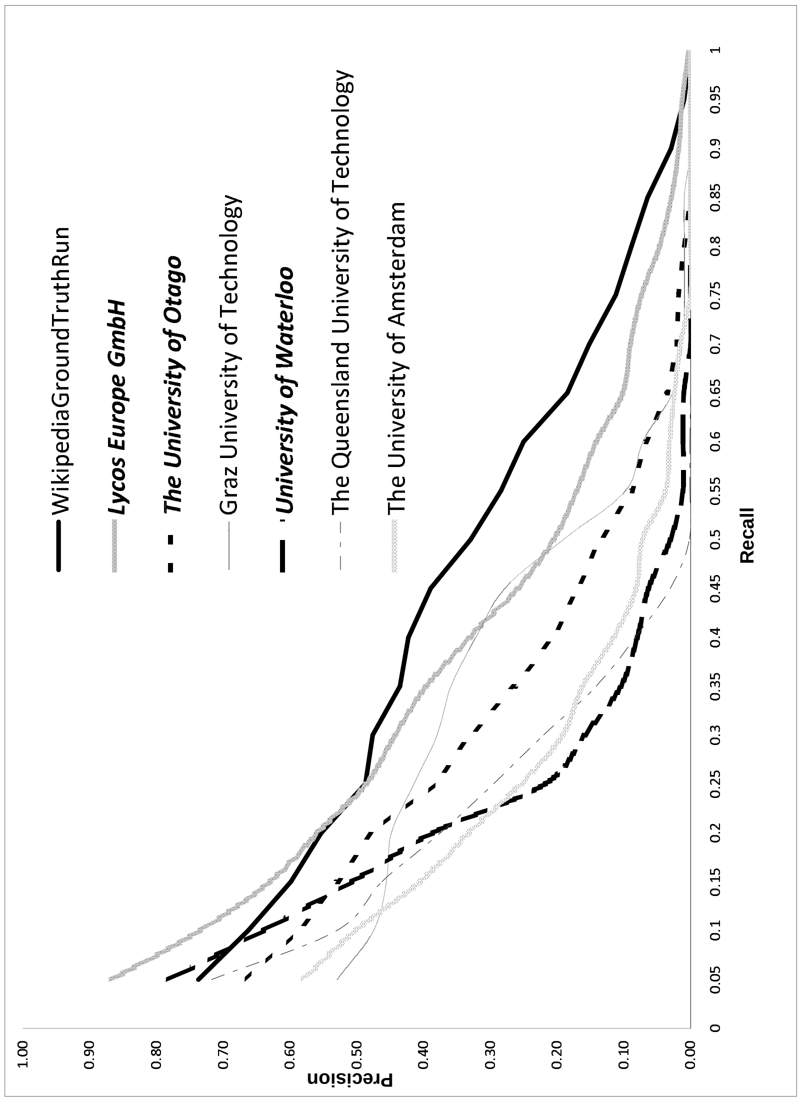


Figure 7.3: INEX 2008 Top 6 Anchor-to-BEP runs and ground truth evaluated at file level against manual assessments (Bolded *runs* are based on the Structural Threshold) [58]

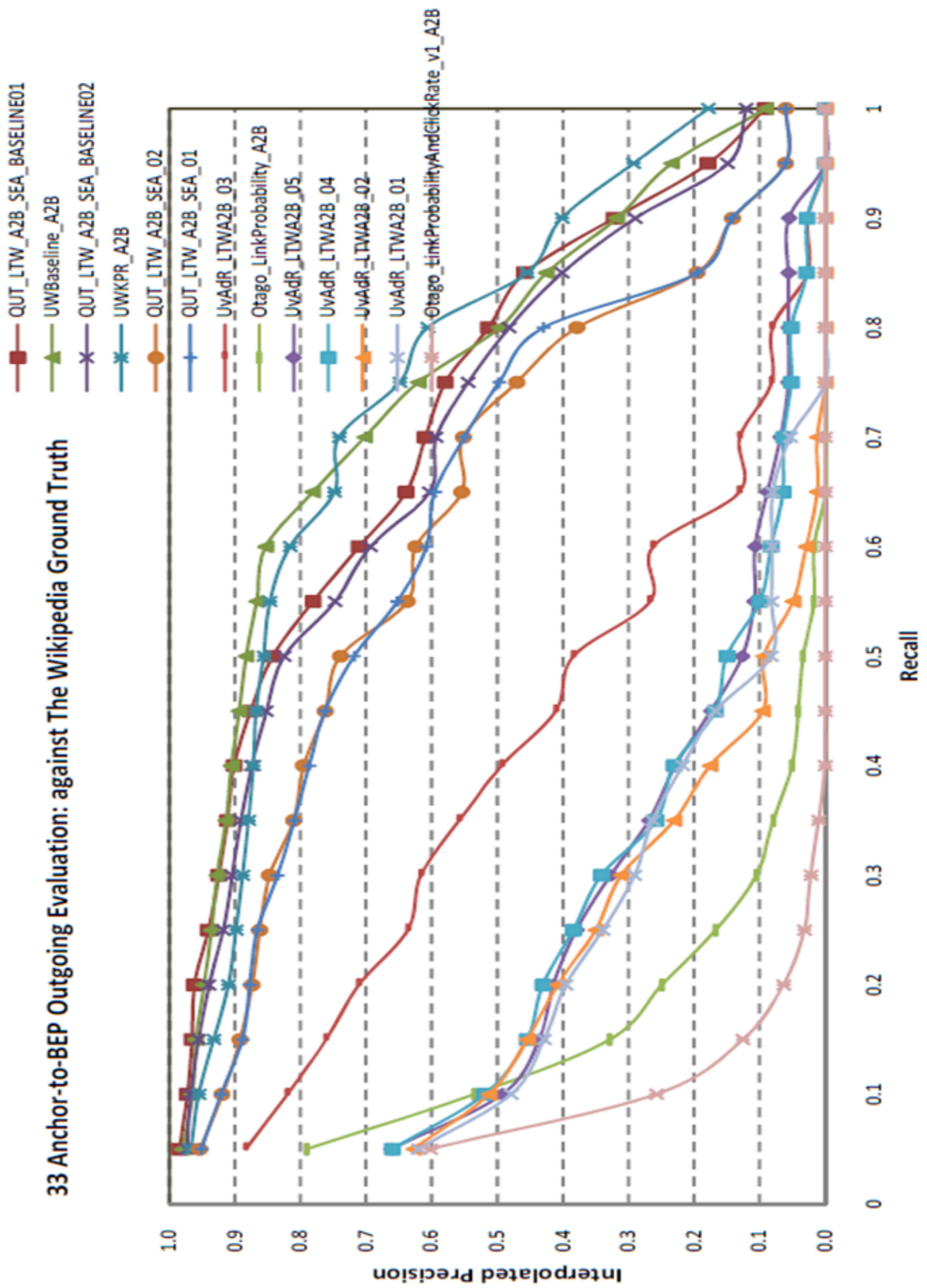


Figure 7.4: INEX 2009 Anchor-to-BEP Run Assessed Against Wikipedia as Ground Truth Reproduced from Huang et al. [60]

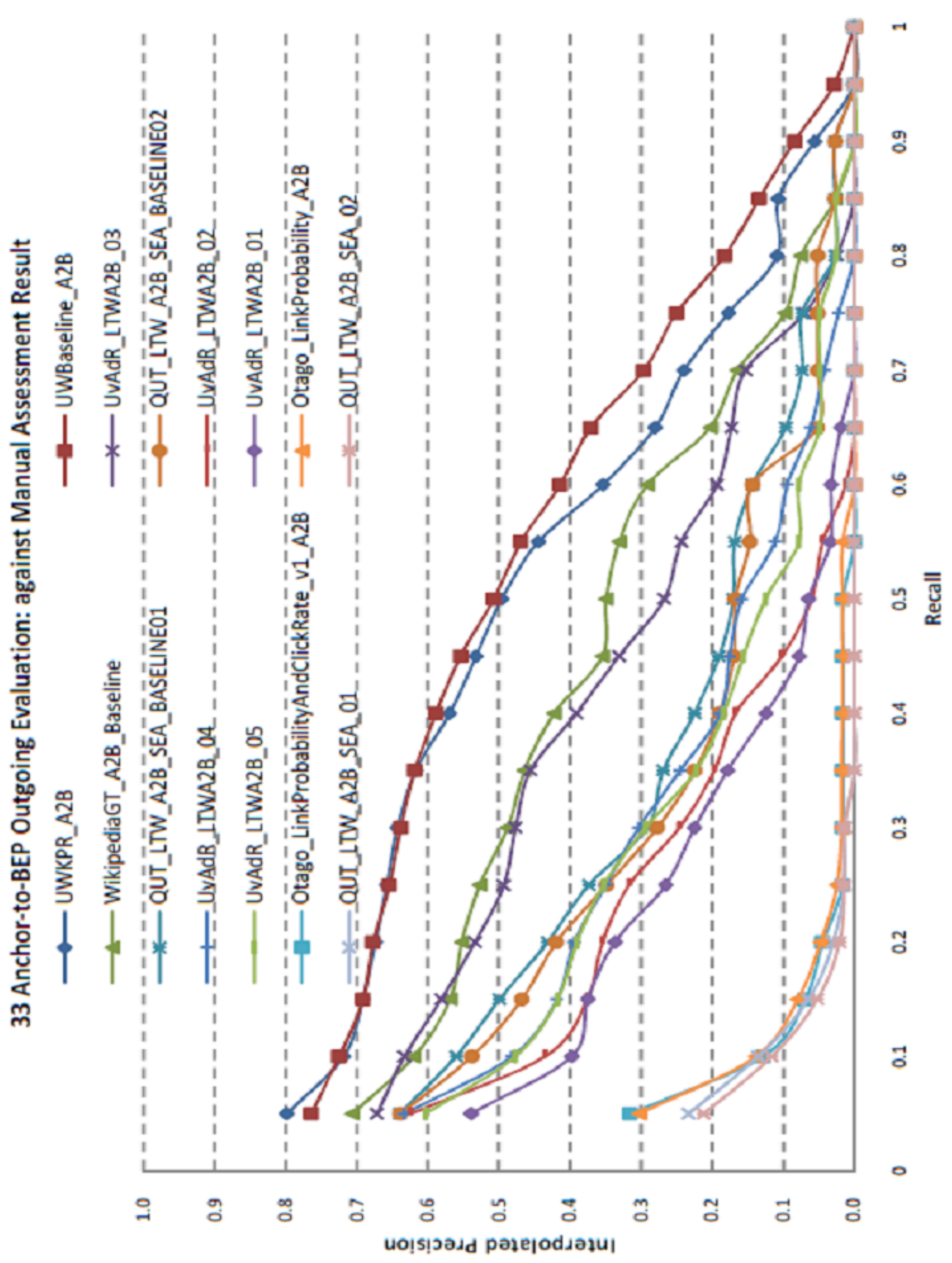


Figure 7.5: INEX 2009 Ancohr-to-BEP Run Assessed Against Manual Assessment Reproduced from Huang et al. [60]

Anchor	γ
country	0.01
province	0.07
Dunedin	0.61
New Zealand	0.68
South Island	0.79
Pinot Noir	0.81
Merlot	0.82

Table 7.1: The γ values for some anchors in Figure 7.1

verbatim (with some whitespace normalization). For each potential anchor phrase a, assign the destinations d in order of frequency. For each most frequent destination d, compute the probability estimate γ that a phrase will be linked as an anchor as follows:

$$\gamma = \frac{\sharp \text{ of pages containing a link from anchor } a \text{ to a file } d}{\sharp \text{ of pages in which } a \text{ appears at least once}} \quad .$$

For example, the phrase "New Zealand" is used to link to the articles "New Zealand","New Zealand cricket team", "New Zealand national rugby league team", "New Zealand national soccer team", and "New Zealand women's cricket team", 7680, 321, 48, 23, and 21 times respectively. Therefore, select "New Zealand" as the destination of the phrase. Since 11,312 articles in the corpus contain the phrase "New Zealand", compute $\gamma = 7680/11312 = 0.68$.

Table 7.1 lists γ values for some of the phrases in Figure 7.1. Common nouns such as "country" and "province" have low γ values, whereas proper nouns such as "New Zealand" and "Pinot Noir" have high γ values. Interestingly, when γ is plotted against rank (by frequency) the result is close to linear. Figure 7.6 shows γ vs. frequency for phrases with more than 100 links.

Links are suggested by the ST algorithm in order of decreasing γ values. Once obtaining a

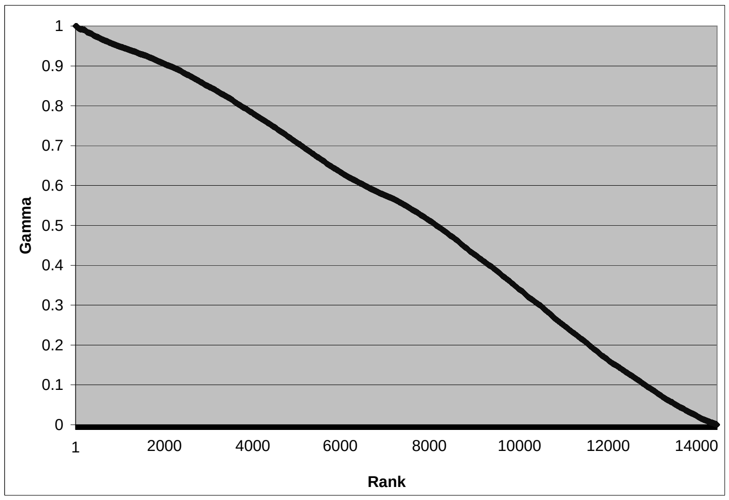


Figure 7.6: γ v.s. rank

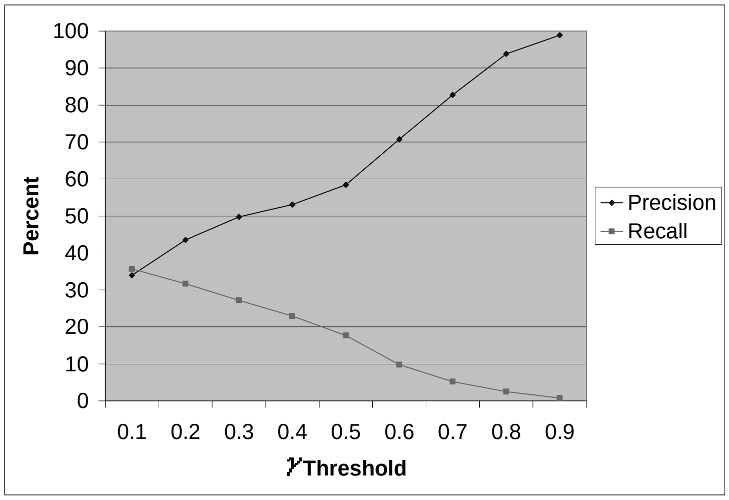


Figure 7.7: Precision and recall with varying γ threshold

list of anchor-destination pairs ordered by γ, search for the anchor phrases in the topic files and suggest the pairs in the order of γ. When there are phrases that have multiple matches in the γ list, assign the longer phrases. For example, when a topic file contains the phrase "University of Otago", link to the destination "University of Otago", rather than the destination "Otago".

The ST algorithm essentially creates a ranked list of possible links from an article. Since Wikipedia articles do not link to all possible destinations, the actual list of links to add to an article might be determined by setting a threshold for γ. Figure 7.7 shows a plot created by varying the threshold value. The usual trade-off between precision and recall is evident.

INEX 2007 provides an example of the effectiveness of the ST algorithm. Table 7.2 shows the results of University of Waterloo's INEX 2007 submission for various metrics, and the highest scores of all the other official INEX 2007 runs, none of which used the ST algorithm. Comparisons

	MAP	rPrec	P@5	P@10	P@20
ST algorithm	0.61	0.63	0.85	0.82	0.75
Others' Best Scores	0.32	0.42	0.77	0.68	0.58

Table 7.2: The Structural Threshold at INEX 2007 Link-the-Wiki

are against the Wikipedia ground truth. The official results in INEX 2008 and 2009 corroborate this; for both file-to-file level and anchor-to-BEP level runs, all the top runs, except for KPR runs, use ST algorithm as the basis [58, 60].

7.4.2 Anchor Density

Another method for selecting a threshold for adding links to an article, is to consider density of anchors in the article. Define an *anchor density* δ of a page p as follows.

$$\delta_p = \frac{\text{\# of article linked from page } p}{\text{size of page } p \text{ without Tags in KB}} \ .$$

The average anchor density of a corpus with N pages is, therefore,

$$\delta = \sum_p \frac{\delta_p}{N} \ .$$

Estimate anchor density by allocating articles into 5KB bins, according to their size, and then computing the average for each of the bins. Figure 7.8 shows that the density is roughly linear and there are $35.457/5 = 7.09$ anchors for every KB. The roughness with larger file sizes is caused by few number of files in these file sizes. Similar observations have been made by Zhang and Kamps [142]. Thus, instead of selecting links according to a γ threshold, we may add links to an article in γ order, until an anchor density of $\delta = 7.09/KB$ is reached.

I incorporated anchor density into an INEX 2008 run based on the ST algorithm by cutting the list of anchor phrases ranked by γ for each topic by δ times the file size. For example a topic

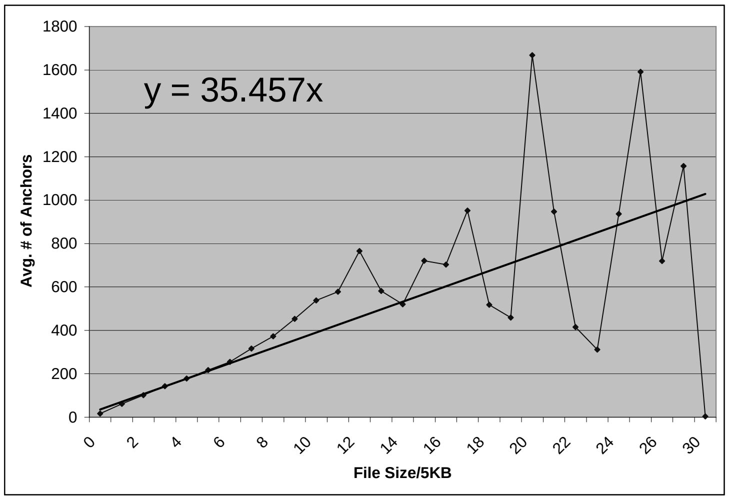


Figure 7.8: File size vs. number of anchor strings

file “Otago” has the size of 2.789KB. Therefore, I cut off the number of anchors to return for the topic to $\lceil 2.789 \times 7.09 \rceil = 20$.

I evaluate the effect of δ by computing a set precision and recall against the Wikipedia ground truth. I obtain the precision of 62.88% and recall of 65.21%. Mihalcea and Cosmai’s link identification-only accuracy is 53.37% precision and 55.90% recall [104], and Milne and Witten with more complex disambiguation technique achieves precision of 74.4% and recall of 73.8% [106]. Thus the simpler ST approach works reasonably well despite its lack of any topical considerations.

7.5 Link Analysis

The ST algorithm identifies all anchor phrases with high γ regardless of the topic of an article. Thus, an anchor phrase “Peninsular War” (a European War during Napoleonic period) with $\gamma = 0.90$ would be suggested as an anchor for nearly any page in which it occurs, from “Napoleon” to “Otago” to “wine”. According to the Wikipedia ground truth, “Peninsular War” should be indeed linked. However, the manual assessments deem the anchor phrase to be not topically relevant to the “Otago” page. From an assessor’s point of view, this European war is not related to New Zealand. On the other hand, the page “Otakou”, from where “Otago” derived its name, seems more topically relevant and manual assessments deem it relevant, but has a rather weak γ value of 0.63 and the Wikipedia ground truth judges it not relevant. Therefore, there is a need to balance the topicality of an anchor phrase with the popularity of an anchor phrase measured by γ.

In this section, I enlist a variant of PageRank to estimate the topicality of anchor phrases. Instead of measuring the popularity of anchor phrases by the γ value, through KPR algorithm, I balance both topicality and popularity by computing the contribution to K-L divergence of scores between a standard PageRank (PR) and a traditional topic-oriented PageRank (TPR).

Throughout this section, I use a process that computes KPR with all links present (links related to the test topics are not stripped). Initially, my goal is to show that even though there are both topical and structural links in Wikipedia, topical links indicated by KPR better matches manual judgements. I also present KPR as an alternative method for specifying ground truth for evaluation. For these purposes, using an unstripped version of the Wikipedia corpus is appropriate. Later in the paper, I apply KPR to solve the link discovery problem. In this case, I use stripped Wikipedia corpus, use γ instead to provide initial linkages for the test topic, and then I use KPR to compute the final linkages.

7.5.1 PageRank

The classic PageRank algorithm of Brin and Page [6] may be summarized as follows: Assume the existence of a *follow matrix* $F[1...N, 1...N]$ and a *jump vector* $J[1...N]$ An entry $F[i, j]$ in the follow matrix contains the probability that a user clicks from page i to page j. To accommodate a jump preference, an entry $J[i]$ contains the probability that a user randomly jumps to page i. The following equation then describes the PageRank algorithm.

$$r(\alpha) = \delta F \left(\sum_{\beta \to \alpha} \frac{r(\beta)}{out(\beta)} + \sum_{\gamma \in \Gamma} \frac{r(\gamma)}{N} \right) + J(1 - \delta) \quad , \tag{7.1}$$

In the equation, the variable δ is a dumping factor (I used $\delta = 0.75$), N is the total number of pages, $r(\alpha)$ is the PageRank value of a page α, $out(\beta)$ is the outdegree of a page β, and Γ is a set of sinks. In the basic version of PageRank, all elements of the jump vector have the same value $(1/N)$. The equation may be solved using fixed point iteration (i.e., the power method).

I use the INEX 2008 Wikipedia corpus [25] to create a page-level directed link graph, represented by the follow matrix. I use this graph to compute a PageRank for each node in the graph using a uniform jump vector (the total number of pages in INEX Wikipedia corpus is $N = 659387$). Table 7.3 lists the top 10 articles with the highest PageRank. To normalize these

Page	PR
United States	2671.10
2000	1459.86
United Kingdom	1336.37
2005	1304.94
France	1109.47
Canada	1059.67
2003	1023.52
Square Metre	1017.11
Population Density	906.32
Germany	839.18

Table 7.3: Top 10 Wikipedia pages by PageRank

values, converting them into probabilities, divide the values in the table by N. We see that the top entries are mostly major countries and recent years.

7.5.2 Topical PageRank

In this section, I modify the jump vector J such that a user has a much higher probability of jumping to a specific target page, than to other pages. By computing PageRank with this modified jump vector, we can determine pages closely related to the target page [112]. Specifically, modify the jump vector by assigning a probability of 50% to the target page, and distributing the remaining probability mass evenly to the rest of the pages. For example, compute topic-oriented PageRank (TPR) for the "Otago" page by modifying the jump vector such that $J[\text{"Otago"}] = 1/2$ and $J[p] = 1/2(N-1)$ for all pages $p \neq \text{"Otago"}$.

Table 7.4 lists the top 10 pages with highest TPR with respect to the topic "Otago". We see

Page Title	Relevance	TPR	PR	ρ	γ
New Zealand	1	7524	321.33	0.016	0.068
Dunedin	1	4302	1.28	0.017	0.61
South Island	1	4259	23.37	0.015	.79
1861	0	3353	10.54	0.008	0.61
Maori	0	3304	29.98	0.010	0.56
Invercargill	0	3299	3.84	0.015	0.80
Central Otago Gold Rush	1	3278	1.49	0.017	0.83
1860s	0	3248	53.90	0.009	0.58
Southland Region	1	3033	3.45	0.014	0.33
Queenstown, New Zealand	1	3007	2.12	0.014	0.78

Table 7.4: Top 10 pages with highest TPR for a topic "Otago" (P@10=0.6 for Relevance=1)

that the potentially non-topical articles may have a high TPR (but also have high PR).

7.5.3 K-L Divergence Adjustment

After computing PR and TPR, I compute point-wise KL divergence. To my knowledge, this additional step represents a novel extension to the standard link analysis toolkit.

Let $f(p)$ and $g(p)$ be the TPR and PR values for a page p respectively, normalized to be probability values. Then, the contribution to K-L divergence (which we call ρ) is,

$$\rho(p) = f(p) \log \frac{f(p)}{g(p)} \ .$$

Then compute the contribution to K-L divergence for all Wikipedia articles. For example, the PR value of "Dunedin" is $g(\text{“Dunedin”}) = 10.2779/659387$. The TPR of "Dunedin" with respect to "Otago" is $f(\text{“Dunedin”}) = 4302.2559/659387$. The contribution to K-L divergence of "Dunedin"

Destination File	Relevance	ρ	TPR	PR	γ
Dunedin	1	0.017	4302	1.28	0.61
Central Otago Gold Rush	1	0.017	3278	1.49	0.83
South Otago	1	0.017	2962	0.62	0.44
New Zealand	1	0.016	7524	321.33	0.68
Otakou	1	0.015	2688	0.52	0.63
Balclutha, New Zealand	1	0.015	279	0.77	0.55
Gabriel Read	1	0.015	2643	0.52	0.71
Invercargill	0	0.015	3299	3.84	0.80
South Island	1	0.015	4259	23.37	0.79
Queenstown, New Zealand	1	0.014	3007	2.12	0.78

Table 7.5: Top 10 results for "Otago" ranked by ρ (P@10=0.9 for Relevance=1)

with respect to "Otago" is

$$\begin{aligned}\rho(p) &= f(p)\log\frac{f(p)}{g(p)}\\ &= \frac{4302.2559}{65938}\log\frac{4302.2559}{10.2779}\\ &= 0.0171\ .\end{aligned}$$

Table 7.5 lists the top 10 results in the order of ρ for the topic "Otago". Compare this against Table 7.6 that are the top 10 results in the order of γ. A relevance of 0 or 1 is given from the manual assessments. Even though the number of results manually assessed as relevant may be similar, we see that in Table 7.5, the results are closer to the topic of "Otago" as there are more geographically related topics than in Table 7.6. Moreover, we see that many of the topically non-relevant files in Table 7.4, such as "1861" and "1860s", are eliminated.

Anchor	Relevance	γ	ρ	TPR	PR
Central Otago goldrush	1	1.00	0.017	3278	1.49
Maori Language	0	1.00	0.002	685	11.49
Janet Frame	1	0.92	0.014	2581	0.90
Arrowtown	1	0.90	0.014	2552	0.58
Otago Peninsula	1	0.86	0.014	2677	0.84
Taieri Plains	1	0.85	0.014	2571	0.68
Firth of Clyde	1	0.84	0.011	2519	4.18
Gabriel's Gully	1	0.83	0.002	406	0.51
William Cargill	1	0.83	0.014	2555	0.57
Invercargille	0	0.80	0.015	3299	3.84

Table 7.6: Top 10 results for "Otago" using Structural Threshold by γ (P@10=0.8 for Relevance=1)

7.6 Experiments

My main goal is to show that the Wikipedia contains two kinds of links, topical and structural, but that the topical links suggested by KPR better reflects manual assessments. I also then try to automatically create a “manual” assessment set using KPR. For these two purposes, the use of the Wikipedia ground truth to perform KPR is appropriate. Later in Chapter 7, when I show that KPR can produce an improvement over an existing run against the manual assessments, I use a stripped version of the Wikipedia (i.e., without the knowledge of the Wikipedia ground truth, to make a run appropriate for INEX 2008).

Specifically, the first experiment takes the links in the Wikipedia ground truth for INEX 2008 topics, and ranks them by both γ and ρ. Evaluating the top results for each ranking against the manual assessments shows that ρ is a better measure of manual relevance than γ. The second experiment creates a Wikipedia ground truth *run* and a run that applied KPR on top of it. In the third experiment, instead of using the Wikipedia ground truth run, I use University of Waterloo’s INEX 2008 run and apply KPR on top to make a new run. In both the second and the third experiments, the manual assessments show that KPR gives improvements over the existing runs. The last two experiments try to automatically create “manual” assessments by running KPR on top of various versions of Wikipedia ground truth runs. It shows that the evaluation result using one of the new assessment set follows closely to the evaluation result using the manual assessment.

As a final note, since approaches taken at INEX are ranked based, and thus we have ranked-based evaluation measures, we cannot directly compare our results to those of Mihalcea and Csomai [104], and Milne and Witten [106]. However, since our ST approach works comparably to the above works as discussed in Section 7.4.2, we compare the improvements of KPR algorithm over the ST algorithm.

	P@5	P@10	P@20
by γ	54.80	56.20	51.03
by ρ	66.40	63.40	59.93
Δ	11.60	7.20	8.90
p-value	0.0007	0.0337	0.0003

Table 7.7: Performance of ranking Wikipedia ground truth by γ and ρ against manual assessments

7.6.1 Ranking Wikipedia Ground Truth

To support my hypothesis that ρ values are a better indicator of manual relevance than γ values, I rank the Wikipedia ground truth by both γ and ρ. I then compute P@5, P@10, and P@20 for each ranking against the manual assessments. Table 7.7 shows that ranking by ρ gives a significant improvement over ranking by γ, therefore suggesting that even though Wikipedia contains both topical links with high ρ values and structural links with high γ values, the manual assessors prefer those with high ρ values.

Table 7.8 and Table 7.9 further illustrate this difference. Both tables return the top ten destination files when the Wikipedia ground truth for topic "Otago" is ranked by γ and ρ respectively. All the results, but one, in Table 7.9 contain pages that are topically relevant to "Otago". In Table 7.8, "Peninsular War" (a war in Europe during Napoleonic Wars"), "census", and "Invercargill" (the Southernmost city in New Zealand, not part of Otago) have little or no topical relationship to "Otago", yet their γ values are high. Again, ρ eliminates nearly meaningless links, such as "census" probably created by a bot and appear in numerous pages about geographical region. Overall, the relevant results in Table 7.9 seem more intuitively related to the topic than those in Table 7.8.

Relevance	γ	Topic
1	0.92	Janet Frame
1	0.90	Arrowtown
0	0.90	Peninsular War
1	0.86	Otago Peninsula
1	0.85	Taieri Plains
1	0.84	Firth of Clyde
1	0.83	Central Otago Gold Rush
1	0.83	William Cargill
0	0.82	census
0	0.80	Invercargill

Table 7.8: Ranking Wikipedia ground truth by γ for topic "Otago"

7.6.2 KPR and Wikipedia Ground Truth Run

In the following two sections, I apply the KPR algorithm on two very different runs to show that KPR can give improvement against the manual assessments over existing runs. In this section, I construct a *Wikipedia ground truth run* from the Wikipedia ground truth. This will be utilized in the last experiment to try to construct an alternative ground truth.

The Wikipedia ground truth supplied by INEX 2008 consists of all the links in the topic that appear in the Wikipedia corpus. Because the official INEX 2008 evaluation software only considers the first 50 links in the ranked result set, but the Wikipedia ground truth may contain more than 50 links unordered, the choice of how to pick the links in the Wikipedia ground truth to make an evaluatable run is important. I think links in the order of appearance is reasonable and collected the first 50 links in the order of appearance, and call this run *WGT50*. I use the

Relevance	ρ	Topic
1	0.017	Dunedin
1	0.017	Central Otago Gold Rush
1	0.017	South Otago
1	0.016	New Zealand
1	0.015	Otakou
1	0.015	Balclutha, New Zealand
1	0.015	Gabriel Read
0	0.015	Invercargill
1	0.015	South Island
1	0.014	Qeenstown, New Zealand

Table 7.9: Ranking Wikipedia ground truth by ρ for topic, "Otago"

links in WGT50 run to create a link graph, and run KPR to create a result set, call this new run *ρWGT50*. For both WGT50 and ρWGT50, I evaluate against the manual assessments. Table 7.10 shows that applying KPR on top of WGT50 gives a noticeable improvement against the manual assessments.

	MAP	rPrec	P@5	P@10	P@20	P@30
WGT50	0.46	0.44	0.60	0.58	0.52	0.45
ρWGT50	0.54	0.45	0.67	0.63	0.56	0.53

Table 7.10: Manual assessment for WGT50 and ρWGT50

7.6.3 KPR and Waterloo's Run

The ρWGT50 run uses the knowledge of the Wikipedia ground truth to construct a link graph, before applying KPR. In the INEX setting, however, this is not permitted. In this section, I instead use University of Waterloo's run, UW to apply KPR to see if this new run, ρUW also provides improvement over the original run (which I call UW), against the manual assessments.

I first run the ST algorithm to return the top 50 links. I create a link graph using the links in the Wikipedia corpus, except those involving the test topic files, and then added to this graph the links returned by the ST algorithm. I run the KPR algorithm using this link graph, and then returned the top 50 pages for each topic in the order of ρ to create a run we call ρUW. Table 7.11 shows that KPR on top of the ST improves performance against the manual assessments, although the improvement is not substantial.

Next I run the same experiment on INEX 2009 dataset, on the new 60GB Wikipedia corpus [125] with the new ST run. This time, I assign an anchor phrase by filtering the list of pages that are in the order of ρ by the existence of the page titles in the topic page. The best entry point is set at the beginning of the topic files. Figure 7.5 shows that when assessed against manual judgements, my KPR run outperforms my ST run at early precisions. Table 7.12 shows the top 10 results returned by my ST run and my KPR run for topic "Politics of New Zealand". Although my ST run returns a decent number of topically related links, they still return unrelated entries such as Auckland. The KPR results, on the other hand, returns all topically related results that are important enough to link to.

7.6.4 Approximating Manual Assessments

My discussion so far suggests that the Wikipedia ground truth is not an ideal standard for assessing the performance of automatic link discovery. Moreover, ρ seems to be a better indicator of manual relevance than γ, and that applying KPR on top of existing runs seems to improve the

	MAP	rPrec	P@5	P@10	P@20	P@30
UW	0.41	0.36	0.52	0.49	0.42	0.38
ρUW	0.46	0.37	0.57	0.51	0.49	0.44

Table 7.11: Manual assessment for UW and ρUW

performance of the runs against the manual assessments. These observations lead to the idea of simulating a topically oriented assessment set.

Here, I view the manual assessments as the gold standard. However, manual assessments are not easily scalable to future experiments, with thousands of topics. However, if ρ is a good indicator of topical relevance and, applying KPR on top of runs create a run with higher scores against the manual assessments, perhaps I can approximate the manual assessments by applying KPR on top of various runs and use the run with the highest scores as an alternative standard?

Since for the purpose of creating an alternative standard, using the knowledge of the Wikipedia ground truth is acceptable and that in the previous experiments running KPR on top of WGT50 produces better performance against the manual assessments than on top of the run UW, I decide to make a variation of the Wikipedia ground truth run, WGT50.

Recall that in creating WGT50, I select the first 50 links in the Wikipedia ground truth in the order of appearance. I then use these top 50 links to create a link graph to run KPR to obtain the run ρWGT50. I vary X, the number of top links to select from the Wikipedia ground truth, to create runs WGTX. For each WGTX, I run KPR to produce ρWGTX. Table 7.13 shows the results of the manual assessments of ρWGTX. It seems that the value of X above 30 appears to give similar performance.

Next, I evaluate runs in Table 7.14 and assess those against ρWGTX runs in Table 7.15 as a standard to see which of the standards emulates the manual assessments best.

Rank	Destination by ST	Destination by KPR
1	Auckland	Electoral reform in New Zealand
2	Comparison of Australian and New Zealand governments	Comparison of Australian and New Zealand governments
3	Referendums in New Zealand	Electoral system of New Zealand
4	Rogernomics [a]	Fifth Labour Government of New Zealand
5	Electoral system of New Zealand	Don Brash [b]
6	Wellington	Winston Peters [c]
7	Single transferable vote	Referendums in New Zealand
8	Timeline of New Zealand's links with Antarctica	Peter Dunne[c]
9	Sian Elias [d]	New Zealand First
10	Monarchy of New Zealand	Treaty of Waitangi

Table 7.12: Top 10 Entries for INEX 2009 Outgoing Links

[a]NZ economics policy
[b]former politician
[c]politician
[d]Chief Justice

Standard	MAP	rPrec	P@5	P@10	P@20	P@30
ρWGT5	0.32	0.19	0.64	0.45	0.29	0.22
ρWGT10	0.41	0.27	0.71	0.61	0.41	0.31
ρWGT20	0.48	0.35	0.66	0.65	0.53	0.41
ρWGT30	0.51	0.39	0.67	0.62	0.57	0.47
ρWGT40	0.53	0.42	0.68	0.63	0.57	0.50
ρWGT50	0.54	0.45	0.67	0.63	0.56	0.53
ρWGTall	0.55	0.49	0.63	0.59	0.55	0.53

Table 7.13: Performance of ρWGTX standards against manual assessments

Run ID	Description	Ranking
UW	ST algorithm	γ
ρUW	KPR on top of UW run	ρ
UO	Official U. Otago run at INEX 2008	γ

Table 7.14: Runs used in Table 7.16

Standard	Description	Ranking
Manual	Manual Assessments from INEX	no
WGT	The Wikipedia Ground Truth	no
ρWGT50	62 results from KPR on WGT50 ; the first 50 links from the Wikipedia Ground Truth in the order of appearance.	ρ
ρWGT5	ρWGT50, but with 5 links	ρ
ρWGT10	ρWGT50, but with 10 links	ρ
ρWGT20	ρWGT50, but with 20 links	ρ
ρWGT30	ρWGT50, but with 30 links	ρ
ρWGT40	ρWGT50, but with 40 links	ρ
ρWGTall	ρWGT50, but with all the links	ρ

Table 7.15: Standards used in Table 7.16

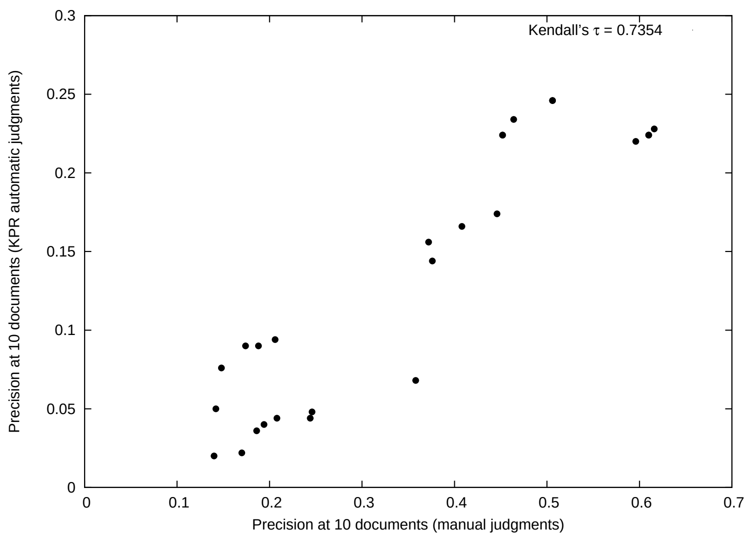


Figure 7.9: Comparison of Manual Judgments and KPR Automatic Judgments

I see in Table 7.16 that ρWGT30 best emulates the results of the manual assessments. Since ρWGT30 also performs fairly well against the manual assessments and better than all the official INEX 2008 runs, I think that the run ρWGT30 is a reasonable alternative standard in place of the manual assessments.

7.6.5 Manual Assessments v.s. KPR Assessments

Results in Section 7.6.4 may be coincidental. In this section, I reinforce the idea that KPR can be used to make an automatic manual assessment by showing correlation between manual assessment results and the KPR assessment results of INEX 2008 runs.

In order to create an automatic assessment set, I take the top 10 links in the Wikipedia ground truth by ρ as in Section 7.6.1.

Run	Standard	MAP	rPrec	P@5	P@10	P@20	P@30
UW	Manual	0.41	0.36	0.52	0.49	0.42	0.38
	WGT	0.82	0.75	0.89	0.89	0.85	0.80
	ρWGT5	0.18	0.15	0.21	0.20	0.18	0.17
	ρWGT10	0.25	0.20	0.30	0.28	0.26	0.23
	ρWGT20	0.33	0.28	0.40	0.37	0.34	0.31
	ρWGT30	0.43	0.35	0.50	0.49	0.44	0.40
	ρWGT40	0.51	0.42	0.59	0.57	0.53	0.49
	ρWGT50	0.56	0.47	0.63	0.62	0.59	0.54
	ρWGTall	0.71	0.58	0.82	0.79	0.73	0.68
ρUW	Manual	0.46	0.37	0.57	0.51	0.49	0.44
	WGT	0.78	0.72	0.83	0.83	0.79	0.77
	ρWGT5	0.17	0.16	0.20	0.19	0.17	0.16
	ρWGT10	0.25	0.22	0.33	0.27	0.24	0.23
	ρWGT20	0.34	0.30	0.44	0.38	0.32	0.30
	ρWGT30	0.42	0.37	0.53	0.45	0.41	0.38
	ρWGT40	0.50	0.45	0.59	0.54	0.49	0.47
	ρWGT50	0.55	0.50	0.64	0.60	0.54	0.52
	ρWGTall	0.75	0.60	0.83	0.84	0.78	0.74
UO	Manual	0.40	0.36	0.44	0.45	0.41	0.38
	WGT	0.82	0.76	0.90	0.87	0.84	0.80
	ρWGT5	0.17	0.15	0.20	0.20	0.18	0.17
	ρWGT10	0.24	0.21	0.27	0.25	0.25	0.23
	ρWGT20	0.34	0.29	0.41	0.38	0.35	0.32
	ρWGT30	0.42	0.37	0.49	0.45	0.43	0.40
	ρWGT40	0.50	0.44	0.58	0.53	0.51	0.48
	ρWGT50	0.55	0.49	0.60	0.58	0.56	0.54
	ρWGTall	0.70	0.61	0.81	0.77	0.71	0.67

Table 7.16: Comparison of various runs using different standards

Figure 7.9 compares precision10 values for INEX 2008 runs under both the original manual assessments and the KPR automatic assessments. I use Kendall's τ to compare the rankings under the two assessment sets. While a value of $\tau = 0.7354$ falls short of the level that would allow us to replace the manual assessments with automatic assessments [138], it suggests a close relationship between the two sets. Given the simplicity of this experiment, where the top-10 links are simply assumed to be relevant, additional development of this approach may produce a stronger correspondence.

7.7 What about Incoming Links?

Identification of incoming links has not attracted as much attention as the identification of outgoing links in Link-the-Wiki Track.

In Section 7.7, I show a simple baseline that seems to outperform other participants' run, and then apply an inverse KPR to show that KPR can also be used to improve the baseline runs in the incoming links.

7.7.1 Baseline

I score each article in the Wikipedia corpus using topic titles as query and returned the top 250 articles. In addition to file-to-file level run, anchor-to-BEP run was created by setting the anchor phrase as the page title and the BEP as the title of the topic file.

7.7.2 Inverse KPR

The idea behind using KPR for outgoing link discovery is to balance the popularity as measured by how frequently a page is link to, with topicality measured by topical PageRank. For incoming links, we may want to balance the *promiscuity*, a measure of how frequently the page has a link to other pages, with the same topicality as before.

To measure promiscuity, I invert the Wikipedia link graph with outgoing edges added from the topic files according to my ST algorithm, and apply a topical PageRank algorithm for each topic. Note that when applying KLD, I use the PageRank value based on un-inverted graph. After obtaining the list of pages in the order of τ, I screen the list by the existence of topic title in the page. The anchor phrase and the BEP are set to the page and topic titles respectively.

Fig 7.10 and Fig 7.11 show the results of assessment at file-to-file and anchor-to-BEP levels. Both assessments are against the Wikipedia ground truth. Fig 7.11 shows that the baseline performs better compared to one other participants. Together with Fig 7.10, it shows that KPR produces better results than the baseline. Table 7.17 shows an example results for the topic "Politics of New Zealand". We see that the results in the KPR run produces the links that are more general than the results from the baseline.

7.8 Conclusions and Future Work

In this chapter, as part of understanding the motivation behind user clicking links, I hypothesize that the links in the Wikipedia may be categorized into topical links and structural links. The simple statistical ST algorithm performs well as a method for predicting structural links, but manual assessors think topical links are more important, and the ST algorithm cannot predict topical links well. Because I think users are looking for topical links while still taking into account the frequency of the links in the corpus, I balance the topicality indicated by topical PageRank with the popularity indicated by PageRank through contribution to pointwise K-L divergence. Experiments show that the KPR algorithm selects destination files that are more related to the topics, more relevant according to the manual judgements, than using γ or topical PageRank. When KPR was applied on University of Waterloo's INEX 2008 run, it produces

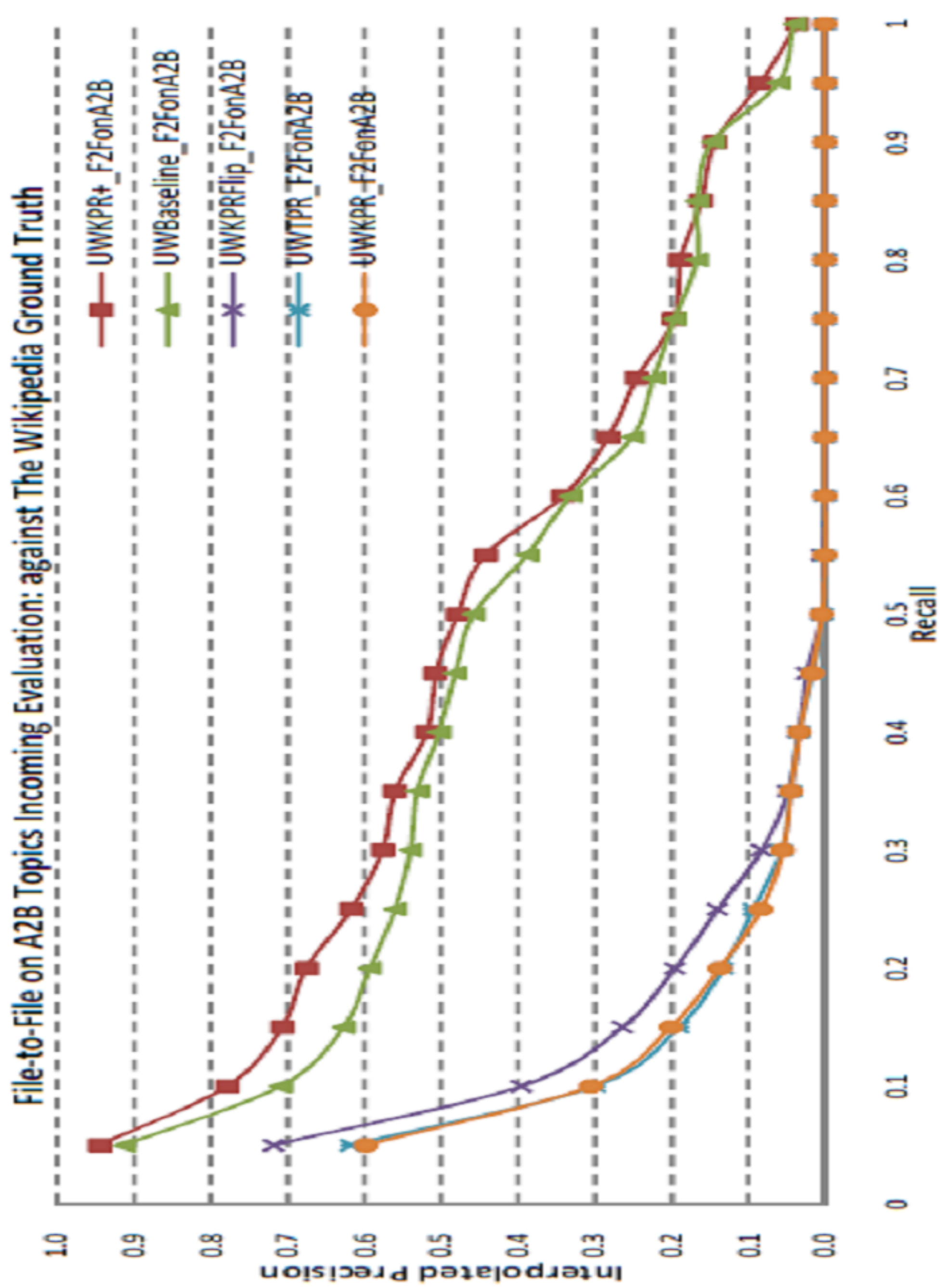


Figure 7.10: INEX 2009 File-to-File Incoming Run Assessed Against Wikipedia as Ground Truth Reproduced from Huang et al. [60]

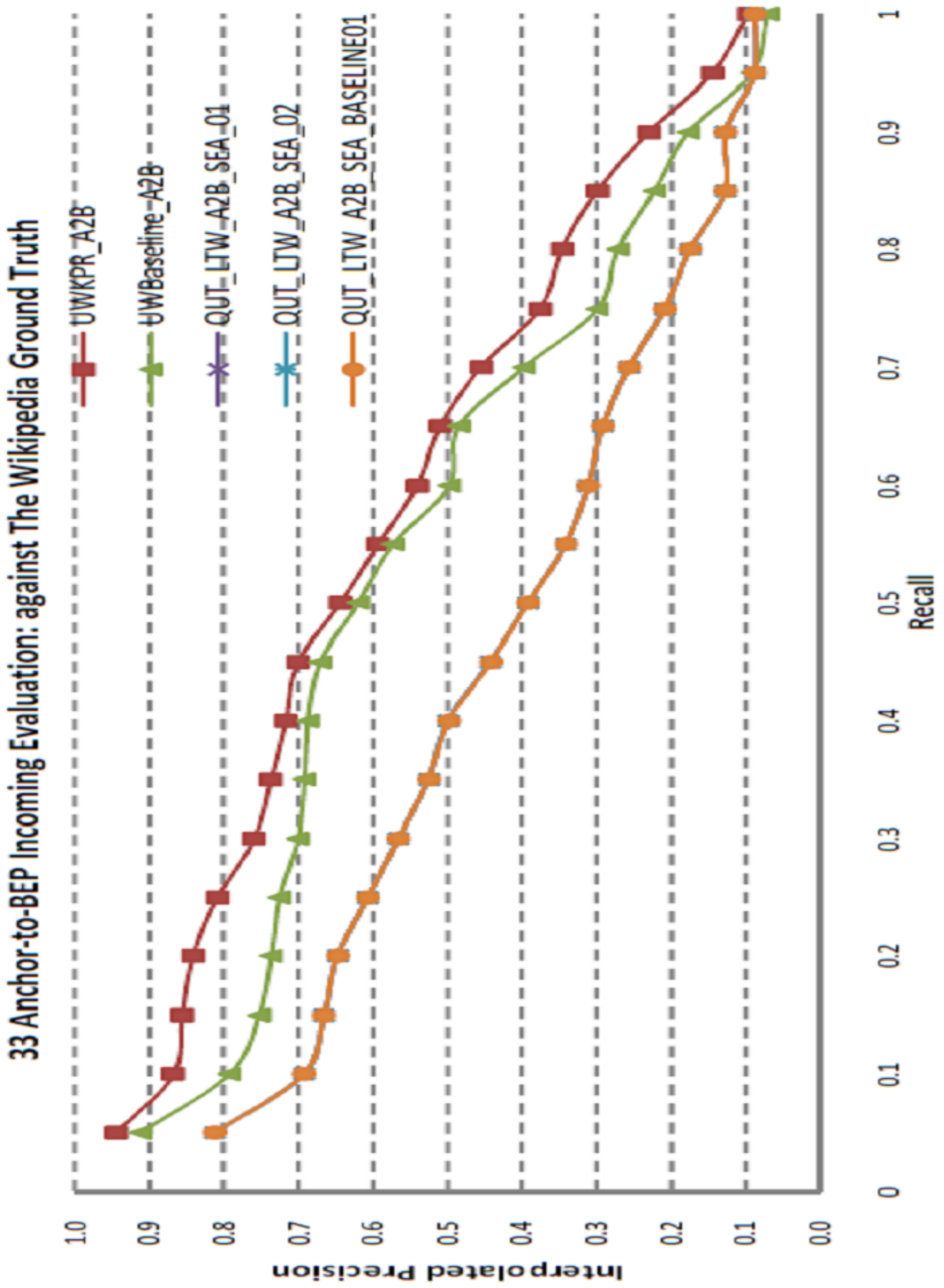


Figure 7.11: INEX 2009 Ancohr-to-BEP Incoming Run Assessed Against Wikipedia as Ground Truth Reproduced from Huang et al. [60]

Rank	Destination by Baseline	Destination by KPR
1	Portal [a]	Electoral reform in New Zealand
2	Worker's Charter [b]	New Zealand First
3	Royal Commission on Genetic Modification	Electoral system of New Zealand
4	Institute of Policy Studies (New Zealand)	Helen Clark [c]
5	OBERAC [d]	New Zealand Cabinet
6	Museum of New Zealand Te Papa Tongarewa Act 1992	Unitary authority
7	CARE (New Zealand)	Don Brash[c]
8	Bright Future (policy)	Governments of New Zealand
9	Maritime Transport Act 1994	Comparison of Australian and New Zealand governments
10	Coalition for Open Government	Rogernomics

Table 7.17: Top 10 Entries for INEX 2009 Incoming Links

[a] a portal page containing anything related to NZ
[b] a political movement in NZ
[c] politician
[d] related to cash flaw

an improvement against the manual assessments. The same phenomenon is observed in official INEX 2009 submissions.

Future work includes linking encyclopedic pages without any existing links. It would also be interesting to analyze which set of links in the Wikipedia are manually created and which links are created by bots, along with the analysis of topicality versus structurality. I could also apply our knowledge of the frequency of topical and structural links into different kinds of hyperlinked documents, that may help in detecting a spam site from normal sites in the same category of hyperlinked documents.

Chapter 8

Focused Spam Filtering

Chapters 5 and 6 discusses focused search of text to answer users' queries. Chapter 7 discusses one application of focused retrieval, focused links; how to link a portion of a document to another portion of another document. This chapter discusses another application of focused retrieval, focused spam filtering, a way to identify a portion of a document that contains vandalism.

Section 8.1 introduces the concept of vandalism in the Wikipedia. Section 8.3 discusses related work. Section 8.4 explains experiments. Section 8.5 shows the experimental results. Section 8.6 discusses the experimental results. Section 8.7 concludes Chapter 8.

8.1 Introduction

The Wikipedia [139] allows any user to contribute and edit articles. This freedom sometimes invites *vandalism*, defined by the Wikipedia as "a deliberate attempt to compromise the integrity" [1] of its content. Figure 8.1 illustrates an example of a vandalism. The topmost statement [2] is con-

[1]`http://en.wikipedia.org/wiki/Revert_vandalism`, August 15, 2010

[2]`http://en.wikipedia.org/w/index.php?title=Utamaro&oldid=89481505` August 16, 2010

sidered vandalism because of the subjective statement. The bottom statement [3], on the other hand, is not considered vandalism because it adds an objective knowledge to the topic of vegetarianism.

The problem of vandalism detection in the Wikipedia differs from other spam filtering tasks, such as email and Web spam filtering. In Figure 8.3 an entire email is spam (Personal communication.) and the entire Web page is spam [4]. Thus both emails and web spams tend to consist solely of spam content. To some extent, email and Web spam has predictable content, in this case diploma mills. Vandalism detection in the Wikipedia, on the other hand, requires us to spot parts of an article that are spam, that may consist of nothing more than a few words or characters. For example, an article may get vandalized by the addition of a nonsensical sequence of characters such as "asdfg". An identical edit may never appear again.

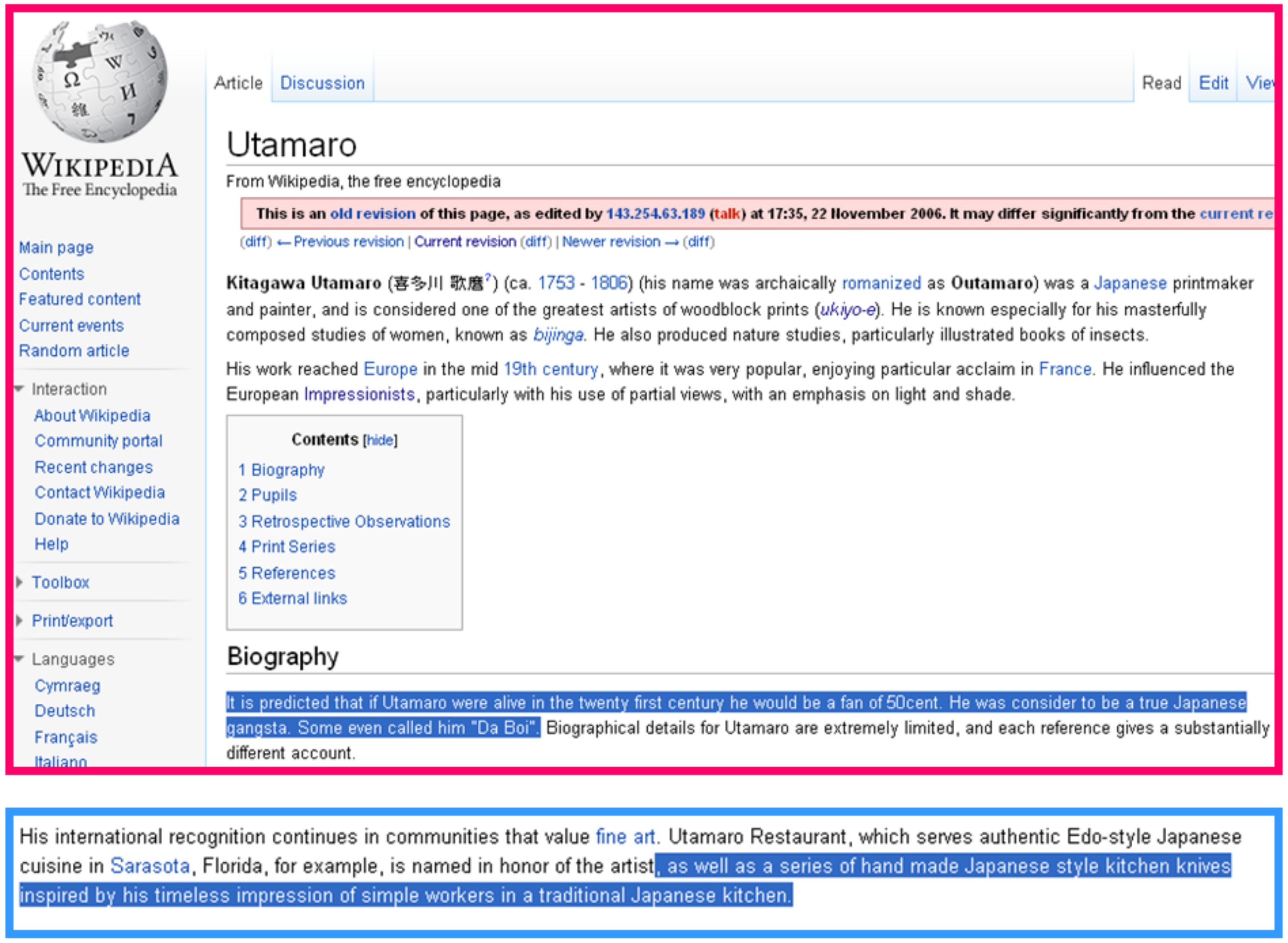
Similar models are widely employed for email and web spam filtering. My algorithm performs comparable to that of Potthast et al. [115], who use machine learning in combination with manually crafted rules to classify spam edits. My algorithm performs somewhat less than that of Chin et al. [12] who only use two articles in the study. Perhaps more surprisingly, my algorithm appears to outperform similar work, based on a different compression model, conducted by Smets et al. [129]. In that work, they classify entire revisions, which could contain many types of vandalism and concluded that a compression model alone does not perform well. To explore exactly where their compression model fails, I focus on classifying individual types of vandalism. In trying to better apply a compression model on vandalism detection, I use the same number of spam and ham edits for training. In addition, I apply Dynamic Markov Compression [20] on English Wikipedia corpus [5], instead of using prediction by partial match [17] on Simple English Wikipedia [6] corpus. In contrast to the conclusions reached by Smets et al., I show that a sim-

[3]http://en.wikipedia.org/w/index.php?title=Utamaro&oldid=312792364August16,2010

[4]http://www.pluoxy.com/q/phd-on-line

[5]http://en.wikipedia.org/wiki/Main_Page

[6]http://simple.wikipedia.org/wiki/Main_Page



Utamaro

From Wikipedia, the free encyclopedia

This is an old revision of this page, as edited by 143.254.63.189 (talk) at 17:35, 22 November 2006. It may differ significantly from the current re

(diff) ← Previous revision | Current revision (diff) | Newer revision → (diff)

Kitagawa Utamaro (喜多川 歌麿) (ca. 1753 - 1806) (his name was archaically romanized as **Outamaro**) was a Japanese printmaker and painter, and is considered one of the greatest artists of woodblock prints (*ukiyo-e*). He is known especially for his masterfully composed studies of women, known as *bijinga*. He also produced nature studies, particularly illustrated books of insects.

His work reached Europe in the mid 19th century, where it was very popular, enjoying particular acclaim in France. He influenced the European Impressionists, particularly with his use of partial views, with an emphasis on light and shade.

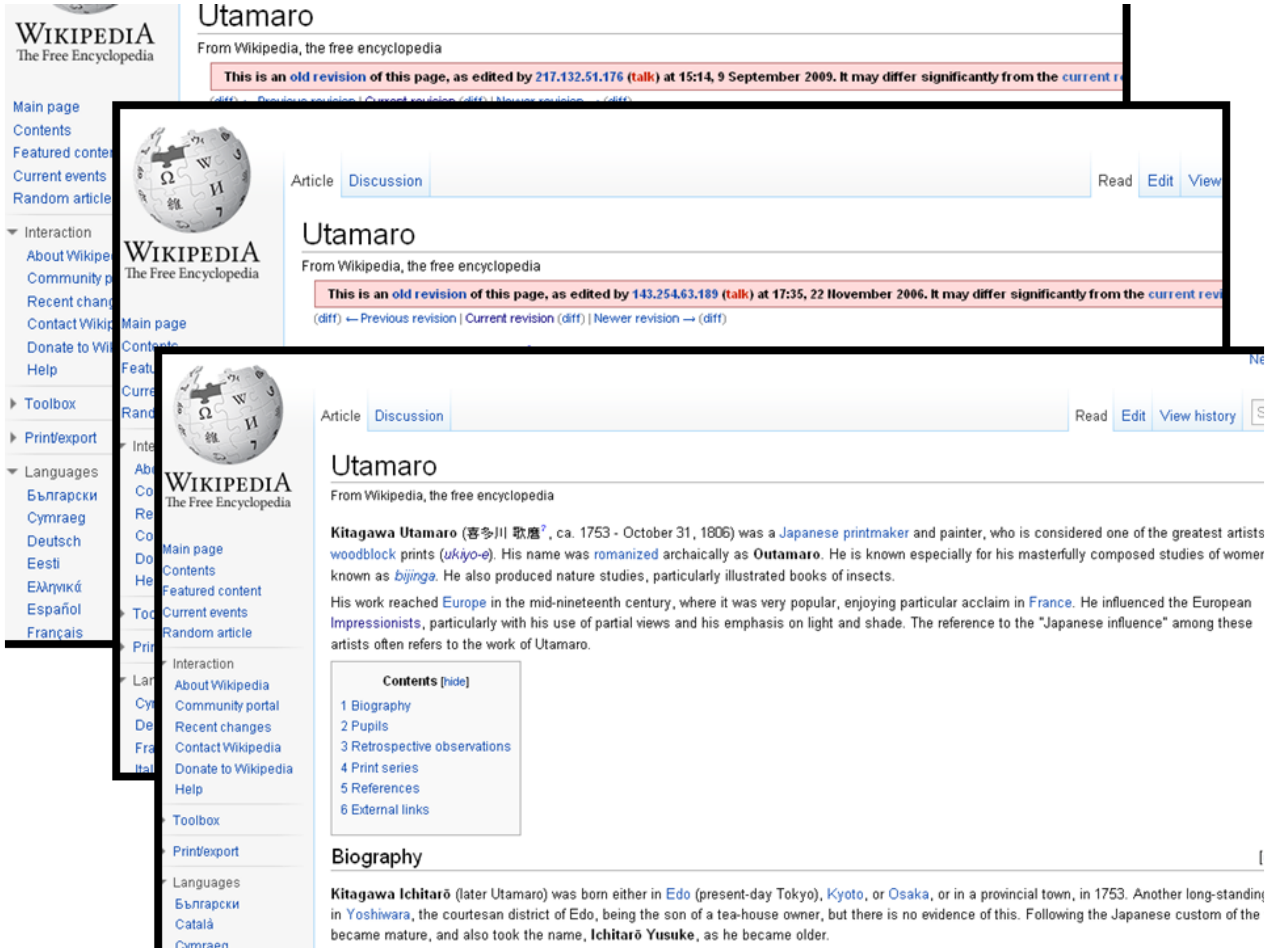
Contents [hide]
1 Biography
2 Pupils
3 Retrospective Observations
4 Print Series
5 References
6 External links

Biography

It is predicted that if Utamaro were alive in the twenty first century he would be a fan of 50cent. He was consider to be a true Japanese gangsta. Some even called him "Da Boi". Biographical details for Utamaro are extremely limited, and each reference gives a substantially different account.

His international recognition continues in communities that value fine art. Utamaro Restaurant, which serves authentic Edo-style Japanese cuisine in Sarasota, Florida, for example, is named in honor of the artist, as well as a series of hand made Japanese style kitchen knives inspired by his timeless impression of simple workers in a traditional Japanese kitchen.

Figure 8.1: Example of Spam and Ham Edit



Utamaro

From Wikipedia, the free encyclopedia

This is an old revision of this page, as edited by 217.132.51.176 (talk) at 15:14, 9 September 2009. It may differ significantly from the current r

Article Discussion Read Edit View

Utamaro

From Wikipedia, the free encyclopedia

This is an old revision of this page, as edited by 143.254.63.189 (talk) at 17:35, 22 November 2006. It may differ significantly from the current revi

(diff) ← Previous revision | Current revision (diff) | Newer revision → (diff)

Article Discussion Read Edit View history

Utamaro

From Wikipedia, the free encyclopedia

Kitagawa Utamaro (喜多川 歌麿?, ca. 1753 - October 31, 1806) was a Japanese printmaker and painter, who is considered one of the greatest artists woodblock prints (*ukiyo-e*). His name was romanized archaically as **Outamaro**. He is known especially for his masterfully composed studies of womer known as *bijinga*. He also produced nature studies, particularly illustrated books of insects.

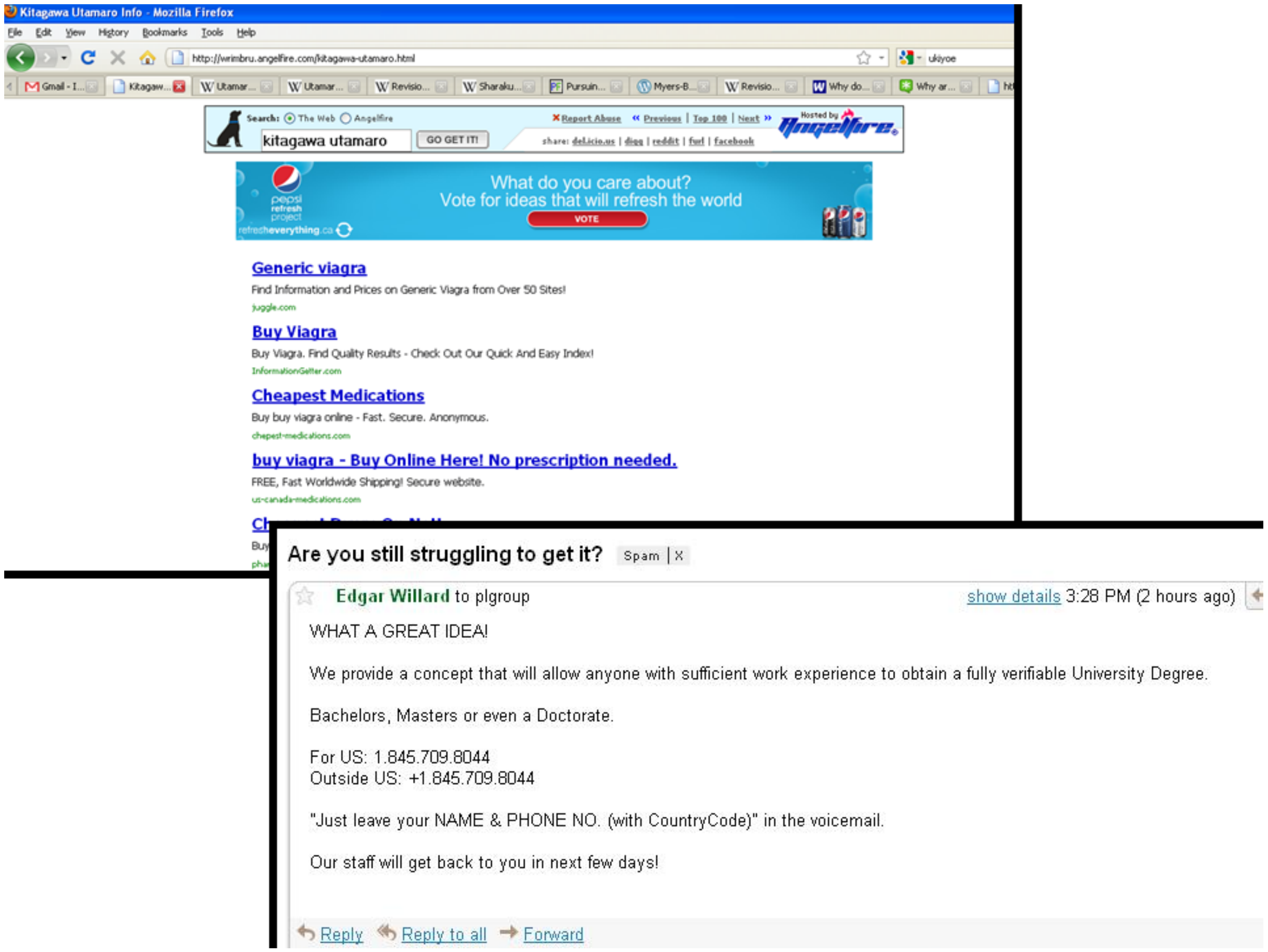
His work reached Europe in the mid-nineteenth century, where it was very popular, enjoying particular acclaim in France. He influenced the European Impressionists, particularly with his use of partial views and his emphasis on light and shade. The reference to the "Japanese influence" among these artists often refers to the work of Utamaro.

Contents [hide]

Biography

Kitagawa Ichitarō (later Utamaro) was born either in Edo (present-day Tokyo), Kyoto, or Osaka, or in a provincial town, in 1753. Another long-standin in Yoshiwara, the courtesan district of Edo, being the son of a tea-house owner, but there is no evidence of this. Following the Japanese custom of the became mature, and also took the name, **Ichitarō Yusuke**, as he became older.

Figure 8.2: A Page and its Revisions


Kitagawa Utamaro Info - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://wrimbru.angelfire.com/kitagawa-utamaro.html
Search: The Web Angelfire
kitagawa utamaro
GO GET IT!
Report Abuse « Previous | Top 100 | Next »
share: del.icio.us | digg | reddit | furl | facebook
What do you care about?
Vote for ideas that will refresh the world
VOTE
refresheverything.ca

Generic viagra
Find Information and Prices on Generic Viagra from Over 50 Sites!
juggle.com

Buy Viagra
Buy Viagra. Find Quality Results - Check Out Our Quick And Easy Index!
InformationGetter.com

Cheapest Medications
Buy buy viagra online - Fast. Secure. Anonymous.
chepest-medications.com

buy viagra - Buy Online Here! No prescription needed.
FREE, Fast Worldwide Shipping! Secure website.
us-canada-medications.com

Are you still struggling to get it? Spam | X

Edgar Willard to plgroup — show details 3:28 PM (2 hours ago)

WHAT A GREAT IDEA!

We provide a concept that will allow anyone with sufficient work experience to obtain a fully verifiable University Degree.

Bachelors, Masters or even a Doctorate.

For US: 1.845.709.8044
Outside US: +1.845.709.8044

"Just leave your NAME & PHONE NO. (with CountryCode)" in the voicemail.

Our staff will get back to you in next few days!

Reply Reply to all Forward

Figure 8.3: Web and Email Spams

ple compression-based algorithm works reasonably well, and could provide the foundation for an operational spam filter.

8.2 Problem Definition

In order to frame the problem of Wikipedia spam detection, I first define the following terminology:

- **A Page** of Wikipedia is an entity that pertains to human knowledge. In Figure 8.2 [7], the entity is a concept of an Ukiyoe painter named "Utamaro".
- **A Revision** of a page is an instance of the page at a discrete time point. In Figure 8.2, there are three revisions of the page "Utamaro", one current, one at 5pm on November 22th, 2006 [8], and one at 3pm on September 9th, 2009 [9].
- **An Edit** is a set of textual acts that changes a revision at one time point to create a new revision. In Figure 8.1, in both top and bottom figures, an insertion of a new statement created a new revision.
- **An Insertion Edit** is an edit created by inserting new statements. Figure 8.1 shows two examples of insertion edits.
- **A Change Edit** is an edit that changes part of a sentence. For example, changing a phrase "Wikipedia vandalism detection" into "Wikipedia spam detection" is considered a change edit.
- **A Spam Edit** is a set of vandalism in a revision of a given page. In Figure 8.1, the topmost highlighted statement is a spam edit.

[7] `http://en.wikipedia.org/wiki/Utamaro` August 16, 2010

[8] `http://en.wikipedia.org/w/index.php?title=Utamaro&oldid=89481505` August 16, 2010

[9] `http://en.wikipedia.org/w/index.php?title=Utamaro&oldid=312792364` August 16, 2010

- **A Ham Edit** is an attempt to improve the content of the page. In Figure 8.1, the bottom highlighted statement is a ham edit.

Finally, I define the problem of spam detection in Wikipedia as follows.

Problem 1 (Wikipedia Vandalism Detection) *Given a set of unseen edits, decide whether the edit is vandalism or not.*

8.3 Previous work

Dynamic Markov Compression (DMC) [20] is a compression algorithm that compresses a file by predicting the next bit based on previously seen bits. DMC has been used successfully for classifying email and Web spam [5]. For classification, separate compression models are built for ham and spam. The model that achieves the greatest compression ratio over new data indicates the class of that data.

Little formal research has been conducted on the identification of vandalism in the Wikiepdia. Potthast et al. [115] use logistic regression with 16 manually crafted features, to categorize an edit as spam or ham. Smets et al. [129] use the Prediction by Partial Match (PPM) compression model [17] to classify an entire revision of a page in the Wikipedia. Using the Simple English Wikipedia as corpus, they sum probabilities of a revised page containing various types of spam. They also include user groups and revision comments as additional features, but remove Wikipedia tags from edits. Instead of parsing the actual edits, West et al. [2] use pure metadata such as the time of the edit, and the length the editor is active to achieve performance comparable to Smets et al.'s. Chin et al. [12] use language model to measure various features such as perplexity of edits from previous edits. The features are then used to predict the classification.

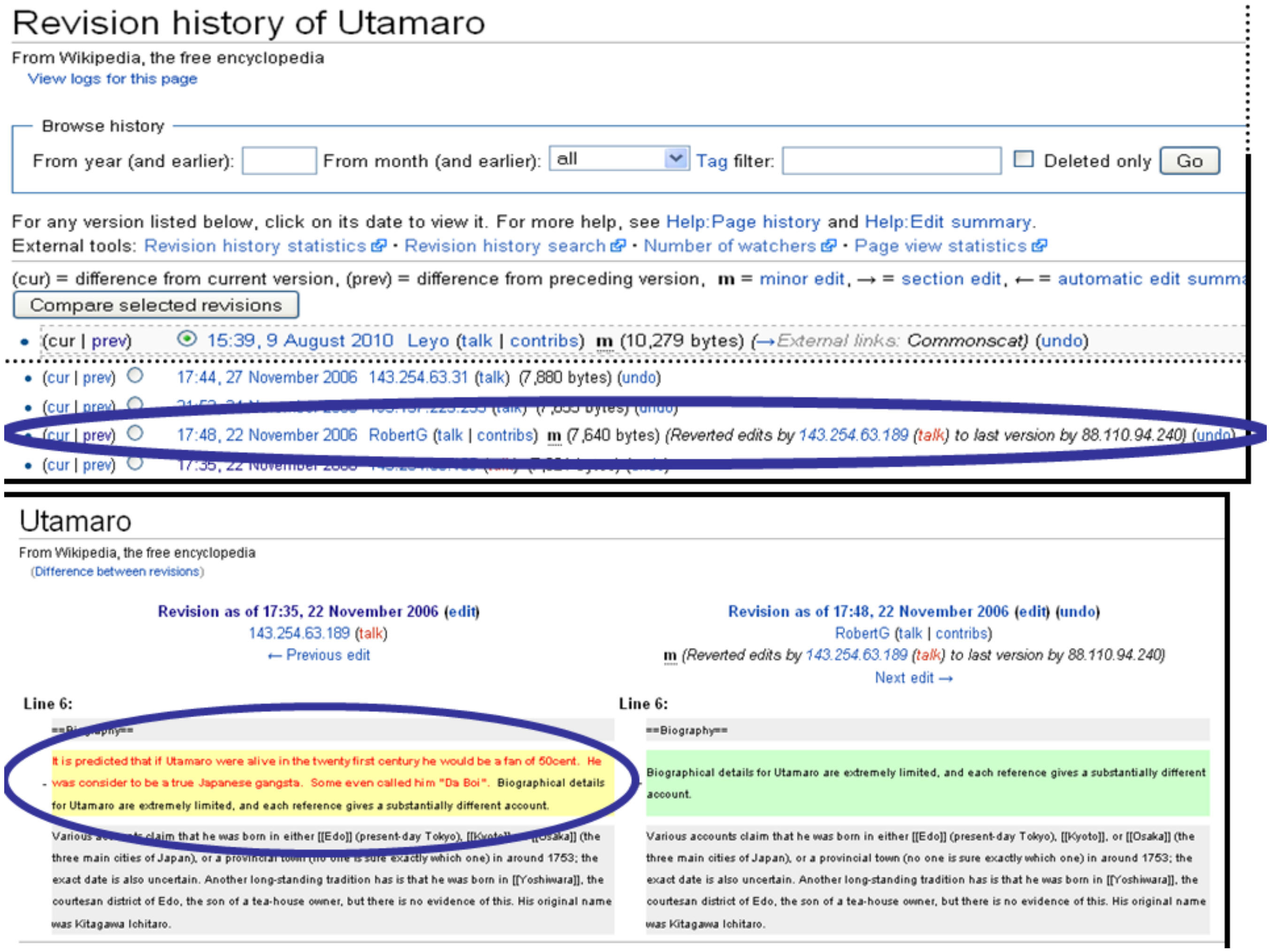
8.4 Experiments

Instead of classifying entire revision pages, which could include multiple types of spam as Smets et al. [129] did, I break down the problem into smaller steps. The goal is to determine if it is possible to identify two types of spam edits, insertion and change edits, separately. Thus I consider a revision consisting solely of insertion or change edits to create a corpus. My Wikipedia vandalism corpus is created as follows:

1. Collect random English Wikipedia pages by following "Random article" links.
2. For each page, obtain history pages.
3. In each of history page, find entries that contains the terms "revert" and "undid". In Figure 8.4 [10], the circled entry in the top-most figure has a "reverted" comment.
4. For each revert/undid entry, collect a diff page that lists the difference between the previous revision and the reverted/undid version. I retain the Wikipedia tags such as '[[link]]" for internal links.
5. A pool of spam edits are collected from edits in the previous versions that are reverted.
6. Ham edits are collected from those that spam edits are reverted into.
7. Finally make the number of spam and ham edits in the corpus even. An edit that is later reverted by a user by removal or change is treated as spam, and an edit that is reverted to by addition or change is treated as ham.

Table 8.1 illustrates the number of edits in my corpus. It is not clear if Smets et al. [129] use a balanced training corpus as Pothast et al. [115] did. Priedhorsky et al. [116] reports the rate of

[10] `http://en.wikipedia.org/w/index.php?title=Utamaro&limit=500&action=history` and `http://en.wikipedia.org/w/index.php?title=Utamaro&diff=89484099&oldid=89481505` August 16, 2010



Revision history of Utamaro

From Wikipedia, the free encyclopedia
View logs for this page

Browse history
From year (and earlier): From month (and earlier): all Tag filter: Deleted only Go

For any version listed below, click on its date to view it. For more help, see Help:Page history and Help:Edit summary.
External tools: Revision history statistics • Revision history search • Number of watchers • Page view statistics

(cur) = difference from current version, (prev) = difference from preceding version, **m** = minor edit, → = section edit, ← = automatic edit summa

Compare selected revisions

- (cur | prev) 15:39, 9 August 2010 Leyo (talk | contribs) **m** (10,279 bytes) (→*External links: Commonscat*) (undo)
- (cur | prev) 17:44, 27 November 2006 143.254.63.31 (talk) (7,880 bytes) (undo)
- (cur | prev) 21:53, 24 November 2006 [illegible] (talk) (7,833 bytes) (undo)
- (cur | prev) 17:48, 22 November 2006 RobertG (talk | contribs) **m** (7,640 bytes) (*Reverted edits by 143.254.63.189 (talk) to last version by 88.110.94.240*) (undo)
- (cur | prev) 17:35, 22 November 2006 [illegible]

Utamaro

From Wikipedia, the free encyclopedia
(Difference between revisions)

Revision as of 17:35, 22 November 2006 (edit) 143.254.63.189 (talk) ← Previous edit	Revision as of 17:48, 22 November 2006 (edit) (undo) RobertG (talk | contribs) **m** (*Reverted edits by 143.254.63.189 (talk) to last version by 88.110.94.240*) Next edit →
Line 6:	**Line 6:**
==Biography==	==Biography==
- It is predicted that if Utamaro were alive in the twenty first century he would be a fan of 50cent. He was consider to be a true Japanese gangsta. Some even called him "Da Boi". Biographical details for Utamaro are extremely limited, and each reference gives a substantially different account.	Biographical details for Utamaro are extremely limited, and each reference gives a substantially different account.
Various accounts claim that he was born in either [[Edo]] (present-day Tokyo), [[Kyoto]], or [[Osaka]] (the three main cities of Japan), or a provincial town (no one is sure exactly which one) in around 1753; the exact date is also uncertain. Another long-standing tradition has is that he was born in [[Yoshiwara]], the courtesan district of Edo, the son of a tea-house owner, but there is no evidence of this. His original name was Kitagawa Ichitaro.	Various accounts claim that he was born in either [[Edo]] (present-day Tokyo), [[Kyoto]], or [[Osaka]] (the three main cities of Japan), or a provincial town (no one is sure exactly which one) in around 1753; the exact date is also uncertain. Another long-standing tradition has is that he was born in [[Yoshiwara]], the courtesan district of Edo, the son of a tea-house owner, but there is no evidence of this. His original name was Kitagawa Ichitaro.

Figure 8.4: Creating Test Collection

Data Set	**# Spam Edits**	**# Ham Edits**
insertion training	5758	5768
insertion test	481	481
change training	5768	5768
change test	600	600

Table 8.1: Corpus Size

vandalism to be 5%. For my experiment, I collect the same number of spam and ham edits for each set. For insertion edits, I also vary the ratio of spam to ham training data to determine its effect on performance. The testing is carried out as follows:

1. Attach an unlabled test edit to both the spam and ham training texts.
2. Compute a compression ratio using a modified version of Cormack's implementation of DMC [19].
3. The test edit is judged to be spam if the compression ratio with the spam training set was higher than the compression ratio with the ham training set.

Moreover, I change the threshold x such that the spam compression ratio must be higher than x times the ham compression ratio to be classified as spam. Adjusting x allows me to compute a recall precision break-even point (BEP) for each collection.

8.5 Results

Table 8.2 shows the results of my experiments. Precision and recall values are calculated for both spam and ham. Spam precision is computed as the number of spam edits correctly identified

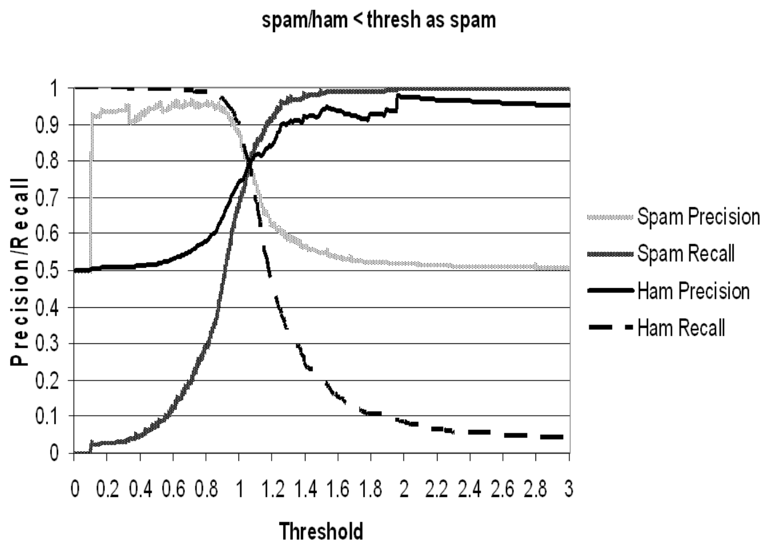


Figure 8.5: Insertion Edit: Precision-Recall Curve on Various Thresholds

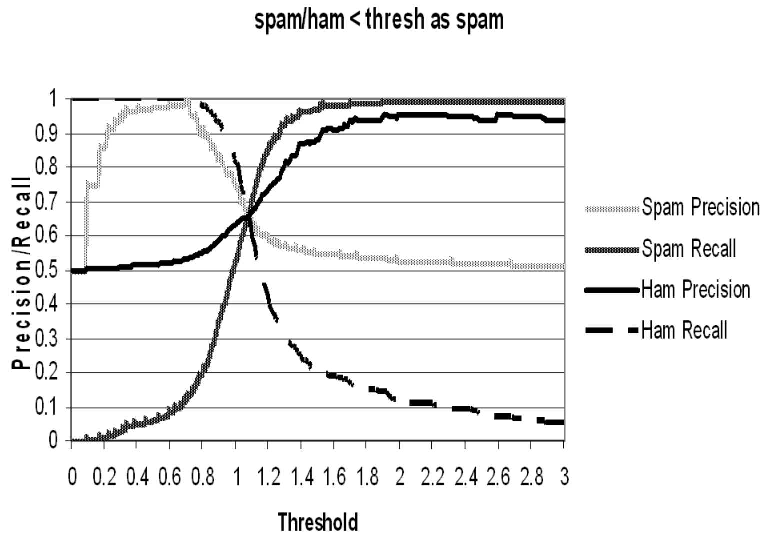


Figure 8.6: Change Edit: Precision-Recall Curve on Various Thresholds

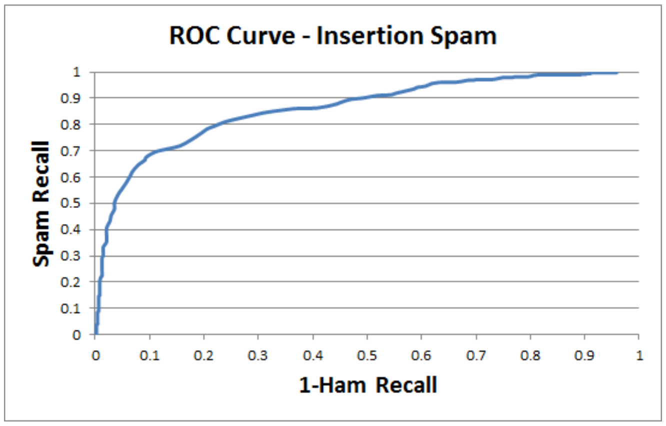


Figure 8.7: Insertion Edit: ROC Curve

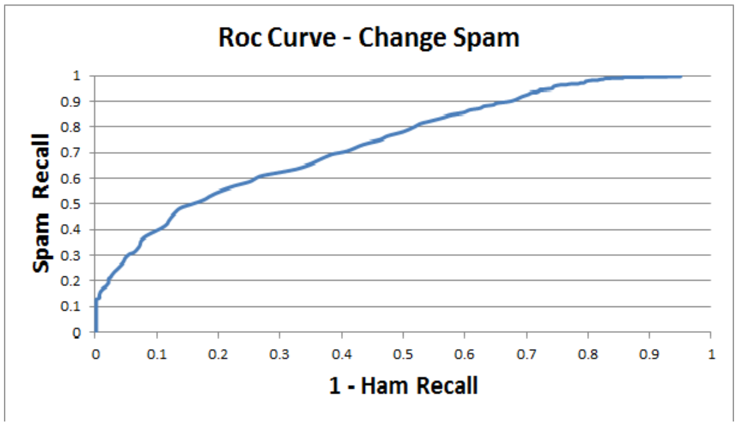


Figure 8.8: Change Edit: ROC Curve

Experiment	**Precision**	**Recall**	**Break Even Point**
spam insertion	86.56	69.65	78.26
ham insertion	74.61	89.19	78.26
spam change	74.07	53.33	65.28
ham change	63.54	81.33	65.28

Table 8.2: Precision, Recall, and Break Even Point for Spam and Ham Test Set

as spam over all edits identified as spam. Spam recall is computed as the number of spam edits correctly identified as spam over all spam edits. Ham precision and recall values are computed similarly. Figure 8.5 and Figure 8.6 show the precision-recall curves for change and insertion spam and ham edits by varying thresholds. The break even points are obtained by varying the threshold x, the spam to ham compression ratio, to categorize a test edit as spam. Figure 8.7 and Figure 8.8 show ROC curves for insertion and change edits. We see that the curve is very steep for insertion edits, while the curve for change edits is gradual. Table 8.3 shows the results of varying spam to ham edits ratio in the training set. It shows that when they are even, we obtain the best F-measure.

8.6 Discussion

The results indicate that identifying insertion spam is reasonably effective. On the other hand, change spam seems harder to identify. This may be because change edits tend to contain chunks of non-consecutive phrases that are changed and may need some surrounding context to improve effectiveness.

In practical application, at least for email spam filtering, we would like to avoid false positives; ham messages classified into a spam folder, as not so many people check their spam folders. In

Spam/Ham Ratio	**Spam Prec.**	**Spam Rec.**	**Ham Prec.**	**Ham Rec.**	**Spam F**	**Best F**
0.125	94.39	38.46	61.36	97.71	54.65	73.91
0.25	91.79	51.14	66.14	95.43	65.68	75.16
0.5	90.37	60.5	72.58	93.56	72.48	76.19
1	86.56	69.65	74.61	89.19	77.19	78.26
2	79.53	76.72	77.51	80.25	78.10	77.02
4	70.5	82.74	79.15	65.49	76.13	74.53
8	63.02	88.57	80.77	48.02	73.64	73.29

Table 8.3: Varying Spam to Ham Edit Ratio in the Training Set

this case, we may choose the threshold to be lower than 1 to have higher spam precision and ham recall. It is unclear, however, if this also applies in the case of Wikipedia vandalism detection. To keep the quality of article high, an administrator may want to classify all edits suspected of vandalism as spam. In this case, we may choose our threshold to be higher than 1. Alternatively, instead of changing the threshold values, we could change the spam/ham edits ratio in the training set. To decrease false positives, one needs to increase more ham edits in the training set. To increase spam recalls, one needs to increase the number of edits in the training set.

My performance may be compared with the results reported in Smets et al. [129]. Overall, my results appear to outperform theirs, even those that use probabilistic sequence modeling. Their precision and recall values for classifying revision pages using only insertions models are 12.74% and 92.81%, and those using only change models are 11.77% and 83.62%. When they combine various predictors to judge if a revision page is a spam or not, their reported precision is 32.09% and their reported recall is 91.71%. However, their test edits contained multiple types of vandalism, where I focus on single-type edits. Moreover, I do not attempt to reproduce their results, and I may have misinterpreted some aspects of their experiments. Nonetheless, in contrast to their conclusions, my results suggest that compression models may provide a useful method for identifying spam edits.

My differing performance might stem from a more balanced training corpus than that used by Smets et al. [129]. The effect of preserving Wikipedia tags may also have contributed to the overall performance.

My performance may also be compared with that of Potthast [115]. I slightly underperform their approach. However, I demonstrate that compression algorithms alone can compete reasonably well against classifiers that use hand crafted features.

8.7 Conclusions

In this chapter, I employ the DMC compression model to classify new edits in Wikipedia articles as spam or ham. In contrast to previous work, my results suggest that compression models might provide a useful method for classifying these edits. I also showed that having a balanced training set is important. Future work might combine the results of compression models with other features and methods. It may also be possible to vary training methods, or increase the size of the training sets, to obtain additional improvements.

Chapter 9

Conclusions and Future Work

9.1 Contributions of my Thesis

In this thesis, I study the problem of focused heterogeneous information retrieval. Focused retrieval provides a concise summary to a user's information needs, as opposed to presenting to a user entire documents, such as whole books. Current search engines such as Google do provide pseudo-focused results in the form of snippets, but because their goals are to rank relevant documents, some relevant passages of rather irrelevant documents may be omitted.

Retrieval of heterogeneous sources such as books and web pages would be helpful in real life. Although the major search engines studied in the thesis do perform heterogeneous search, only Google performs heterogeneous focused search of books and web pages by returning the relevant book parts.

To arrive at a unified framework for focused heterogeneous retrieval, I look at the three questions that arise when comparing traditional homogeneous document retrieval against the heterogeneous focused retrieval.

- **Efficiency:** Scoring all possible passages in a document takes $\mathcal{O}\left(n^2\right)$ times more than scoring an entire document. Is it possible to reduce the computational overload?
- **Size:** A passage can be as short as a single character (or even a single bit) to as long as an entire document. What is the best unit of retrieval?
- **Extra-textual Information:** One way to define passage boundaries is to use markup such as XML. Is it possible to take advantage of such structural information?

9.1.1 Mathematical Model of Information Retrieval

Chapter 3 helps answer the first question of computational overload inherent in focused retrieval, by presenting a possible approach to an almost linear time algorithm to perform passage retrieval. Having an almost linear time algorithm means we only need to look at each term in the text once, whereas my current approach of passage retrieval requires scoring overlapped parts multiple times.

Here, I rephrase the mathematical problem of passage retrieval colloquially:

> A document with n query terms has $\mathcal{O}\left(n^2\right)$ possible passages that start and end with query terms. Of these, return a set of passages such that the passages are in decreasing order of scores, and no passage of lesser score overlaps with any passages of higher scores.

This problem is motivated by Ad Hoc track Focused task where overlapped text are not permitted to return. To devise an algorithm, I also need a scoring function. For this purpose, I created a simple passage scoring function based on pivoted length normalization [128]:

$$\sum_{p_i \in [u,v]} \frac{W_i}{l + \Delta} \quad . \tag{9.1}$$

When Equation 9.1 is applied to an overlapping pair of passages, the comparison of the scores becomes the comparison of term densities. I show that this could be used to help reduce the problem of passage retrieval into that of finding a densest segment that contains the mid-point of a passage.

My final goal in Chapter 3 is to analyze the two commonly-used scoring functions, BM25 [122] and language model [114] in the passage retrieval setting. For a single-term BM25, I show that comparing two passages with single-BM25 is the same as comparing the passages with density function, with a constant Δ added to the denominator.

$$\begin{align} & S\,[u,v] \geq S\,[u',v'] \tag{9.2} \\ \Leftrightarrow \quad & \tfrac{\vec{f}}{l_1+\Delta} \geq \tfrac{\vec{g}}{l_2+\Delta} \quad . \tag{9.3} \end{align}$$

This leads to the similar reduction as Equation 9.1.

For language modeling, I show that comparison of two single-term LM score reduces into the form similar to, but more convoluted than, Equation 9.1.

$$\begin{align} & S\,[u,v] \geq S\,[u',v'] \tag{9.4} \\ \Leftrightarrow \quad & \tfrac{t_1\alpha+1}{\beta l_1+1} \geq \tfrac{t_2\alpha+1}{\beta l_2+1} \quad . \tag{9.5} \end{align}$$

where $\alpha = (l_{\mathcal{C}}) \,/\, (n_t \mu)$ and $\beta = 1/\mu$.

My contributions are as follows.

- **A Reduction** of finding the highest scoring passage using my pivoted-length-normalization-based scoring function into that of finding the densest segment that contains the mid-point.

- **Derivations** of the single-term BM25, and the single-term language modeling scores into the form of my scoring function. The derivations give a new perspective of how each scoring function performs length normalization.

9.1.2 Granularity of Retrieval

Using the model and the problem of focused retrieval established in Chapter 3, Chapter 4 explains implementation details of the model I use for experiments in this thesis. Chapter 5 then explores the second question of focused retrieval; the ideal unit of retrieval.

The first experiment shows that simply scoring XML elements is superior to Trotman and Geva's proposition [133] of turning the best passages into elements by taking the smallest elements that completely contains the passages.

The second experiment runs my focused retrieval approach in Chapter 4, but with five different levels of granularities of retrieval. Surprisingly, when trained on mean average precision, retrieving only documents performed better than any other retrieval strategies. In fact, the performance of different retrieval units is in the order of article, single elements, range of elements, semi-arbitrary passages, and arbitrary passages. However, most of the differences between different granularity of retrieval on both iP[0.01] and MAP are not statistically significant and thus I can not draw a firm conclusion other than that, for the purpose of INEX Ad Hoc track Focused task on the Wikipedia collection [25], retrieving single element suffices.

I conclude as follows:

- **Single XML Elements** are indeed a reasonable unit of retrieval for focused retrieval on Wikipedia documents, hence closing an open question in INEX as well as my second question to focused retrieval.

- **Evaluation Measure** that only considers the first two results may not capture the effectiveness of focused retrieval.

- **Different Strategies** may be needed for different units of retrieval.

9.1.3 Scoring Function

Chapter 6 answers the last question of focused retrieval; utility of XML structure and extra-textual information. It develops on the prior work by Robertson et al. [92, 119] to utilize BM25F in a simpler manner. It introduces only two fields, the body field and the characteristic field. The latter field can contain information that characterizes the element of interest by absorbing contextual information such as section titles. The characteristic field can also accept extra-textual information such as keywords and Library of Congress classification numbers. My contributions are as follows.

- **Characteristic field** lets us take advantage of structural information as well as extra-textual information. It answers my third question of focused retrieval.

- **Heterogeneous Retrieval** is made easier by having a characteristic field that is not necessarily tied to an XML element.

9.1.4 Focused Links

Focused retrieval applies to more than a retrieval of texts. Chapter 7 studies linking focused part of texts to another focused part of texts; finding an anchor phrase and a destination best-entry-point in a Wikipedia page without link annotations.

I first showed the *ST algorithm* to be a simple, yet effective way of linking Wikipedia pages together. The ST algorithm builds a statistics of frequency of anchor phrase-destination article pairs, and then finds and embeds the most frequent pairs into a new Wikipedia document without any link information. The ST algorithm is successfully extended to anchor-to-best-entry-point linkage. This approach and its variations have been implemented by different participants including myself at INEX Link-the-Wiki Tracks to show difficult-to-beat performance [57, 58, 60].

In trying to improve over the ST algorithm, I combine the popularity of a page measured by PageRank [6] with the semantic similarity of pages by Topical PageRank [112] using K-L divergence. The result show that the new approach, KPR algorithm, gives an improvement at an early recall level over the ST algorithm. Though the KPR algorithm targets embedding outgoing links from a new Wikipedia page, I successfully apply the inverted version of the KPR algorithm in finding incoming links to a new Wikipedia page [60, 69]. My contributions are as follows:

- **The ST algorithm** and its variation is sufficient to reproduce outgoing links from a linkless Wikipedia page, hence closing an open question at INEX Link-the-Wiki Track.
- **Topical Links and Structural Links** The former is characterized by ST algorithm, latter by KPR algorithm.
- **The KPR algorithm** improves over the ST algorithm a little on embedding outgoing links, however its main contribution is to embedding incoming links to a linkless Wikipedia page.

9.1.5 Focused Spam Filtering

Another application of focused retrieval is focused spam filtering. Unlike traditional email or web pages, in which only the entire content can be a spam, any part of a Wikipedia document can be a vandalism, or spam.

In my experiment, I used a pre-marked-up edit on Wikipedia history pages to detect whether a new edit is a vandalism or not. I improve on the similar prior work of Smets et al. [129] by, among other things, only judging a page with similar kinds of edits. In this limited framework, a compression model can be shown to be effective on detecting vandalism. I also show that having a balanced spam and ham training sets help in improving the F-measure. My contributions are as follows.

- **A Compression Model** can work well in a specific setting, in my case, when only one kinds of edits are judged.
- **A Balanced Training Set** produces better prediction than a training set that consists mostly of ham edit, reflecting the real-world scenario.

9.2 Future Work

This thesis can be extended in various ways. Most of the future work described in this section would strengthen my points in the thesis. However, I think there is a strong need to move forward to a new open problem; a more interesting data set and applications in either focused retrieval or XML retrieval.

9.2.1 Mathematical Model of Information Retrieval

I reduced the problem of finding the highest scoring passage in linear time into that of finding the densest passage that contains the mid-point in linear time. I also found an algorithm by Chung and Lu [13] that may help me in constructing a linear time algorithm to find the densest extent that passes through the mid-point. Their motivation implies bioinformatics application. The next step would be to decide how I actually apply their algorithm in my setting. Specifically,

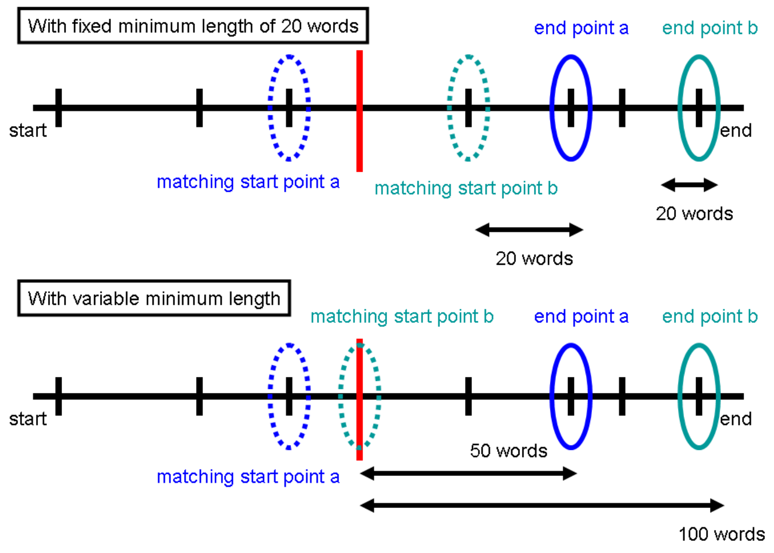


Figure 9.1: Fixed Min v.s. Variable Min

in the first part of their paper, Chung and Lu discusses a linear-time algorithm to find the densest segment with a minimum length constraint. That is, in the setting of information retrieval, in an article, we are looking for a sequence of words of at least certain length, that contains the most number of query terms per word. The algorithm moves the end point of a segment from the beginning of the entire segment till the end. In Figure 9.1, an end point moves from the left to the right. For each end point, the algorithm finds the matching start point of the current segment ending at the end point that maximizes the density with the minimum length constraint. In Figure 9.1, in the top, we see the matching start points that maximizes the density of segments ending at end points a and b. Both matching start points are at least 20 words away from the end points because the minimum length constraint is 20 words. Thus, in the case of Chung and Lu, at each end point, the minimum length is fixed.

However, if it was possible to change the minimum length at every end point without affecting the run-time, I could do the following to find the densest segment that includes the mid point: After the end point passes the mid-segment, increase the minimum length such that all segments considered after passing the mid point must include the midpoint. In Figure 9.1 bottom, the minimum length for the end point a was increased from the original 20 words to 50 words; if it was less than 50 words, the densest segment ending at an end point a might not include the mid point. Similarly, the minimum length for a segment ending at an end point b was increased from 20 to 100 words.

Thus the algorithm used to find the densest segment with minimum length may possibly be converted into an algorithm to find the densest segment that includes the mid point. Though the proofs of the paper seem applicable in the variable minimum length constraint, I have been unable to provide a simple reduction to prove this.

Once I found a linear time algorithm to find the densest segment that contains the mid point, I could use it to find the highest scoring passage, by pivoted-length normalization function, in $\mathcal{O}(n \log n)$. From there, my goal is to find all highest scoring passages in a similar time frame.

Once I solve my passage retrieval problem on my own scoring function, I would like to solve the problem using BM25 and language modeling as scoring functions. So far, I reduced the single-term BM25 and language model scores in forms similar to my scoring function, but the multiple-term counterparts seems to have an added convolutedness.

9.2.2 Granularity of Retrieval

My first conclusion in Section 9.1.2 implies that I need to look for a more appropriate venue in which XML will be helpful in the retrieval of textual data. One such venue is the new INEX 2010 Data-Centric Track [1]. It uses the Internet Movie Database [2] as a corpus. The IMDB corpus has annotations such as movie titles, actors, and directors, that may be useful for a retrieval of information that answers various user needs. For example, a user may be looking for a movie

[1] `http://www.inex.otago.ac.nz/tracks/strong/strong.asp`

[2] `http://www.imdb.com/`

title in which a particular actress has appeared.

My second conclusion in Section 9.1.2 implies that I need to look for a good evaluation metrics that better reflects the user's view. As I discussed in Section 5.3.3, the current official metrics of interpolated precision at 0.01 recall level for focused retrieval at INEX Ad Hoc Track only considers the first couple of results returned by search systems. The mean average precision measure may be reasonable, however, when my article-only retrieval system performs the best when trained on MAP, either a different evaluation measure is needed or a different way of implementing varied-granularity retrieval system is needed.

This leads to the discussion of the third conclusion in Section 9.1.2. It is possible that under some metrics, all units of retrieval perform the same. What it should not be possible is a true arbitrary passage retrieval system performing much worse than an article-only retrieval system. Since my arbitrary passage retrieval system returns a passage starting and ending with a query word, whereas a user's highlighted passage would highly likely to start at least at the beginning of a sentence, if not the beginning of a paragraph, it is natural for my arbitrary passage retrieval system to perform slightly worse than other systems. My task then is to realize how an arbitrary passage retrieval system performs when the beginning of a passage is pushed back to the beginning of a sentence or paragraph boundaries. A comparison of highlighted passages against the results returned by my algorithm is also needed.

Overall, I think the best course of action for the next step in the research of XML text retrieval is to shift into a more data-centric retrieval task such as using IMDB as a corpus as in INEX 2010 Data-centric track. The problems I see as a current state (up until 2009) of INEX are the kind of markup used and the kind of information to retrieve. First, when a user reads text, sections and paragraphs seem like a natural unit of retrieval. It would only seem to me to be of help if other textual information is marked up. For example, proper nouns can be marked up to indicate if it is a person or a brand of cars. Second, even with the help of non-structural markup, I am not sure if the retrieval of text itself is interesting. Information extraction, such as extracting relationships of royalties or proteins may be more useful with markup.

9.2.3 Scoring Function

In order to support my first conclusion in Section 9.1.3 of the effectiveness of the characteristic field, more experiments are needed. For example, in the Wikipedia focused retrieval, I only use two-levels of titles, article and section titles, as part of the characteristic field. With the new INEX 2009 Wikipedia collection with deeper sub-headings, I can expand the content of the characteristic field. In addition to titles, I could incorporate other parts of a Wikipedia article, marked-up by XML, into the characteristic field. For example, the Wikipedia article on "Impressionism" contains four `<category>` elements in the heading; "Semi-protected", "French art", "Impressionism", and "Art movements". Though the first category does not make sense, adding the other three categories into the characteristic field, in addition to the article and section headings may be useful.

My second point in Section 9.1.3 is only tested half-way. Using the characteristic field does give improvement over different collections, Wikipedia and Books, when tested separately, but I did not test on a heterogeneous collection. One way to test is to use data from INEX Heterogeneous Track from 2004 to 2007, however it is unclear if there exists an entire test set with relevance assessments [31].

9.2.4 Focused Links

The ST algorithm seems to have closed the open question of embedding Wikipedia links and I think it is time to move on to a different task. There are still two open questions in the INEX Link-the-"Wiki" track. The first question is to embed a set of incoming links into a new linkless Wikipedia page. The other is to embed links in pages from a corpus that does not contain existing link structure. INEX provides TeAra collection [3], an encyclopedia about New Zealand to perform this task.

To embed a set of incoming links, the KPR algorithm performs well in my INEX 2009 submission [58, 69]. Recall, however, that the KPR algorithm filters the number of candidate incoming pages by the existence of a topic title in the pages. The baseline incoming run, on the other hand, in which I return the top documents when a topic title is used as a query, does not have this filtering process. Thus the first thing I should do, is to evaluate the performance of incoming baseline run when the existence of a topic title is used to filter the candidate pages.

To tackle the TeAra corpus, I would first use the statistics I use for the ST algorithm to embed outgoing links in the TeAra corpus. The incoming links would be created by querying each page title into all other pages in the corpus. On top of that KPR algorithm could be performed on both incoming and outgoing links.

9.2.5 Focused Spam Filtering

In 2010, there was a new initiative of Wikipedia vandalism detection, at CLEF [4]. Submitting runs based on my algorithm for the next year's iteration to compare the performance against others is the first step of my future work.

For Wikipedia documents, each edit is already annotated as insertion or change edits, reducing the problem of focused retrieval of vandalized passages into judgements of passages. From the administrator's point of view, they need only check the edit against vandalism. From a user's point of view, however, they may not have an access to the older version of the page. In this case, if a user wants to be sure that there is no vandalism in a document, the user needs to find parts of a document that may be vandalized.

[3] `www.teara.govt.nz/`

[4] `http://www.uni-weimar.de/medien/webis/research/workshopseries/pan-10/`

When there is no "edit" markup, the task of focused vandalism retrieval becomes similar to a plagiarism detection, in a way that one looks for passages whose stylistic is dissimilar to the overall text. In these scenarios, I would keep track of a compression ratio of the passage starting from the beginning of a document/paragraph and ending at each character/bits. Any change of compression ratio may indicate the start or end of plagiarism. CLEF has also been offering plagiarism detection track since 2007, so participating in CLEF is the next step.

9.3 Conclusion

In summary, I closed two open problems, one large, and one small, in INEX, and XML text retrieval in general.

- **What is the ideal granularity of retrieval? (Ad Hoc Track)** Single XML elements suffice for the purpose of Ad Hoc retrieval at least on the current and past Wikipedia collections.
- **How do we link Wikipedia pages together? (Link-the-Wiki Track)** Create a statistics on anchor-phrase-destination-page pairs from the Wikipedia corpus, find the anchors in the new Wikipedia page.

My work is limited by the corpora used, and both strategies are simple, however, multiple experiments by myself and other participants failed to submit any run that performs statistically significantly better in the Ad Hoc Track, and no better run was submitted in the Link-the-Wiki Track [33, 45, 57, 58, 60, 73] during 2007 and 2009.

In addition, my answers to the questions asked in Section 1 are as follows.

- **Unit of Retrieval:** This is one of the open questions I closed above.

- **Computational Overload:** The answer to the above question implies that the number of passages we need to score is limited.
- **Extratextual Information:** The characteristic field lets us lump extra-textual information, be it an XML element or not, into the single field for BM25F-based retrieval.

Bibliography

[1] Steven Abney, Michael Collins, and Amit Singhal. Answer extraction. In *Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP2000)*, pages 296–301, 2000.

[2] Sampath Kannan Andrew G. West and Insup Lee. Detecting wikipedia vandalism via spatio-temporal analysis of revision metadata. Technical Report MS-CIS-10-05, University of Pennsylvania, 2010.

[3] Jeremy Barbay, Ehsan Chiniforooshan, Alexander Golynski, Jui-Yi Kao, and Aleh Veraskouski. Generalized labeled LCA queries. Technical Report CS-2006-31, University of Waterloo, 200 University Ave. W., Waterloo, ON, Canada, 2006.

[4] Michel Beigbedger. ENSM-SE at INEX 2007: Scoring with proximity. In *Pre-Proceedings of INEX 2007*, pages 53–55, 2007.

[5] Andrej Bratko, Bogdan Filipič, Gordon V. Cormack, Thomas R. Lynam, and Blaž Zupan. Spam filtering using statistical data compression models. *Journal of Machine Learning Research*, 7:2673–2698, 2006.

[6] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[7] Andreas Broschart, Ralf Schenkel, and Martin Theobald. Experiments with proximity-aware scoring for XML retrieval at INEX 2008. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 29–32. Springer-Verlag, 2009.

[8] James P. Callan. Passage-level evidence in document retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1994)*, pages 302–310, 1994.

[9] Kuan-Yu Chen and Kun-Mao Chao. Optimal algorithms for locating the longest and shortest segments satisfying a sum or an average constraint. *Information Processing Letters*, 96(6):197–201, 2005.

[10] Mao-Lung (Edward) Chen, Richi Nayak, and Shlomo Geva. Link-the-Wiki: Performance evaluation based on frequent phrases. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 326–336. Springer-Verlag, 2009.

[11] Yen-Hung Chen, Hsueh-I Lu, and Chuan-Yi Tang. Disjoint segments with maximum density. In *Proceedings of the International Conference on Computational Science (ICCS2005): International Workshop on Bioinformatics Research and Applications*, pages 845–850, 2005.

[12] Si-Chi Chin, W. Nick Street, Padmini Srinivasan, and David Eichmann. Detecting wikipedia vandalism with active learning and statistical language models. In *Proceedings of the 4th Workshop on Information Credibility on the Web (WICOW2010)*, pages –, 2010.

[13] Kai-Min Chung and Hsueh-I Lu. An optimal algorithm for the maximum-density segment problem. *SIAM Journal on Computing*, 34(2):373–387, 2005.

[14] Charles L. A. Clarke. Controlling overlap in content-oriented XML retrieval. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2005)*, pages 314–321, 2005.

[15] Charles L. A. Clarke, Gordon V. Cormack, and Thomas R. Lynam. Exploiting redundancy in question answering. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2001)*, pages 358–365, 2001.

[16] Charles L. A. Clarke, Gordon V. Cormack, and Elizabeth A. Tudhope. Relevance ranking for one to three term queries. *Information Processing & Management*, 36(2):291–311, 2000.

[17] John G. Cleary and Ian H. Witten. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, 32(4):396–402, 1984.

[18] comScore, Inc. Press Release. comScore Releases April 2010 U.S. Search Engine Rankings. `http://www.comscore.com/Press_Events/Press_Releases/2010/5/comScore_Releases_April_2010_U.S._Search_Engine_Rankings`.

[19] Gordon V. Cormack. Dynamic Markov compression (DMC) version 0.0.0, 1993/1987. [Online; accessed 09-February-2009].

[20] Gordon V. Cormack and R. Nigel Horspool. Data compression using dynamic Markov modelling. *The Computer Journal*, 30(6):541–550, 1987.

[21] Nick Craswell, Hugo Zaragoza, and Stephen Robertson. Microsoft Cambridge at TREC 14: Enterprise track. In *The Fourteenth Text REtrieval Conference Proceedings (TREC 2005)*, pages –, 2005.

[22] Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th Annual Inter-*

national ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2005), pages 400–407, 2005.

[23] Nicolas Usunier David Buffoni and Patrick Gallinari. LIP6 at INEX09: OWPC for Ad Hoc Track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Focused Retrieval and Evaluation*, volume 6203 of *LNCS*, pages 59–69. Springer-Verlag, 2010.

[24] Stefan Decker, Frank van Harmelen, Jeen Broekstra, Michael Erdmann, Ian Horrocks, Michel Klein, and Sergey Melnik. The Semantic Web: the roles of XML and RDF. *IEEE Internet Computing Magazine*, 4(5):63–73, 2000.

[25] Ludovic Denoyer and Patrick Gallinari. The Wikipedia XML corpus. *SIGIR Forum*, 40(1):64–69, 2006.

[26] Lei Dong and Carolyn Watters. Improving efficiency and relevance ranking in information retrieval. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI2005)*, pages 648–651, 2004.

[27] Philipp Dopichaj. The University of Kaiserslautern at INEX 2005. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors, *Advances in XML Information Retrieval and Evaluation*, volume 3977 of *LNCS*, pages 196–210. Springer-Verlag, 2006.

[28] Philipp Dopichaj, Andre Skusa, and Andreas He. Stealing anchors to link the Wiki. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 343–353. Springer-Verlag, 2009.

[29] Khairun Nisa Fachry, Jaap Kamps, Marijn Koolen, and Junte Zhang. Using and detecting links in Wikipedia. In Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *Focused Access to XML Documents*, volume 4862 of *LNCS*, pages 388–403. Springer-Verlag, 2008.

[30] Hui Fang and ChengXiang Zhai. An exploration of axiomatic approaches to information retrieval. In *Proceedings of the 28th international ACM SIGIR conference on Research and development in information retrieval (SIGIR2005)*, pages 480–487, 2005.

[31] Ingo Frommholz and Ray Larson. Report on the INEX 2006 heterogeneous collection track. *SIGIR Forum*, 41(1):75–78, 2007.

[32] Norbert Fuhr and Norbert G editors. *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX).*

[33] Norbert Fuhr, Jaap Kamps, Mounia Lalmas, Saadia Malik, and Andrew Trotman. Overview of the INEX 2007 Ad Hoc track. In Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *Focused Access to XML Documents*, volume 4862 of *LNCS*, pages 1–23. Springer-Verlag, 2008.

[34] Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors. *Overview of the INEX 2007 Ad Hoc Track*, Schloss Dagstuhl, Germany, 2008. Springer.

[35] Norbert Fuhr, Mounia Lalmas, and Saadia Malik, editors. *INEX 2003 Workshop Proceedings*, Schloss Dagstuhl, Germany, 2004.

[36] Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors. *Advances in XML Information Retrieval and Evaluation*, Schloss Dagstuhl, Germany, 2006. Springer.

[37] Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors. *Advances in XML Information Retrieva*, Schloss Dagstuhl, Germany, 2005. Springer.

[38] Norbert Fuhr, Mounia Lalmas, and Andrew Trotman, editors. *Comparative Evaluation of XML Information Retrieval Systems*, Schloss Dagstuhl, Germany, 2007. Springer.

[39] Norbert Fuhr, Saadia Malik, and Mounia Lalmas. Overview of the INitiative for the Evaluation of XML retrieval (INEX) 2003. In *INEX 2003 Workshop Proceedings*, pages 1–11, 2004.

[40] Marijn Koolen Gabriella Kazai, Antoine Doucet and Monica Landoni. Overview of the inex 2009 book track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Focused Retrieval and Evaluation*, volume 6203 of *LNCS*, pages 145–159. Springer-Verlag, 2010.

[41] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of The 20th International Joint Conference for Artificial Intelligence (IJCAI-07)*, pages 1606–1611, 2007.

[42] James J. Gardner and Li Xiong. Automatic link detection: a sequence labeling approach. In *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM2009)*, pages 1701–1704, 2009.

[43] Shlomo Geva. GPX gardens point XML information retrieval at INEX 2004. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors, *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages –. Springer-Verlag, 2005.

[44] Shlomo Geva. GPX gardens point XML IR at INEX 2005. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors, *Advances in XML Information Retrieval and Evaluation*, volume 3977 of *LNCS*, pages 240–253. Springer-Verlag, 2006.

[45] Shlomo Geva, Jaap Kamps, Miro Lethonen, Ralf Schenkel, James A. Thom, and Andrew Trotman. Overview of the INEX 2009 Ad Hoc track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Focused Retrieval and Evaluation*, volume 6203 of *LNCS*, pages 4–25. Springer-Verlag, 2010.

[46] Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors. *Advances in Focused Retrieval*, Schloss Dagstuhl, Germany, 2009. Springer.

[47] Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors. *Focused Retrieval and Evaluation*, Brisbane, Australia, 2010. Springer.

[48] Michael H. Goldwasser, Ming-Yang Kao, and Hsueh-I Lu. Linear-time algorithms for computing maximum-density sequence segments with bioinformatics applications. *Journal of Computer and System Sciences*, 70(2):128–144, 2005.

[49] Google Inc. Press Center. Google Begins Move to Universal Search. `http://www.google.com/intl/en/press/pressrel/universalsearch_20070516.html`.

[50] Nobert Gövert and Mohammad Abolhassani. Content-oriented XML retrieval with HyREX. In *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 26–32, 2003.

[51] Norbert Gövert and Gabriella Kazai. Overview of the initiative for the evaluation of XML retrieval (INEX) 2002. In *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 1–17, 2003.

[52] Michael Granitzer, Christin Seifert, and Mario Zechner. Context based Wikipedia linking. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 354–365. Springer-Verlag, 2009.

[53] Teresa Hackett. Global library statistics 1990-2000. pages –, 2003.

[54] Djoerd Hiemstra and Wessel Kraaij. Twenty-one at trec-7: ad-hoc and cross-language track. In *The Seventh Text REtrieval Conference (TREC-7)*, pages 227–238, 1999.

[55] Eduard Hovy, Ulf Hermjakob, and Chin yew Lin. The use of external knowledge in factoid QA. In *The Tenth Text REtrieval Conference (TREC 2001)*, pages 644–652, 2001.

[56] Chong Huang, Yonghong Tian, Zhi Zhou, and Tiejun Huang. Towards multi-granularity multi-facet e-book retrieval. In *Proceedings of the 16th International Conference on World Wide Web (WWW2007)*, pages 1331–1332, 2007.

[57] Darren Wei Che Huang, Yue Xu, Andrew Trotman, and Shlomo Geva. Overview of INEX 2007 Link the Wiki track. In Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *Focused Access to XML Documents*, volume 4862 of *LNCS*, pages 373–387. Springer-Verlag, 2008.

[58] Wei Che Huang, Shlomo Geva, and Andrew Trotman. Overview of the INEX 2008 Link the Wiki track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 314–325. Springer-Verlag, 2009.

[59] Wei Che Huang, Andrew Trotman, and Shlomo Geva. The importance of manual assessment in link discovery. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR2009)*, pages 698–699, 2009.

[60] Wei Che (Darren) Huang, Shlomo Geva, and Andrew Trotman. Overview of the INEX 2009 Link the Wiki track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Focused Retrieval and Evaluation*, volume 6203 of *LNCS*, pages 312–323. Springer-Verlag, 2010.

[61] Weihua Huang, Andrew Trotman, and Richard O'Keefe. Elemement retrieval using a passage retrieval approach. In *Proceeding of the 11th Australian Document Computing Symposium (ADCS2006)*, 2006.

[62] Xiaoqui Huang. An algorithm for identifying regions of a DNA sequence that satisfy a content requirement. *Bioinformatics*, 10(3):219–225, 1994.

[63] Fidelia Ibekwe-SanJuan and Eric SanJuan. Use of multiword terms and query expansion for interactive information retrieval. In Shlomo Geva, Jaap Kamps, and Andrew Trotman,

editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 54–64. Springer-Verlag, 2009.

[64] Kelly Y. Itakura and Charles L. A. Clarke. From passages into elements in XML retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2007) Workshop on Focused Retrieval*, pages 17–27, 2007.

[65] Kelly Y. Itakura and Charles L. A. Clarke. University of Waterloo at INEX2007: Adhoc and Link-the-Wiki tracks. In Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *Focused Access to XML Documents*, volume 4862 of *LNCS*, pages 417–425. Springer-Verlag, 2008.

[66] Kelly Y. Itakura and Charles L. A. Clarke. University of Waterloo at INEX 2008: Adhoc, Book, and Link-the-Wiki tracks. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 132–139. Springer-Verlag, 2009.

[67] Kelly Y. Itakura and Charles L. A. Clarke. Using dynamic Markov compression to detect vandalism in the Wikipedia. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2009)*, pages 822–823, 2009.

[68] Kelly Y. Itakura and Charles L. A. Clarke. A framework for BM25F-based XML retrieval. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2010)*, pages 843–844, 2010.

[69] Kelly Y. Itakura and Charles L. A. Clarke. University of Waterloo at INEX 2009: Ad Hoc, Book, Entity Ranking, and Link-the-Wiki tracks. In Shlomo Geva, Jaap Kamps, and

Andrew Trotman, editors, *Focused Retrieval and Evaluation*, volume 6203 of *LNCS*, pages 331–341. Springer-Verlag, 2010.

[70] Abraham Ittycheriah, Martin Franz, and Salim Roukos. IBM's statistical question answering system-TREC-10. In *The Tenth Text REtrieval Conference (TREC 2001)*, pages 258–264, 2001.

[71] Dylan Jenkinson, Kai-Cheung Leung, and Andrew Trotman. Wikisearching and Wikilinking. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 374–388. Springer-Verlag, 2009.

[72] Jaap Kamps, Maarten de Rijke, and Börkur Sigurbjörnsson. Length normalization in XML retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2004)*, pages 80–87, 2004.

[73] Jaap Kamps, Shlomo Geva, Andrew Trotman, Alan Woodley, and Marijn Koolen. Overview of the INEX 2008 Ad Hoc track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 1–28. Springer-Verlag, 2009.

[74] Jaap Kamps and Marijn Koolen. On the relation between relevant passages and XML document structure. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2007) Workshop on Focused Retrieval*, pages 28–32, 2007.

[75] Jaap Kamps, Maarten Marx, Maarten de Rijke, and Börkur Sigurbjörnsson. Structured queries in XML retrieval. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management (CIKM2005)*, pages 4–11, 2005.

[76] Marcin Kaszkiel and Justin Zobel. Passage retrieval revisited. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1997)*, pages 178–185, 1997.

[77] Gabriella Kazai, Antoine Doucet, and Monica Landoni. Overview of the INEX 2008 Book track. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Advances in Focused Retrieval*, volume 5631 of *LNCS*, pages 106–123. Springer-Verlag, 2009.

[78] Gabriella Kazai and Mounia Lalmas. INEX 2005 evaluation measures. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors, *Advances in XML Information Retrieval and Evaluation*, volume 3977 of *LNCS*, pages 16–29. Springer-Verlag, 2006.

[79] Gabriella Kazai, Mounia Lalmas, and Arjen P. de Vries. The overlap problem in content-oriented XML retrieval evaluation. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR2004)*, pages 72–79, 2004.

[80] Gabriella Kazai and Natasa Milic-Frayling. Effects of social approval votes on search performance. In *Proceedings of the 6th International Conference on Information Technology : New Generations (ITNG2009)*, pages 1554–1559, 2009.

[81] Koichi Kise, Markus Junker, Andreas Dengel, and Keinosuke Matsumoto. Experimental evaluation of passage-based document retrieval. In *Proceedings of the 6th International Conference on Document Analysis and Recognition (ICDAR2001)*, page 592, 2001.

[82] Aniket Kittur, Ed H. Chi, Bryan A. Pendleton, Bongwon Suh, and Todd Mytkowicz. Power of the few vs. wisdom of the crowd: Wikipedia and the rise of the bourgeoisie. In *Proceedings of the 25th Annual ACM Conference on Human Factors in Computing Systems (CHI2007)*, 2007.

[83] Daniel Knaus, Elke Mittendorf, and Peter Schäuble. Improving a basic retrieval method by links and passage level evidence. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 241–246, 1994.

[84] Birger Larsen, Anastasios Tombros, and Saadia Malik. Is XML retrieval meaningful to users?: searcher preferences for full documents vs. elements. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2006)*, pages 663–664, 2006.

[85] Gary G. Lee, Jungyun Seo, Seungwoo Lee, Hanmin Jung, Bong-Hyun Cho, Changki Lee, Byung-Kwan Kwak, Jeongwon Cha, Dongseok Kim, JooHui An, Harksoo Kim, and Kyungsun Kim. SiteQ: Engineering high performance QA system using lexico-semantic pattern matching and shallow NLP. In *The Tenth Text REtrieval Conference (TREC 2001)*, pages 442–451, 2001.

[86] Marc Light, Gideon S. Mann, Ellen Riloff, and Eric Breck. Analyses for elucidating current question answering technology. *Natural Language Engineering*, 7(4):325–342, 2001.

[87] Andrew Lih. Wikipedia as participatory journalism: Reliable sources? Metrics for evaluating collaborative media as a news resource. In *Fifth International Symposium on Online Journalism*, 2004.

[88] Rung-Ren Lin, Wen-Hsiung Kuo, and Kun-Mao Chao. Finding a length-constrained maximum-density path in a tree. *Journal of Combinatorial Optimization*, 9(2):147–156, 2005.

[89] Jingjing Liu, Hongfei Lin, and Bing Han. Study on reranking XML retrieval elements based on combining strategy and topics categorization. In *Pre-Proceedings of INEX 2007*, pages 170–176, 2007.

[90] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[91] Xiaoyong Liu and W. Bruce Croft. Passage retrieval based on language models. In *Proceedings of the 11th ACM Conference on Information and Knowledge Management (CIKM2002)*, pages 375–382, 2002.

[92] Wei Lu, Stephen Robertson, and Andrew MacFarlane. Field-weighted XML retrieval based on BM25. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors, *Advances in XML Information Retrieval and Evaluation*, volume 3977 of *LNCS*, pages 161–171. Springer-Verlag, 2006.

[93] Wei Lu, Stephen Robertson, and Andrew Macfarlane. CISR at INEX 2006. volume 4518 of *LNCS*, pages 57–63. Springer-Verlag, 2007.

[94] John Makhoul, Francis Kubala, Timothy Leek, Daben Liu, Long Nguyen, Richard Schwartz, and Amit Srivastava. Speech and language technologies for audio indexing and retrieval. *Proceedings of the IEEE*, 88(8):1338–1353, 2000.

[95] Saadia Malik, Gabriella Kazai, Mounia Lalmas, and Norbert Fuhr. Overview of INEX 2005. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors, *Advances in XML Information Retrieval and Evaluation*, volume 3977 of *LNCS*, pages 1–15. Springer-Verlag, 2006.

[96] Saadia Malik, Mounia Lalmas, and Norbert Fuhr. Overview of INEX 2004. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors, *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 1–15. Springer, 2005.

[97] Saadia Malik, Andrew Trotman, Mounia Lalmas, and Norbert Fuhr. Overview of INEX 2006. In Norbert Fuhr, Mounia Lalmas, and Andrew Trotman, editors, *Comparative Evalu-*

ation of XML Information Retrieval Systems, volume 4518 of *LNCS*, pages 1–11. Springer-Verlag, 2007.

[98] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*.

[99] Rianne Kaptein Marijn Koolen and Jaap Kamps. Focused search in Books and Wikipedia: Categories, Links and Relevance Feedback. In Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors, *Focused Retrieval and Evaluation*, volume 6203 of *LNCS*, pages 273–291. Springer-Verlag, 2010.

[100] Yosi Mass and Matan Mandelbrod. Retrieving the most relevant XML components. In *INEX 2003 Workshop Proceedings*, pages 53–58, 2004.

[101] Yosi Mass and Matan Mandelbrod. Component ranking and automatic query refinement for XML retrieval. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors, *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 73–84. Springer-Verlag, 2005.

[102] Yosi Mass, Matan Mandelbrod, Einat Amitay, David Carmel, Yoelle Maarek, and Aya Soffer. JuruXML - an XML retrieval system at INEX'02. In *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 73–80, 2003.

[103] Massimo Melucci. Passage retrieval: a probabilistic technique. *Information Processing & Management*, 34(1):43–68, 1998.

[104] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM2007)*, pages 233–242, 2007.

[105] David R. H. Miller, Tim Leek, and Richard M. Schwartz. A hidden markov model information retrieval system. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1999)*, pages 214–221, 1999.

[106] David Milne and Ian H. Witten. Learning to link with Wikipedia. In *Proceedings of the 17th ACM conference on Information and Knowledge Management (CIKM2008)*, pages 509–518, 2008.

[107] Elke Mittendorf and Peter Schäuble. Document and passage retrieval based on hidden Markov models. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1994)*, pages 318–327, 1994.

[108] Alistair Moffat and Justin Zobel. Self-indexing inverted files for fast text retrieval. *ACM Transactions on Information Systems*, 14(4):349–379, 1996.

[109] Alistair Moffat, Justin Zobel, and Ron Sacks-Davis. Memory efficient ranking. *Information Processing & Management*, 30(6):733–744, 1994.

[110] Sung Hyon Myaeng, Don-Hyun Jang, Mun-Seok Kim, and Zong-Cheol Zhoo. A flexible model for retrieval of SGML documents. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1998)*, pages 138–145, 1998.

[111] J. O'Connor. Answer-passage retrieval by text searching. *Journal of the American Society for Information Science*, 31(4):227–239, 1980/2007.

[112] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRand citation ranking: Bringing order to the Web. Technical report, Stanford InfoLab, 1999.

[113] Nils Pharo. The effect of granularity and order in XML element retrieval. *Information Processing & Management*, 44(5):1732–1740, 2008.

[114] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1998)*, pages 275–281, 1998.

[115] Martin Potthast, Benno Stein, and Robert Gerling. Automatic vandalism detection in Wikipedia. In *Proceedings of the Annual BCS-IRSG European Conference on Information Retrieval (ECIR2008)*, pages 663–668, 2008.

[116] Reid Priedhorsky, Jilin Chen, Shyong (Tony) K. Lam, Katherine Panciera, Loren Terveen, and John Riedl. Creating, destroying, and restoring value in Wikipedia. In *Proceedings of the International Conference on Supporting Group Work (Group2007)*, pages 259–268, 2007.

[117] Georgina Ramirez, Thijs Westerveld, and Arjen P. de Vries. Using small xml elements to support relevance. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2006)*, pages 693–694, 2006.

[118] Ian Roberts and Robert Gaizauskas. Evaluating passage retrieval approaches for question answering. In *Proceedings of the Annual BCS-IRSG European Conference on Information Retrieval (ECIR2004)*, pages 72 – 84, 2004.

[119] Stephen Robertson, Wei Lu, and Andrew MacFarlane. XML-structured documents: Retrievable units and inheritance. In *Proceedings of the 7th International Conference on Flexible Query Answering Systems, (FQAS2006)*, pages 121–132, 2006.

[120] Stephen Robertson, Hugo Zaragoza, and Michael Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of the 13th ACM Conference on Information and Knowledge Management (CIKM2004)*, pages 42–49, 2004.

[121] Stephen E. Robertson, Steve Walker, and Micheline Hancock-Beaulieu. Okapi at TREC-7: Automatic ad hoc, filtering, VLC and interactive track. In *The Seventh Text REtrieval Conference (TREC-7)*, pages 253–264, 1998.

[122] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, Mike Gatford, and A. Payne. Okapi at TREC-4. In *The Fourth Text REtrieval Conference (TREC-4)*, pages 73–96, 1996.

[123] Walter L. Ruzzo and Martin Tompa. A linear time algorithm for finding all maximal scoring subsequences. In *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology (ISMB1999)*, pages 234–241, 1999.

[124] Gerard Salton, Andrew Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[125] Ralf Schenkel, Fabian Suchanek, and Gjergji Kasneci.

[126] Börkur Sigurbjörnsson, Jaap Kamps, and Maarten de Rijke. An element-based approach to XML retrieval. In *INEX 2003 Workshop Proceedings*, pages 19–26, 2004.

[127] Börkur Sigurbjrnsson, Jaap Kamps, and Maarten de Rijke. Mixture models, overlap, and structural hints in XML element retrieval. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors, *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 196–210. Springer-Verlag, 2005.

[128] Amit Singhal, Chris Buckley, and Mander Mitra. Pivoted document length normalization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1996)*, pages 21–29, 1996.

[129] Koen Smets, Bart Goethals, and Brigitte Verdonk. Automatic vandalism detection in Wikipedia: Towards a machine learning approach. In *AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy (WIKIAI2008)*, pages 43–48, 2008.

[130] Michael Strube and Simone P. Ponzetto. WikiRelate! computing semantic relatedness using Wikipedia. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence(AAAI-06)*, pages 1419–1424, 2006.

[131] Stefanie Tellex, Boris Katz, Jimmy Lin, Aaron Fernandes, and Gregory Marton. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2003)*, pages 41–47, 2003.

[132] Andrew Trotman. Choosing document structure weights. *Information Processing & Management*, 41(2):243–264, 2005.

[133] Andrew Trotman and Shlomo Geva. Passage retrieval and other XML-retrieval tasks. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2006) Workshop on XML Element Retrieval Methodology*, 2006.

[134] Andrew Trotman and Mounia Lalmas. Why structural hints in queries do not help XML-retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2006)*, pages 711–712, 2006.

[135] Andrew Trotman and Börkur Sigurbjörnsson. Narrowed extended XPath I (NEXI). In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors, *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 16–40. Springer-Verlag, 2005.

[136] Anne-Marie Vercoustre, James A. Thom, Alexander Krumpholz, Ian Mathieson, Peter Wilkins, Mingfang Wu, Nick Craswell, and David Hawking. CSIRO INEX experiments:

XML search using PADRE. In *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 65–72, 2003.

[137] Jean-Noël Vittaut and Patrick Gallinari. Supervised and semi-supervised machine learning ranking. In Norbert Fuhr, Mounia Lalmas, and Andrew Trotman, editors, *Comparative Evaluation of XML Information Retrieval Systems*, volume 4518 of *LNCS*, Berlin, Heidelberg, 2007. Springer-Verlag.

[138] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of the 21st international ACM SIGIR conference on Research and development in information retrieval (SIGIR1998*, pages 315–323, 1998.

[139] Wikipedia. Wikipedia — Wikipedia, the free encyclopedia, 2009. [Online; accessed 25-January-2009].

[140] Hengzhi Wu, Gabriella Kazai, and Michael Taylor. Book search experiments: Investigating IR methods for the indexing and retrieval of books. In Mounia Lalmas, Andy MacFarlane, Stefan Rüger, Anastasios Tombros, Theodora Tsikrika, and Alexei Yavlinsky, editors, *Advances in Information Retrieval*, volume 4956 of *LNCS*, pages –. Springer-Verlag, 2008.

[141] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2001)*, pages 334–342, 2001.

[142] Junte Zhang and Jaap Kamps. Link detection in XML documents: What about repeated links. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2008) Workshop on Focused Retrieval*, pages 59–66, 2008.

[143] Justin Zobel, Alistair Moffat, Ross Wilkinson, and Ron Sacks-Davis. Efficient retrieval of partial documents. *Information Processing & Management*, 31(3):361–377, 1995.