# Leveraging Collection Structure in Information Retrieval With Applications to Search in Conversational Social Media

Jonathan L. Elsas

CMU-LTI-11-014
August 25

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

**Thesis Committee:**
Jaime G. Carbonell, chair
Jamie Callan
William Cohen
Susan Dumais, Microsoft Research

*Submitted in partial fulfillment of the requirements*
*for the degree of Doctor of Philosophy*
*in Language and Information Technologies.*

**Keywords:** Information Retrieval, Collection Structure, Social Media

## Abstract

Social media collections are becoming increasingly important in the everyday life of Internet users. Recent statistics show that sites hosting social media and community-generated content account for five of the top ten most visited websites in the United States [4], are visited regularly by a broad cross-section of Internet users [61, 67, 115] and host an enormous quantity of information [119, 48, 9]. The increasing importance and size of these collections requires that information retrieval systems pay special attention to these collections, and in particular pay attention to those aspects of social media collections that set them apart from the general web.

Social media collections are interesting and challenging from the perspective of information retrieval systems. These collections are dynamic, with content being constantly added, removed and modified. These collections are time-sensitive, with the most recently added content often viewed as the most significant. These collections are richly structured, with authorship information, often threading structure and higher-level topical classifications. Although this type of collection structure is frequently critical for comprehension, it is rarely exploited in retrieval algorithms.

This thesis investigates the hypothesis that we can improve retrieval performance in these collections by leveraging this type of structure. To evaluate this hypothesis, we present an exploration of search in several social media collections: blogs and online forums. We demonstrate the utility of leveraging collection structure in three different retrieval tasks: blog post search, blog feed search, and forum thread search. The techniques explored throughout these experiments include evaluating the representation granularity of collections of documents, and methods to incorporate content an author has written throughout the collection. Our results show that, although the retrieval tasks and techniques to leverage this type of collection structure are varied, in many cases substantial and significant retrieval quality improvements can be realized by leveraging this collection structure.

## Acknowledgments

I could have never completed this dissertation without the encouragement and support of my mentors, colleagues, friends, and family. First, my advisor Jaime Carbonell, has provided guidance not only through the experiments here, but also every aspect of graduate school. My committee members Jamie Callan and William Cohen have provided useful insight into the problems I've tacked in this thesis, and Susan Dumais has been helpful both in providing comments on this thesis and as my mentor while working as an intern at Microsoft Research.

My peers at CMU have been an enormous inspiration and help over the years and an invaluable source of feedback and creativity. My collaboration and friendship with Jaime Arguello has been a highlight of my time in graduate school, with countless lunchtime chats about our research and life beyond school. Collaborations with Matthew Bilotti and Vitor Carvalho have been both interesting and fruitful. Many, many others at CMU and elsewhere have helped me through this work; thank you all.

Without support from Lee Jensen and the rest of Ancestry.com a large portion of this work would not have been possible. Through their help and generosity we have been able to build and share with the research community a significant and useful dataset.

Finally, I would like to thank my family: my wife Joanna, my children Aiden, Zachary and Willa, my parents and my sisters. I cannot count the number of ways their patience and support has helped me through my graduate experience, and have been a daily refreshing reminder that life exists outside of graduate school.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Traditional approaches to information retrieval (IR) view document collections as sets of objects independent from each other. In that context, when a query is received by an IR system, a measure of similarity is calculated between the query and the document text, and the most similar documents are presented to the user. The view of independent documents has been challenged throughout the IR literature — dating back to Salton's work in cluster-based retrieval and suggestions that document quality could be inferred from citation networks [107]. But, it was not until the advent of the World Wide Web when hyperlinking between document became a critical aspect of document collections that IR systems began to take advantage of inter-document relations as a critical component of ranking algorithms. Document representations were enriched with "anchor text" from the links pointing to the document, and measures of document quality were developed based on the overall network linking structure or linking volume. This recognition that documents exist not as isolated objects in a collection, but interconnected entities, led to great improvements in retrieval performance and to the success of modern web-scale information retrieval systems [23, 93].

Document collections are currently in the process of evolving again. Social media collections offer rich and interesting organizational dimensions, providing information about relationships among documents beyond hyperlinks. Social media collections often contain contributions from many authors, providing document co-authorship relations in the collection. These collections often support discussion threads, with message-response relationships. Contributors to these collections often have the ability to assign tags or categories to documents, generating co-classification relationships among documents. Not only is this structural data readily available in social media collections, but it is often necessary for understanding the collection and navigating the content. In online forums, for example, individual messages often cannot be understood on their own, but rather need a conversational context from previous messages. Online forums frequently provide access to the data through browsing of a hierarchical topical organization of the message threads.

Over the past several years, user-generated content and particularly "social media" has become recognized as an important and growing source of content online. Longitudinal studies of online behavior from Pew Internet show that the percentage of Internet users using social media websites and consuming user generated content has steadily increased since 2004 [1]. Figure 1.1 shows these trends for several online activities, such as participation is a social network, reading blogs and posting comments on blogs, online forums or photo sites. Recent statistics from Alexa's Internet traffic rating service show that the top three blogging sites Blogger, Wordpress and Tumblr currently receive visits from approximately 14%, 5% and 2% of global Internet users respectively [4, 5, 6, 3]. This volume has been increasing over the past several years, as shown in Table 1.1, with Tumblr currently receiving more than 2.7 billion weekly page views.

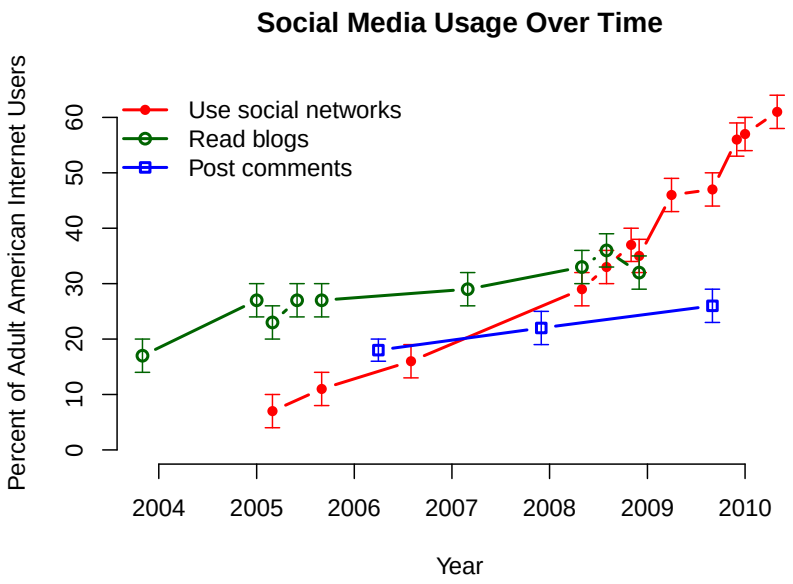



**Figure 1.1:** Growth of different types of activities on social media sites, shown as a percentage of adult American Internet users, based on Pew Internet & American Life Project survey data [1]. Percent refers to positive responses to the following questions: (Red) "Do you ever use online social or professional networking sites like Friendster or LinkedIn?", (Green) "Do you ever read someone else's online journal or blog?" and (Blue) "Do you ever post comments to an online news group, website, blog or photo site?".

| | Weekly Pageviews (in millions) | | | |
|---|---|---|---|---|
| **Blogging Site** | **July 2011 (Current)** | **January 2011** | **January 2010** | **January 2009** |
| Tumblr.com | 2,700 | 875 | 150 | — |
| Wordpress.com | 600 | 600 | 400 | 250 |

**Table 1.1:** Global weekly pageviews from two popular blog hosting sites, shown in millions of page views. Data is approximate and derived from statistics reported by Quantcast (Tumblr) [105] and Wordpress [10].

These social media collections offer new opportunities and challenges for information retrieval systems, in particular with respect to the rich and varied relationships among documents they provide. This thesis presents an exploration of search in several social media collections, with a specific focus on leveraging this type of structural collection information. We focus on two types of social media collections: blogs and online forums. We study three different retrieval tasks in these collections: blog post search, blog feed search and forum thread search. Through retrieval experiments, we demonstrate the utility of leveraging this collection structure in these tasks. Our results show that, although the retrieval tasks and techniques to leverage this type of collection structure are varied, in many cases substantial and significant retrieval quality improvements can be realized by leveraging this collection structure.

## 1.1 Conversational Social Media

Social media is described by the sociologist Marc Smith as "collective goods produced through computer-mediated collective action" [116]. This broad definition encompasses a wide range of media, document collections, and other resources. In the case of our work, we focus on social media where the *collective goods* are document archives and the *collective action* is dialog among contributors. We refer to these collections as *conversational social media*, and these collections are particularly interesting both because of their increasing importance in online life, as well as their distinguishing characteristics.

Social media sites host an enormous and growing volume of information. In addition to the increasing usage of these sites shown in Table 1.1, the blogging services Wordpress.com and Tumblr.com host more than 20 million blogs each and combined serve more than ten billion monthly pageviews [119, 9] [1]. The community question-answering site, Yahoo! Answers[2] has collected over 1 billion answers, and processes over 800,000 questions and answers per day [131]. Recently published statistics show that more than 400,000 new accounts per day are created on the micro-blogging site Twitter.com, and those users publish well over 100 million tweets per day [120]. A large recent study of popular and high-traffic online forums evaluated a collection of more than 3.5 million online forums sites with over 400 million messages written by more than 45 million authors [48].

In addition to the large volume of user-generated content, these content sources play a significant role in the life of Internet users. According to Alexa.com's Internet traffic rankings, five of the top ten most popular websites in the United States host user generated content or provide platforms for online social interaction [4]. According to the most recent Pew Internet surveys of American adult Internet users, 13% use Twitter [115], 53% use Wikipedia to look for information [138] and 47% use some social networking site such as Facebook[3] or MySpace[4] [67]. This usage volume has been increasing steadily since 2004, as shown in Figure 1.1. Major search engines have also recognized the value of social media and user generated content, incorporating content from users' immediate social network into search results [54, 89]. An example of this is shown in Figure 1.2, where results shared by an individual's social network are highlighted in the search results.

These statistics show that social media sites are truly becoming an integral part of online life. Like the advent and popularity of hypertext collections, social media collections' increasing importance and magnitude signals a shift in the landscape of document collections. Just as retrieval systems adapted to the popularity of hyperlinked document collections, retrieval systems must adapt to these collections, with an eye towards the aspects of these collections that distinguish them from traditionally studied collections.

We specifically look at blogs and online forums in this thesis. These collections are particularly appealing for several reasons. First, these sites are typically focused around content generation and information sharing, rather than purely social interactions as in a social networking site. As a results, they produce large archives of text data. Second, collections such as these may have been active for years and contain content content written over a long period of time. The Ancestry.com Forum described in Chapter 3 has been contributed to for over ten years, and as shown in Figure 1.1 blogs have been actively read by a significant portion of the Internet population for at least the last six years. Finally, these collections are readily available. The Text REtrieval Conference (TREC) has supported experimentation with blog collections since 2006 [78, 100].

[1] These two blogging services are considerably less popular than the Google blogging service `blogspot.com`, but that service does not publish usage statistics.

[2] `http://answers.yahoo.com`

[3] `http://facebook.com`

[4] `http://myspace.com`

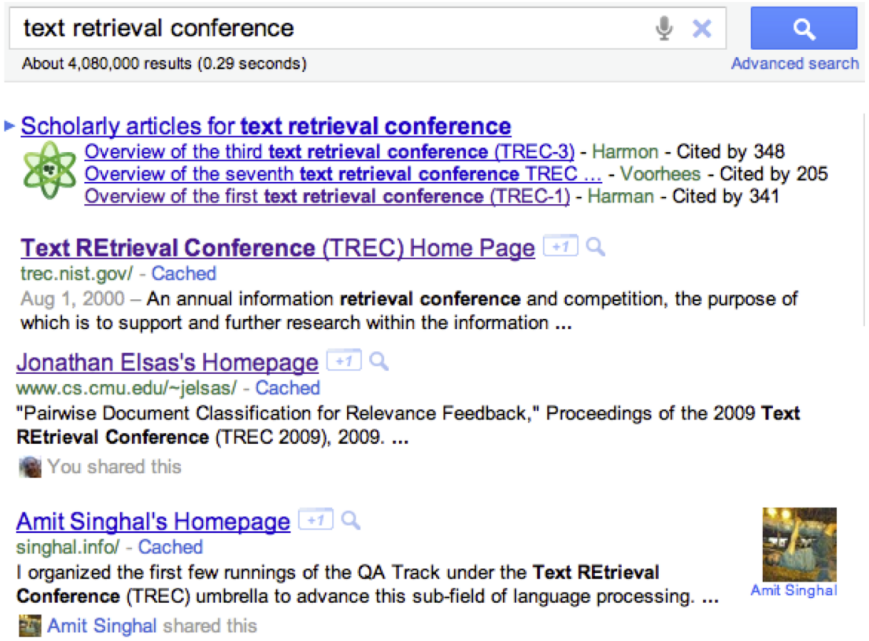



**Figure 1.2:** Search results for the query *[text retrieval conference]* on a major web search engine. Documents shared by the logged-in user's immediate social network are indicated in these results.

### 1.1.1 Distinguishing Properties of Conversational Social Media

With the increasing size and importance of these collections online, the need to effectively provide access is crucial. These collections have several distinguishing characteristics when compared to traditionally studied IR test collections, which provide both challenges and opportunities to information retrieval system designers.

- **Authorship Information**: These collections are often contributed to by many authors and this authorship information is readily available as document metadata. The archives of text generated by individuals opens the opportunity for IR systems to integrate reputation, authority and expertise measures into document ranking algorithms.
- **Message-Response Structure**: These collections often support directly replying to messages via blog commenting systems or discussion threads. The conversational structure of an online forum thread is crucial for understanding the context and meaning of messages within the thread.
- **Multiple Organizational Dimensions**: With the rich metadata present in the collections, there are many organizational axes along which to view the collections. Many axes of organization provide different retrievable objects, for example author retrieval akin to expert search tasks [12], thread retrieval in online forums [46], or even *forum* retrieval to identify communities of experts [48].
- **Temporal**: These collections often have a temporal nature, with the creation and update timestamps of each document being recorded in the metadata. Temporal features are vitally important in real-time micro-blog systems such as Twitter which primarily ranks search results by time. Temporal features have been shown to be useful for blog post retrieval as well [91].
- **Dynamic**: As user-generated content, these collections are continually growing and changing. New documents are constantly added to the collection, as well as old documents being edited and re-

moved. These features of social media collections make maintaining an IR index of the documents a challenging task.

- **Topical**: As opposed to general web or news collections, many social media collections have a strong topical focus both at the collection level and at the sub-collection level. Messages in an online forum thread typically relate to the same topic, often anchored by a question posed in the first message of the thread. Additionally, individual online forums sites tend to focus on a single topic, supporting focused discussion within communities of experts and enthusiasts.
- **Spam**: The ease of contributing content to these collections leaves open the opportunity for malicious use of content- or link-generation mechanisms in the collection.

These distinguishing characteristics of conversational social media offer retrieval systems interesting challenges and opportunities when providing access to the data archives. In this work, we focus on the first three of these aspects: authorship information, message-response structure and threading information, and multiple organizational dimensions.

### 1.1.2 Search Tasks in Social Media Collections

The rich structure present in social media collections opens the door for a variety of search tasks. In this section, we look more closely at online forums as an example social media collection, and describe the variety of potential search tasks relating to these collections.

Online forums or message boards are document collections organized around *message threads*. These message threads are single dialogs between different users on an online forum site, each providing one or more messages in response to those previously written. One author starts a thread, creating the first message, and frequently online forums provide the ability for this author to select a category from a hierarchy to assign to the thread. These categories form "sub-forums" within the online forum site, allowing users to focus their attention on portions of the collection containing discussion on topics most interesting to them. Browsing online forum archives often is centered around navigation of this sub-forum hierarchy. Many online forums sites are actively visited, contributed to and maintained, and each forum site typically focuses on a specific topic such as a computer hardware manufacturer[5], genealogical research[6] or gardening[7]. See Chapter 3 for a detailed description of online forum collections and specific examples.

One dimension to characterize a search task is the unit of retrieval expected by the searcher. In traditional ad hoc retrieval, the search task is based on retrieval of single documents. In conversational social media collections, each of the collection's organizational dimensions provide a distinct potential search task. Several interesting search tasks present themselves when considering the richly structured nature of online forums:

- **Thread Search**: Online forum sites are generally organized around the message thread as the primary unit of consumption, and for this reason it is an obvious candidate for search tasks in these collections. We present experiments on thread search in online forums in Chapter 6.
- **Sub-forum Search**: Any node in the hierarchical organization of the online forum site may be a reasonable result to return for users seeking out a appropriate place to ask a specific question.
- **Forum Discovery**: Non-expert users may be seeking out expert communities focusing on particular topics, without any knowledge of the existence of an online forum. In this case, retrieving the entire

[5] http://forums.macrumors.com/
[6] http://boards.ancestry.com/
[7] http://forums.gardenweb.com/forums/

forum site may be an appropriate course of action. This approach was taken in previous work identifying online forums useful for discussing consumer electronic products [48].

- **Author Search**: Many forum sites provide the ability to send private messages to individuals. A novice user many want to use this facility to directly ask an expert a question, rather than post a open question to the entire forum user base. In this case, providing facilities for author, or expert search in online forums may be particularly useful.

These are just a few of the retrieval tasks applicable to online forum collections. Some, like author search, are similar to other tasks previously studied in the IR literature [11, 40, 117]. But, collections like online forums offer the opportunity to study a huge variety of tasks in the same collection, and use structural dimensions across tasks in ways that have not yet been explored.

## 1.2 The Goal and Contribution of this Thesis

This thesis presents an exploration of search in user-generated conversational social media collections, with a specific focus on leveraging these relationships among documents while searching. We focus on two types of social media collections, blog and online forums, and three different search tasks, blog post search, blog feed search, and online forum thread search. Our work specifically explores the utility of modeling the relationship among groups of elements being retrieved, such as posts in a blog or messages in a thread, as well as methods to integrate content from broader collection structure into the ranking algorithm. With respect to thread search, for example, we find that in one of our collections thread retrieval performance on the hardest queries in our test set is significantly improved when integrating author content from throughout the collection.

The research questions we address in this thesis are as follows:

1. What methods are effective for ranking aggregates of documents in conversational social media collections, such as blog feeds or message threads?
2. When author information is present in the collection, to what degree can we improve retrieval performance by incorporating this evidence into message scoring?
3. When a higher-level document organization is present in the collection, to what degree can we improve retrieval performance by incorporating evidence from that organization?

We find that various retrieval techniques behave differently on the different tasks and collections evaluated here. In blog feed search, we find that treating the blog feed as a single document is an effective technique, but can be outperformed by modeling the blog posts individually and differentially weighting posts closer to the "central topic" of the blog. These results, presented in Chapter 5, demonstrate that an effective method of modeling relevance at the blog post level and aggregation of that evidence up to the feed can result in a 9% improvement over methods that model the blog feed as a single document. In blog post search, we find that relying too much on the language of the blog as a whole risks overly favoring prolific bloggers at the expense of bloggers who write a few relevant posts. Chapter 4 explores this risk-reward tradeoff, and presents a simple model of using evidence at the blog-level to mitigate this risk, achieving a 6.5% improvement in MAP over a model that does not take into blogger-level evidence. In thread search in online forums, we find that including an author language modeling component to the message scoring can significantly improve performance on the long and multi-faceted queries. The experiments in Chapter 6 shows that expanding or smoothing messages with other content an author has written can result in a a significant 10% improvement on these queries over models based only on the content of the thread.

## 1.3 Reader's Guide

- In **Chapter 2** we give an overview of language modeling for information retrieval, a general overview of models for aggregate document search, and previous approaches to search in social media collections. Previous work relating to the specific tasks we investigate is distributed throughout the thesis.
- In **Chapter 3** we present the collections studied in this thesis. First, we briefly discuss the BLOG06 test collection used for both the blog post and blog feed search experiments. Next, we analyze two online forum document collections. In the first of these collections, the MacRumors.com Forum, we present a novel method for identifying queries and relevant message threads for use as an information retrieval test collection. We then discuss the creation of an IR test collection with the Ancestry.com Forum. To our knowledge, this is the largest online forum collection studied for the purposes of information retrieval research.
- In **Chapter 4** we discuss experiments relating to blog post search, with particular attention paid to utilizing the language of the blog as a whole when ranking posts. We apply methods previously proposed for smoothing structured documents as well as general score-combination models to the tasks of integrating blogger content into blog post ranking.
- In **Chapter 5** we discuss work on blog feed search, the task of retrieving blogs, rather than blog posts. We contrast methods of treating the blog feed as a single large-document to methods that treat the feed as a collection of individual documents, the posts. We present a novel feed search model, capable of differentially weighting posts within a blog, as well as investigate the role of feed size in ranking.
- In **Chapter 6** we present experiments with the online forum test collections. We first study methods for effective thread retrieval, in particular considering methods to aggregate a message ranking into a thread ranking. We then consider method to integrate structural information from throughout the collection into the thread ranking, with a specific focus on content the author has written.
- In **Chapter 7** we summarize our contributions, and describe avenues for future work building on this thesis.

# Chapter 2

# Preliminaries and Related Work

This chapter presents a discussion of background material on information retrieval and high-level discussion of related work upon which the experiments in this thesis build. Related work specific to the tasks studied in this thesis are discussed in the corresponding chapters when appropriate.

First, we give a brief overview of language modeling for information retrieval, and specifically focus on several of the retrieval models we use throughout this thesis. Next, we discuss two ways collection structure has been used in previous work — ranking *document aggregates*, or collections of documents, and using document aggregate information when ranking documents. We then describe a variety of tasks that have been studied in the past involving ranking of document aggregates and using aggregate structure. Finally, we will present work relating to search in social media collections, including general approach to search in blogs and other social media.

Table 2.1 outlines the notational conventions used throughout this thesis.

## 2.1 Language Modeling for Information Retrieval

Statistical language models [104] were proposed as a more principled alternative to TF-IDF vector space [28] and other probabilistic models [60], and subsequent work in language modeling is the foundation for many of the ranking algorithms used here [134]. The language modeling approach allows for the combination of a *document language model*, representing the distribution of terms present in the document, and a *background language model*, representing the underlying distribution of terms in general language or the document collection. In this framework, the task of scoring documents with respect to the query is interpreted as estimating a probability $P(D|Q)$, applying Bayes's rule:

$$\begin{aligned} P(D|Q) &= P(Q|D)P(D)/P(Q) \\ &\stackrel{\text{rank}}{=} P(Q|D)P(D) \end{aligned}$$

where we can ignore the probability $P(Q)$ which does not affect the resulting document ranking. The document prior term $P(D)$ provides a means to incorporate query-independent evidence into the document scoring, such as general document quality features. The estimation of the *query likelihood* term $P(Q|D)$ is the focus of a large body of information retrieval research. In this model, typically the query terms $q_i \in Q$ are assumed to be independent, and query likelihood is calculated as:

$$P(Q|D) = \prod_{q \in Q} P(q|D)^{n(q,Q)} \stackrel{\text{rank}}{=} \sum_{q \in Q} n(q,Q) \log P(q|D)$$

| Symbol | Meaning |
|---|---|
| $\overset{\text{rank}}{=}$ | Rank equivalence, i.e. $f(x) \overset{\text{rank}}{=} g(x)$ iff $f = h \circ g$, where $h$ is some monotonically increasing function. |
| $\mathbf{I}(\bullet)$ | Indicator function, $\mathbf{I}(\bullet) = 1$ when $\bullet$ is true, 0 otherwise. |
| $D, P, M, T, S$ | Document, Post, Message, Thread, Subforum. |
| $Q$ | Query. |
| $C$ | Document Collection. |
| $t \in D$ | Terms $t$ in document $D$. |
| $q \in Q, \psi \in Q$ | Terms $q$ or query features $\psi$ in query $Q$. |
| $n(t, D)$ | Term Frequency. Number of occurrences of term $t$ in document $D$. |
| $\|D\|$ | Length of document $D$, i.e. $\|D\| = \sum_{t \in D} n(t, D)$ |
| $b(P)$ | Blog containing Post $P$. |
| $a(M)$ | Author of Message $M$. |
| $t(M)$ | Thread containing Message $M$. |
| $s(T)$ | Subforum containing Thread $T$. |
| $N$ | Number of documents in the collection. |
| $N_X$ | Number of documents in sub-collection $X$ (eg. $N_T$ is number of messages in thread $T$). |
| $P(t\|D)$ | Likelihood of observing term $t$ in document $D$. |
| $P(D)$ | Document prior. |
| $\alpha, \lambda, \beta, \mu$ | Model parameters. |
| $\mathbf{A}, \mathbf{y}$ | Matrix (capital letters), vector (lowercase letters). |
| $\mathcal{R}_k(Q)$ | Set of retrieved documents up to rank $k$ for query $Q$. |
| $\rho_Q(D)$ | Rank of document $D$ for query $Q$. |

**Table 2.1:** Notational conventions used throughout this thesis.

where $n(q, Q)$ is the number of times the query term $q$ occurs in the query $Q$, and $Q$ and $D$ refer to the query and document language models respectively.

The effective and efficient estimation of the term generation likelihood from the document language model, $P(q|D)$ has received much attention in the IR research literature. These probabilities are typically estimated with some form of *smoothing*, which achieves two goals: (1) smoothing places some probability mass on non-observed terms in the document, eliminating zero probabilities and generally improving the maximum likelihood estimate, and (2) smoothing down-weights common and non-discriminative terms, functioning in a similar way to Inverse Document Frequency (IDF) in the vector space models. Zhai and Lafferty's work investigates several smoothing methods providing theoretical motivations and extensive experiments [134]. The most popular of the smoothing methods studied are Dirichlet smoothing and Jelinek-Mercer (JM) smoothing:

$$P_{dir}(q|D) = \frac{n(q, D) + \mu P(q|C)}{|D| + \mu} \qquad \text{Dirichlet Smoothing} \tag{2.1}$$

$$P_{jm}(q|D) = \lambda \frac{n(q, D)}{|D|} + (1 - \lambda) P(q|C) \qquad \text{Jelenick-Mercer Smoothing} \tag{2.2}$$

where $C$ represents the collection language model and $P(q|C)$ is estimated via the maximum likelihood $P(q|C) = \frac{n(q,C)}{|C|}$. The smoothing parameters $\lambda$ and $\mu$ provide the means for the retrieval system designer

to adjust the amount of collection-level evidence to include in these ranking formulae. These parameters are typically trained on previous examples of queries and relevant documents in order to maximize some retrieval performance measure such as Mean Average Precision (MAP).

Dirichlet smoothing typically outperforms the Jelinek-Mercer smoothing, and this is largely attributed to the Dirichlet model's sensitivity to the document length $|D|$. Dirichlet smoothing incorporates more collection-level evidence when documents are short and more effectively normalizes scores across documents of differing length [134, 76]. In order to adjust for the less effective length correction in Jelinek-Mercer smoothing, document length priors are often employed when using that smoothing model [95, 76], frequently exponential and favoring longer documents:

$$P(D) \propto |D|^{\beta}$$

where $\beta$ is a parameter to be learned from training data.

### 2.1.1 Term Dependencies in the Query Likelihood Model

Work by Metzler and Croft relates the query likelihood estimation to estimating parameters in a Markov Random Field (MRF) model, combining term unigram, n-gram and term window features in the ranking function [87, 86]. This model, dubbed the *Term Dependence Model* (DM), is given by the following ranking formula:

$$\begin{aligned} P_{DM}(Q|D) &= \prod_i P(\psi_i(Q)|D)^{w_i} \\ &\stackrel{\text{rank}}{=} \sum_i w_i \log P(\psi_i(Q)|D) \end{aligned}$$

where the term feature functions $\psi_i$ describe sets of query features incorporating the term proximity and ordering information present in the query and the weights $w_i$ weigh these feature sets appropriately. The probabilities $P(\psi_i(Q)|D)$ can be calculated as smoothed probabilities above, with either Dirichlet or Jelinek-Mercer smoothing, using feature-counts in the document and query rather than term-counts. When dealing with term unigram ($T$), ordered window ($O$) and unordered window ($U$) query features, this ranking function becomes:

$$P_{DM}(Q|D) \stackrel{\text{rank}}{=} w_T \sum_{c \in T} \log P(c|D) + w_O \sum_{c \in O} \log P(c|D) + w_U \sum_{c \in O \cup U} \log P(c|D) \tag{2.3}$$

with three weights and three corresponding sets of features ($T$, $O$ and $U$). Throughout the thesis experiments presented in the following chapters, when applying this dependence model we use the *Full Dependence* formulation which takes into account query term features beyond sequential bigrams. We also use the weights $w_i$ proposed by Metzler et al., which have been shown to be effective across a wide range of collections and tasks, $w_T = 0.8, w_O = w_U = 0.1$ [86].

### 2.1.2 Structural Evidence in the Query Likelihood Model

The language modeling approaches above estimate the likelihood of observing a term or query feature given the document and collection contexts. In many retrieval tasks, additional context is available either from document structure such as annotations or from collection structure as studied in this thesis. When a document (or passage) being ranked closely relates to this additional context, such as a sentence within an article or a blog post within a blog, query term matches in the context may be positive evidence that

the document (or passage) is relevant. These smoothing models act as a form of document expansion, including content from the background language model or other sources into the document's language model. By doing this, the ranking model can capture query similarity within the document and the related context.

The smoothing models above have been extended to consider these cases, and primarily applied to tasks such as passage- or extent-retrieval which aim to retrieve a portion of a document rather than an entire document. In these tasks, the retrievable unit $E$ is a portion of a document, thus in addition to smoothing with the collection language model, we may choose to incorporate evidence from the document containing the units to be retrieved.

A natural extension to the Jelinek-Mercer smoothing model, Equation 2.2, that incorporates additional levels of document context adds an additional model into the mixture:

$$P_{jm2}(q|E) = \lambda_E P(q|E) + \lambda_A P(q|D) + (1 - \lambda_E - \lambda_A) P(q|C) \qquad \text{Two-level JM Smoothing} \tag{2.4}$$

where we are retrieving an extent (or passage) $E$ and two smoothing parameters control the degree to which evidence from the context ($A$) and collection ($C$) are included in the estimation. As the weight on the context decreases, $\lambda_A \to 0$, the amount of context-level evidence is diminished this model approaches the one-level Jelinek-Mercer smoothing model. Mixture models of this form have been studied extensively by Ogilvie in tasks such as sentence retrieval for Question Answering an XML element retrieval [95].

Extending the Dirichlet smoothing model, Equation 2.1, to handle multiple levels of context has been studied by Zhao and Callan [137], who also apply this model to several extent retrieval tasks. This model is derived by first apply Dirichlet smoothing to the document language model $D$, then smoothing the extent $E$ with that already-smoothed document language model. The mathematical formulation is given as:

$$\begin{aligned} P_{dir2}(q|E) =& \frac{n(q,E) + \mu_D P_{dir}(q|D)}{|E| + \mu_D} \\ =& \frac{n(q,E) + \mu_D \frac{n(q,D) + \mu_C P(q|C)}{|D| + \mu_C}}{|E| + \mu_D} \end{aligned} \qquad \text{Two-level Dirichlet Smoothing} \tag{2.5}$$

where, again, the two smoothing parameters $\mu_D$ and $\mu_C$ control how much document- and collection-level evidence is included in the term generation likelihood. As $\mu_C \to \infty$, the two-level Dirichlet smoothing model approaches the single-level smoothing, with a decreasing amount of document level smoothing.

The two-level Jelinek-Mercer smoothing model suffers from similar biases as the one-level model, namely favoring short documents. While this can be corrected for by applying a document length prior [95], the two-level Dirichlet model attempts to correct the length bias within the smoothing model. Zhao and Callan present an analytical comparison of the two smoothing models and show that the two-level Jelinek-Mercer smoothing model has a strong bias for short documents (or document fields), whereas the two-level Dirichlet model does not [137]. For this reason, we generally prefer the two-level Dirichlet model in our experiments below.

Although these smoothing models have primarily been applied to sub-document retrieval tasks, they are equally applicable to document ranking tasks where the document itself belongs to an intermediate context. In these tasks, such as federated search or cluster-based retrieval, documents belong to sub-collections, and two-level smoothing models are a natural fit. These models can be employed to smooth a document with related documents, and then with the collection. Several other tasks, such as cluster-based retrieval [71] and email message retrieval [103], described in more detail below, have employed these two-level smoothing models as well.

The smoothing methods presented here incorporate two levels of context, but in some cases multiple levels of organization may be present in a collection. Ogilvie and Callan [97] present a smoothing method that takes advantage of multiple levels of nesting in structured XML documents, first smoothing up the element tree from the leaves to the root, and then smooth down from the root to the leaf elements. A similar model could be applied to collection structure with multiple levels of nesting, although we do not simultaneously consider multiple levels of nesting in the work presented below.

## 2.2 Feature-Based Retrieval Models

While most of this thesis focuses on applications of the Language Modeling retrieval framework, we also make some use of general feature-based retrieval models. These retrieval models combine a large number of document features to produce a document score:

$$Score(D, Q) = h(\mathbf{x}^{(D;Q)}; \mathbf{a})$$

where $h$ is an scoring function, $\mathbf{x}^{(D;Q)}$ is a vector of real-valued feature functions calculated over the document $D$ and query $Q$:

$$\mathbf{x}^{(D;Q)} = \langle f_1(D, Q), f_2(D, Q), \ldots, f_n(D, Q) \rangle$$

and $\mathbf{a}$ is a parameter vector to be learned from training data. This scoring function $h$ can take the form of any function over vectors, $h : \Re^n \rightarrow \Re$, and state-of-the-art ranking algorithms typically apply highly non-linear functions such as neural networks or boosted decision trees [25]. Identifying effective weights $\mathbf{a}$, the "Learning to Rank" problem, is a growing subfield of IR and machine learning. Learning-to-Rank algorithms are out of the scope of this thesis, and the reader is referred to Liu's tutorial on the subject for background [70].

Often, as in the case of our work below, a straightforward linear combination (dot-product) of the document-query features and the learned weights is applied:

$$Score(D, Q) = \mathbf{a} \cdot \mathbf{x}^{(D;Q)} = \sum_i a_i f_i(D, Q)$$

Some language modeling retrieval functions fall into this category of linear feature-based rankers, for example the Dependence Model (Equation 2.3) is a linear combination of three log-probabilities, each representing features over the unigram, ordered and un-ordered window features of the query-document pair. However, feature-based retrieval functions tend to be more flexible than language modeling approaches, allowing negative feature weights and easier incorporation of many features into a single function.

## 2.3 Document Aggregates

This thesis is concerned with collection structure, and often this structure takes the form of defining groups or partitions of documents within the collection. We refer to these groups of documents as *document aggregates*, and ranking of these aggregates has been a integral part of several widely studied IR tasks. While these tasks and the structures that define the aggregates differ in many ways, the techniques applied are often similar. We distinguish the tasks of *ranking document aggregates* and *aggregated search* — the former typically deals with ranking subsets of homogeneous document types, while the latter is concerned with selecting and combining results from multiple heterogeneous search engines in a single results display.

Below, we describe several of the dimensions along which ranking of document aggregates can be characterized:

- **Implicit vs. Explicit structure**. Document aggregates may be defined explicitly, such as through document metadata like authorship information, or implicitly through some measure of text similarity. We are primarily concerned with explicit definitions of collection structure in this thesis, but similar techniques have been applied to both types of structure.
- **Unit of Consumption**. Some document aggregates are best consumed as single text objects, such as an online forum message thread. Without the context from previous messages and subsequent responses, a single message may not be understandable. Other document aggregates are not intended to be consumed as a single unit. For example, blog posts belong to a blog, but are typically intended to stand on their own as a single document.
- **Retrieval Task Output**. Document aggregate ranking may be the final goal of the search task, as in blog feed search, or may be an intermediate step of a document ranking task. In the latter case, the goal of document aggregate ranking is improving the performance or efficiency of a final document ranking. Resource selection in federated search is a typical example of this type of aggregate document ranking. Note that, as in the case of blog feed search, the retrieval task output is not necessarily the unit of consumption.

Several research themes have repeatedly arisen in work on document aggregate ranking. In this section we describe a few of the dominant themes in this line of research.

### Aggregate Representation Granularity

Retrieval algorithms use document features such as query term counts and document length when scoring a document. When scoring document aggregates, these features could be derived from each element in the aggregate individually, or to derived from the aggregate as a whole such as through a concatenation of the documents. These two approaches differ in the granularity of representation of the aggregate — whether to represent the aggregate as a single document or as a collection — and is a recurring theme in the development of aggregate document ranking algorithms. We refer to the two general approaches to aggregate document representation as *small document* and *large document* models.

Large-document models tend to be easy to implement with existing retrieval tools and many leverage well-understood mechanisms for balancing term specificity and document length. But, large-document models have several drawbacks:

1. Some, such as document concatenation models, do not allow for re-weighting or selection of documents within the aggregate.
2. Although they use document length, many do not take into account *aggregate size* in the ranking algorithm.
3. Large-document representations may be difficult to update when new documents are added to the system.

While these drawbacks apply to many early large document models such as the concatenation model (Equation 2.9), they can be addressed by others. For example Balog's Model 1 (Equation 2.7) uses fractional term counts to balance across documents, weighing by an association factor to favor some documents over others [12].

Small-document models address these criticisms directly, and generally allow a more flexible aggregate document scoring approach. But, while small-document models may be appealing in their flexibility, they are not always the most effective choice of retrieval algorithm. Large- and small-document models have been compared in many aggregate document ranking tasks, such as resource selection in federated search [114, 27], cluster-based retrieval [71, 72], expert search [13, 12], blog feed search [15, 45] and

online forum thread search [46]. These tasks are discussed in more detail below.

#### Normalization of Aggregate Size

The numbers of documents contained within an aggregate is rarely constant, and for this reason normalization for aggregate sizes is often appropriate. This normalization must take into account the length biases inherent in the scoring function, but also the utility of large or small aggregates in the results set. For some tasks, such as resource selection in federated search, favoring large aggregates is generally beneficial — these collections are more likely to contain relevant documents than small collections [114]. In other tasks such as blog post search (described in Chapter 4), there is considerable risk in favoring prolific bloggers.

#### Selection and Weighting of Elements

As mentioned above, one of the benefits of small-document models is their ability to select or re-weight documents within the aggregate. This technique has proven effective in several aggregate ranking tasks. We explore several methods of weighting blog posts when ranking blog feeds in Chapter 5. Selection of elements for cluster representation [73], blog feed representation [110] and message thread representation [108] has proven to be an effective technique.

## 2.4 Methods and Models for Ranking Document Aggregates

A variety of models have been applied to document aggregate ranking. In this section we give an overview of these models, and highlight how these models have addressed the research themes described above. In an effort to make the following model descriptions, we refer to a general document aggregate $A$, containing some number of documents $N_A$.

### 2.4.1 Resource Ranking Methods

Federated search, or distributed IR, aims to retrieve documents from several collections rather than a single centralized index. One step in this process is the ranking and selection of collections most likely to contain relevant documents. A variety of methods have been proposed for this task, representing the collections as large pseudo-documents and using vector-space or language modeling approaches to measure the similarity of the query and collection [27, 26].

#### Si's Relevance Estimation Model

Si and Callan's Relevant Document Distribution Estimation (ReDDE) [114] is a notable departure from previous resource ranking methods for several reasons. First, it take a small-document approach, modeling the relevance of documents sampled from collections individually and aggregating that document-level relevance into collection-level relevance. Second, it explicitly takes into account the collection sizes, favoring collections with a large number of documents in the ranking.

This algorithm is based on the assumption that the collection statistics may not be readily available, and a sampling process must be performed to estimate the those statistics for each of the databases. ReDDE ranks collections based on their estimated number of relevant documents, which is calculated as follows:

$$Score(A) = \sum_{D \in A} P(rel|D)P(D|A)N_A \qquad \text{ReDDE} \tag{2.6}$$

where $P(rel|D)$ is a thresholded approximation of the probability of relevance of document $D$, $N_A$ is the number of documents in the collection $A$, and $P(D|A)$ is the likelihood of sampling document $D$ from the collection. Thus, this scoring formula models the relevance of individual elements within the sub-collection in order to estimate the relevance of the sub-collection. The ReDDE model, and others based on aggregating document scores have been shown to be superior to the "large document" approach used by previously proposed resource ranking methods.

### 2.4.2 Language Modeling

A variety of language modeling approaches to aggregate document ranking have been proposed, and many of these directly address the research themes outlined above. This section presents several notable examples from the literature.

#### Balog's Model 1 and Model 2

Balog et al. [12] develop two probabilistic retrieval models for expert search, which have also been applied to blog feed search. These two generative models, referred to as Model 1 and Model 2, directly address the issue of representation. These models both make use of an probabilistic association between documents and aggregates, $P(D|A)$, which allows for the differential weighting of documents within the aggregate. Neither of these models take into account aggregate size in the scoring.

Model 1 takes a large-document approach, representing the document aggregate as additive combination of the term occurrences in the associated documents and calculating the query score at the aggregate level:

$$P(Q|A) = \prod_{q \in Q} \left(\lambda P(q|A) + (1-\lambda)P(q|C)\right)^{n(q,Q)} \qquad \text{Model 1} \tag{2.7}$$

In this model, the query term likelihood is defined by

$$P(q|A) = \sum_D P(q|D)P(D|A) = \sum_D \frac{n(q,D)}{|D|} P(D|A)$$

which counts fractional term occurrences for each document based on their respective lengths, and weights contributions from documents by their strength of association with the aggregate $P(D|A)$. This model uses with Jelinek-Mercer smoothing to combine collection-level evidence at the aggregate-level. Although this model is considered a large-document retrieval model, the document language model is balanced across documents in order to avoid a single long document dominating the aggregate.

Model 2, on the other hand, calculates a query score for each document and combines those scores to form the aggregate score, thereby taking a small-document approach. Taking a similar approach to the ReDDE model (Equation 2.6), this model is given by:

$$\begin{aligned} P(Q|A) &= \sum_D P(Q|D)P(D|A) \\ &= \sum_D \left( \prod_{q \in Q} \left(\lambda P(q|D) + (1-\lambda)P(q|C)\right)^{n(q,Q)} \right) P(D|A) \qquad \text{Model 2} \end{aligned} \tag{2.8}$$

In this case, the query term likelihoods are calculated and smoothed at the document level.

Balog et al. also explore a variety of task-specific methods for estimating association between documents and aggregates $P(D|A)$. In the context of expert search, the level of association between a document and candidate expert is not clean-cut, but generally based on mentions of an expert in the document. The authors present several methods, including boolean models of association, models that incorporate the number of times a reference to an expert occurs in the document, and methods that weight by IDF-like factors to favor less frequent experts [14]. In the case of blog feed search where there is an unambiguous relationship between a blog post and blog feed, the authors use a uniform model of association [15].

### Liu's Cluster Representation Models

Liu and Croft explore a variety of aggregate ranking models, primarily applied to cluster-based retrieval [71, 73, 72]. In their work, several document clustering algorithms are explored, and given the output of those methods a cluster ranking model can be applied. A discussion of document clustering algorithms is out of the scope of this thesis, and we present the cluster ranking models below assuming the output of a suitable partitioning clustering method.

All of the Liu and Croft cluster ranking models rank document aggregates by the query likelihood $P(Q|A)$, and like Balog's models, none of these explicitly take into account aggregate size. The major models studied are:

- **Document Concatenation**. Aggregates represented by a pseudo-document, summing term counts across all documents in the cluster. This is a straightforward large-document model, not attempting to control for varying document lengths. Aggregates scored by

$$P(Q|A) = \prod_{q \in Q} \left( \lambda \frac{\sum_{D \in A} n(q, D)}{\sum_{D \in A} |D|} + (1 - \lambda) P(q|C) \right)^{n(q,Q)} \tag{2.9}$$

- **Term Frequency Mixture**. Like Balog's Model 1 (Equation 2.7), pseudo-documents are used to represent aggregates, and aggregate's representation is balanced across documents. The scoring is give by:

$$\begin{aligned} P(Q|A) &= \prod_{q \in Q} \left( P(q|A) \right)^{n(q,Q)} \\ &= \prod_{q \in Q} \left( \lambda \frac{n(q, A)}{|A|} + (1 - \lambda) P(q|C) \right)^{n(q,Q)} \end{aligned}$$

  where the cluster term counts are calculated by

$$n(q, A) = \sum_{D_i \in A} \alpha_i n(q, D)$$

  and the aggregate length is $|A| = \sum_t n(t, A)$. The authors suggest setting document weights $\alpha_i, \sum \alpha_i = 1$ proportional to the document's query generation likelihood $P(Q|D)$.

- **Document Model Mixture**. Aggregates scored by a mixture of query-likelihood scores over documents:

$$P(Q|A) = \sum_{D_i \in A} \beta_i P(Q|D_i)$$

  where, again, the weights $\beta_i, \sum \beta_i = 1$ are assigned proportionally to the document's query likelihood score. This model is almost identical to Balog's Model 2 (Equation 2.8), assuming $\beta_i = P(D_i|A)$.

- **Best/Worst Document**. Aggregates scored by

$$P(Q|A) = \max_{D \in A} P(Q|D) \qquad \text{Best Document Model}$$

or

$$P(Q|A) = \min_{D \in A} P(Q|D) \qquad \text{Worst Document Model}$$

The Best Document model is identical to the Geometric Mean model (below) when $k = 1$.

- **Geometric Mean / Pseudo Cluster Selection (PCS)**. Aggregates scored by a log-linear combination of the top-$k$ document query likelihoods with respect to the query. Letting $R_A = A \cap \mathcal{R}(Q)$ be the set of retrieved documents for the aggregate $A$ and $D_{\min} = \arg\min_{D \in \mathcal{R}(Q)} P(Q|D)$ be the minimum-scoring document for the query:

$$P(Q|A) = \begin{cases} \left( \prod_{D \in \text{top-}k \text{ retrieved from } R_A} P(Q|D) \right)^{\frac{1}{k}} & \text{if } |R_A| \geq k \\ \left( P(Q|D_{\min})^{k-|R_A|} \prod_{D \in R_A} P(Q|D) \right)^{\frac{1}{k}} & \text{otherwise} \end{cases} \qquad \text{PCS} \tag{2.10}$$

where the product is taken over the top-$k$ scoring documents in the aggregate $A$, padding with the minimum-scoring document for the query if $|A \cap \mathcal{R}(Q)| < k$. When $k = 1$, this model is equivalent to the Best Document model above. This model is also known as the Pseudo Cluster Selection Model (PCS).

Although none of these models *explicitly* control for aggregate size, several of the models *implicitly* have controls. The Best Document, Worst Document and PCS models always score an aggregate based on the same number of associated documents — Best and Worst Document methods always use one, and PCS uses $k$ documents. The PCS model has the tendency to dis-favor small aggregates, particularly those with less than $k$ documents that are padded by the worst-scoring retrieved document. In contrast, methods such as the Document Mixture have a tendency to favor short aggregates with a small number of high-scoring documents.

Liu and Croft find that for cluster-based retrieval tasks, the Geometric Mean method consistently outperforms all other methods [73]. Subsequent work by Seo and Croft in blog feed search and online forum thread search confirms its effectiveness for those tasks as well [110, 113]. Seo and Croft [112] also theoretically justify these results by relating the issue of representing groups of documents to estimating the documents' "center of mass". When using Fischer information as a distance metric, rather than Euclidian distance, the geometric mean provides a more accurate estimate of this center of mass.

### 2.4.3 Voting Methods

Voting, or "data fusion", methods are popular approaches to meta-search, where results from several search engines over the same collection are combined [8, 52]. These methods typically treat the retrieval system as a black box, using only the document scores or ranks from each search engine to produce a final document ranking.

Macdonald et al. have adapted these models to aggregate document ranking as well, specifically looking at expert search [79] and blog feed search tasks [80]. When applied to document aggregate ranking, rather than aggregating different systems' scores on a single document, the voting methods aggregate scores from the same system across members of a document aggregate. These voting methods are inherently small-document models, combining scores or ranks of individual documents together to form the score of the document aggregate.

The more effective voting methods applied to ranking of document aggregates are given in Table 2.2. In these model descriptions $s(D, Q)$ is the score of document $D$ assigned by the underlying retrieval system. The MNZ variants of the voting methods tend to be highly effective, but strongly favor large

| Voting Method | Description |
|---|---|
| Votes$(A, Q)$ | $\lvert A \cap \mathcal{R}_k(Q)\rvert$ |
| CombMAX$(A, Q)$ | $\max_{D_i \in A \cap \mathcal{R}_k(Q)} s(D_i, Q)$ |
| CombSUM$(A, Q)$ | $\sum_{D_i \in A \cap \mathcal{R}_k(Q)} s(D_i, Q)$ |
| CombMNZ$(A, Q)$ | $\lvert A \cap \mathcal{R}_k(Q)\rvert \times \sum_{D_i \in A \cap \mathcal{R}_k(Q)} s(D_i, Q)$ |
| expCombSUM$(A, Q)$ | $\sum_{D_i \in A \cap \mathcal{R}_k(Q)} \exp(s(D_i, Q))$ |
| expCombMNZ$(A, Q)$ | $\lvert A \cap \mathcal{R}_k(Q)\rvert \times \sum_{D_i \in A \cap \mathcal{R}_k(Q)} \exp(s(D_i, Q))$ |

**Table 2.2:** Voting methods applied to ranking document aggregates. $\mathcal{R}_k(Q)$ is the set of documents retrieved at rank $k$, $A$ is a document aggregate containing documents $D_i \in A$, and $s(D, Q)$ is the score of document $D$ assigned by the underlying retrieval system.

documents aggregates due to the multiplicative factor $|A \cap \mathcal{R}_k(Q)|$. To adjust for this bias, Macdonald and Ounis apply a variety of length correction techniques, finding a multiplicative correction that slowly grows inversely proportional to the number of documents in the aggregate performs well, termed "Norm2D":

$$Norm2D + CombMNZ(A, Q) = \log(1 + c\frac{\widetilde{N_A}}{N_A}) \times |A \cap \mathcal{R}_k(Q)| \times \sum_{D_i \in A \cap \mathcal{R}_k(Q)} s(D_i, Q)$$

where $c \geq 0$ is a free parameter and $\widetilde{N_A}$ is the average aggregate size.

### 2.4.4 Feature-based and Learning-to-Rank Models

Recently, feature-based Learning-to-Rank (LETOR) models have been applied to ranking of document aggregates. Feature-based models are distinguished from traditional probabilistic or vector-space models in several ways. In feature-based models document-query pairs are represented by vectors of feature values, typically automatic (LETOR) methods are used to learn parameters of a function to combine those feature values into a document score, and these models often support an arbitrary number of features. Some traditional retrieval models such as document field mixture models can be viewed as feature-based retrieval models.

The application of these models to document aggregate ranking pose some challenges, as typically feature-based rankers derive features at the document-level, not over groups of documents. Many of the highly effective document aggregation techniques described above perform non-trivial transformations of the document scores into aggregate scores, for example necessitating rank-ordering of the documents. These transformations make many learning-to-rank algorithms that rely on gradient-based optimization methods impractical. Several approaches have been developed to overcome these difficulties, however, and two are presented below.

#### Macdonald and Ounis's Voting Feature Model

Macdonald and Ounis extend their work on voting models (described above) to the feature-based ranker setting [81]. In that work, the authors apply a variety of voting models with a range of parameterizations

to generate a large number of features for each query-aggregate pair $(Q, A)$. By doing this, the issues of learning a parameterization of both a document-ranker and aggregation model together are avoided.

By varying the voting method (eg. CombMAX, CombMNZ), rank cutoffs $k$ and underlying document scoring functions, the authors generate hundreds of aggregate-level features. A greedy automatic feature selection algorithm [88] is applied to learn a linear feature-based ranking function. This model is applied to good effect on both expert search and blog feed search tasks [81], finding that different models and parameterizations are effective across different tasks.

#### Fang et al.'s Discriminative Model

Fang et al. take a different approach, taking inspiration from Balog's Model 2 (Equation 2.8) and modeling the two components as linear functions [50]. The parameterizations of these two linear functions are jointly learned from training data via gradient methods. In contrast to Balog's generative model, which ranks aggregates by the likelihood of generating the text of the query, the Fang model takes a discriminative approach, directly modeling relevance of documents, document aggregates, and the association between documents and aggregates. Both arithmetic (AMD) and geometric (GMD) models are evaluated:

$$P_{AMD}(r = 1|A, Q) = \sum_{D \in A} P(r_1 = 1|Q, D)P(r_2 = 1|A, D)P(D) \tag{2.11}$$

$$P_{GMD}(r = 1|A, Q) = \frac{1}{Z} \prod_{D \in A} P(r_1 = 1|Q, D)P(r_2 = 1|A, D) \tag{2.12}$$

where $r, r_1, r_2$ are relevance variables, $P(r_1 = 1|Q, D)$ is analogous to the query likelihood $P(Q|D)$, and $P(r_2 = 1|A, D)$ is analogous to the document-aggregate associations $P(D|A)$. In all cases, the probabilities $P(r_1 = 1|D, Q)$ and $P(r_2 = 1|A, D)$ are modeled as logistic functions, over query-document features and document-aggregate features respectively. This model is applied to expert search tasks significant improvements are found over the baseline, Balog's Model 2.

## 2.5 Methods for Using Aggregate Structure in Document Ranking

Some tasks, such as blog feed search or expert search, necessitate ranking of document aggregates. Other tasks, such as blog post ranking, may not explicitly require the ranking of aggregates, but aggregate structure within the collection can inform the document ranking. In this section we describe several models for including document aggregate information into the document ranking function.

#### Liu and Croft's Cluster Smoothing

In addition to the variety of aggregate ranking models above, Liu and Croft utilize cluster-smoothing models in document ranking tasks [71]. They apply the two-level Jelinek-Mercer smoothing model (Equation 2.4) to smoothing the document language model with the cluster and collection language models. Liu and Croft report that smoothing documents with cluster language models achieves superior performance to a two-stage process of first ranking clusters, then documents within the clusters. This technique is also applied by Petkova and Croft to email message ranking [103].

#### Kurland and Lee Language Modeling Methods

Kurland and Lee also look at document-ranking tasks using clustering information to inform the document ranking [66]. In contrast to the Liu and Croft work, this work defines a cluster of documents as

the $k$-nearest neighbors of a reference document, where Kullback-Leibler divergence defines the distance measure. In this way, the document collection is divided into many overlapping clusters, and the distance measure defines a strength of association between a document and cluster, rather than a binary membership. The cluster-based retrieval model proposed by Kurland and Lee explore a variety of applications of document clustering to the retrieval process, primarily using cluster membership as a feature to filter documents and scoring documents based on a combination of cluster scores. Their combined approach, termed the "interpolation" model, is given as the following:

$$\text{Score}(D,Q) = \lambda P(Q|D) + (1-\lambda)\left(\sum_{A \in \text{ related clusters}} P(Q|A)P(D|A)\right) \tag{2.13}$$

where $P(Q|D)$ and $P(Q|A)$ are standard query likelihood retrieval scores for the document $D$ and cluster $A$ respectively. In this case, the cluster is represented as a large document, concatenating the contained documents. $P(D|A)$ is a measure of association between the document and cluster, and several heuristics are applied to select "related clusters" to consider.

#### Diaz's Score Regularization Methods

Diaz takes a similar but more formal approach [42]. Rather than explicitly defining clusters of documents, that work uses the inter-document similarities directly to "regularize" retrieval scores so that scores of similar documents are close to each other. The algorithm presented by Diaz is based on an iterative dispersion of retrieval scores across a document graph with edges weighted by a similarity metric. In each step in this iterative process, the documents scores are adjusted as

$$\mathbf{f}^{t+1} = (1-\beta)\mathbf{y} + \beta\mathbf{S}\mathbf{f}^{t}$$

where the vector $\mathbf{y}$ is the original retrieval score, $\mathbf{f}^t$ gives the scores at step $t$, $\mathbf{f}^0 = \mathbf{y}$ and $\mathbf{S}$ corresponds to diffusion operator, roughly analogous to a document-by-document similarity matrix. Although document aggregates are not explicitly defined by Diaz's approach, the structure of the diffusion matrix $\mathbf{S}$ could define groups of documents — for example indicating the membership of blog posts to the same blog, rather than representing a textual document similarity metric.

## 2.6 Document Aggregate Ranking Tasks

The techniques described above have been applied to a variety of document aggregate ranking tasks. In this section we present several of these tasks, highlighting the major research themes and use of the above models.

### 2.6.1 Cluster-Based Retrieval

Cluster-based retrieval has been proposed to improve IR system performance and efficiency, and is typically viewed as a multi-stage process. This work is motivated by the *Cluster Hypothesis*, stating that *documents relevant to the same query are likely to be similar to each other* [121]. Thus, if a system can identify a cluster containing some documents relevant to the query, that cluster is likely to contain other documents also relevant to the query.

Cluster-based retrieval typically proceeds as follows: First, as a preprocessing stage, the document collection is divided into topically related, possibly overlapping clusters of documents. At query time,

those clusters of documents most likely to contain relevant documents are retrieved. Finally, documents are selected from within those clusters to form a final document ranking.

Work on clustering document collections and merging results from multiple clusters is beyond the scope of this work and will not be discussed here. Selecting or ranking clusters, however, is some of the first work dealing with retrieval of sub-collections of documents.

Early work on cluster-based retrieval by Salton [107] and van Rijsbergen[121] viewed the process primarily as an means to improve retrieval system efficiency. By selecting only a small number of document clusters to search, one need not evaluate the query against the entire collection. In their work, clusters are ranked by the degree of match between the query and a "cluster representative", which may be a single document within the cluster or some summarized representation of the cluster contents such as the cluster centroid.

More recent work on cluster-based retrieval in the language modeling framework utilizes document clusters both for document selection, and as an additional source of evidence in the scoring of documents. Liu and Croft apply the variety of cluster representation models above (Section 2.4.2) to the task of cluster selection, as well as the two-stage Jelinek-Mercer smoothing model (Equation 2.4) to smoothing documents with their corresponding cluster's language model [71, 72]. Liu and Croft find that both the cluster-smoothing model and the PCS cluster representation method are particularly effective techniques for this task.

Kurland and Lee evaluate a variety of cluster-based retrieval methods. Their most effective method takes a multi-stage approach, first retrieving documents using a standard language modeling retrieval model, next clustering those results into overlapping nearest-neighbor clusters, and finally combining document scores with associated cluster scores to re-rank the documents [66]. The final stage of this process, scoring of clusters and combining the document- and cluster-scores, is of most interest to the work here. This final probabilistic ranking function is shown in Equation 2.13 and the clusters are scored by a concatenation large-document model.

Diaz takes a similar approach, linearly combing document scores with scores of similar documents in an iterative "score regularization" approach [42], described in more detail above, Section 2.5. This work is similar to Kurland's, but neither necessitates the definition of clusters, only a similarity metric, nor application of heuristics to select which clusters to to draw from when filtering documents.

### 2.6.2 Resource Ranking

Distributed information retrieval, or federated search, is a set of tasks relating to document retrieval across many document collections rather than a single centralized index. One of those tasks, resource selection, is the ranking of available document collections in order to select those most likely to contain many documents relevant to a query.

Early work on resource ranking using belief networks by Callan et al. [26, 27] introduced the CORI ranking algorithm, which scores each collection based on summary statistics such as document frequencies and collection lengths. This algorithm assumes some knowledge of the collection statistics for each of the collections to be ranked and takes a large-document approach. This ranking formula is based on TF-IDF weighting, assuming each collection can be represented by aggregate statistics such as the number of documents containing a query term. Subsequent work by Si and Callan [114] introduced the ReDDE algorithm which models resources via the individual document they contain. This resource ranking method is described in more detail above, Section 2.4.1.

### 2.6.3 Expert search

Expert search has been studied in recent years in the Enterprise Track at the annual Text REtrieval Conference (TREC) [11, 40, 117]. The task of expert search is that of of finding people within an organization with expertise pertinent to a user's query. At TREC, this task has been performed with the use of an enterprise collection of documents, containing references to employee expert names or email addresses within the organization. This task is generally approached as a document aggregate ranking tasks by first associating documents with experts, for example by identifying mentions of candidate experts, and then ranking those document aggregates in response to a query.

Many of the document aggregate ranking methods presented above have been applied to the task of expert search: Balog's Model 1 and Model 2 [14, 12], cluster representation models [103], Macdonald's voting and feature models [79, 81], and Fang's feature-based discriminative model [50]. When comparing the language modeling methods, the general trends are that small-document models (eg. Balog's Model 2) tend to outperform large document models (eg. Balog's Model 1) [12]. When considering the voting techniques, the aggregate-size normalized version of expCombMNZ tends to perform well, both by favoring high-scoring documents and correcting for biases towards large aggregates [77]. Finally, the feature-based models are highly effective, and when adequate training data is available have the potential to outperform most other approaches [50, 81].

## 2.7 Information Retrieval for Blogs and Other Social Media

Social media collections pose new an interesting challenges for information retrieval research. As a form of social interaction, these collections are archives of dialog among the users. These collections can typically be organized along several dimensions, including general topic, discussion thread, and author. In this section, we will review previous research that looks at several social media collections that exhibit this type of structure. We focus here on blogs, email collections, online forums or message boards, and community question answer (CQA) archives.

### 2.7.1 Blog Search

Mishne and de Rijke present some of the first work analyzing the characteristics of blog search, performing an extensive log study from an active blog search engine [92]. They find that blog search queries are primarily of two types: "context queries" looking for specific mentions of a named entity in blogs, and "concept queries" which are more broadly topical. They also find a stronger bias in blog search queries towards timely news events than typically seen in web search. Hearst et al. give a more recent discussion of desirable qualities of blog search engines [58]. They conclude that blog search engines should support information access in several ways:

1. Access at the blog post level, with a focus on the temporal nature of blogs,
2. Access at the blog level, to aid in discovery of which blogs to follow over a period of time, and
3. Access to historical archives of blog posts.

The blog search track at TREC addresses these approaches to access to blog collections to some degree, with a focus on two different search tasks: blog feed search, and blog post search [82, 101, 102].

#### Post Search

Blog post search is the task of retrieving individual blog posts relevant to a query. As studied at TREC, the focus of this task has been on retrieving both relevant and opinionated blog posts, with a multi-stage evaluation considering these aspects separately [82, 101, 102]. The primary approaches to the task at TREC first apply standard retrieval techniques such as language modeling to retrieve blog posts, and then apply a variety of opinion detection and polarity classification methods [101]. The opinion detection and polarity classification techniques applied are out of the scope of this thesis, as we focus exclusively on topical post retrieval.

The major approaches to blog search include divergence from randomness models [55], language modeling [132], and BM25 retrieval models [136]. Several techniques specific to blog collections have been applied to topical blog post search since the TREC evaluations, including spam classification [91, 127], temporal analysis of the posts [91], content analysis methods [94], and methods to identify credible and high-quality bloggers [127]. Section 4.6 describes these methods in more detail, and specifically relates them to the work in this thesis.

#### Feed Search

Blog feed search is the task of ranking blogs with respect to a user query, where the returned blogs have a "principle, recurring interest" in the topic [101, 102]. This is also a task of ranking document aggregates, as the blogs are composed of a series of blog posts, typically written by the same author. This task has been viewed as similar to both expert finding [80, 15] and the resource selection [7, 45], and the existing approaches to blog feed search to some degree mirror approaches to those tasks.

Similar to research in expert search and resource ranking, retrieval models that view blogs as a single "large document" have been compared to models that view blogs as a collection of individual "small documents". Balog et al. directly applied their expert search models to the task of blog feed search [15]. They found that, in contrast to expert finding, Model 1 (the "large document" model) outperformed Model 2 (the "small document" model) for this task. Similar findings were shown by Seo and Croft [109, 110], Arguello et al. [7] and Elsas et al. [44], who all apply resource selection techniques to this task. Macdonald's voting models have been applied to blog feed search as well [80].

### 2.7.2 Email and Newsgroups

Email and newsgroup collections have been the target of much text classification work in the past. Classification tasks studied with regard to email collections include spam classification [106, 37], email foldering [34, 17], and prioritization [133]. Other automatic prediction tasks for email collections beyond document classification include predicting "information leaks" [32] and likely message recipients [33].

Information retrieval over collections of email has also been studied at the TREC enterprise track in 2005 and 2006 [40, 117]. In these years, the enterprise track used a collection of archived email messages from the World Wide Web Consortium (W3C) [39] and several tasks were run, including known-item finding and discussion search. The W3C collection consists of several public email list archives, and messages within each of those lists are organized into message threads. Many of the approaches to the TREC tasks leverage this thread structure.

Several TREC participants took a fielded-retrieval approach, where email messages are modeled as a combination of text from the subject, body and quoted portions of the message. Ogilvie and Callan [96], working within the language modeling framework, modeled the email message as a mixture model, combining evidence from these sources as well as from response messages. Craswell et al. [39] took a similar approach using the BM25F ranking function. More recent work using the same collection by Weerkamp et

al. [126] leverages the language used at different levels of the thread context to influence which terms are used for query expansion. These different contexts are defined as the message itself, the message thread and the entire mailing list. All of these approaches found additional utility in leveraging email message priors, such as the depth of the message in the thread. Several studies additionally found that smoothing the email message with the thread can have a positive affect on retrieval performance [103, 112].

Other work on "search" in email collections by Minkov and Cohen [90] focuses on defining similarity measures between objects in the collection such as authors or messages. These similarity measures are then used to disambiguate person names or identify related messages, likely to belong to the same message thread. This work takes an alternate view of collection structure, rather than using the structure to identify sub-collections, it is used to define explicit relations between the objects, resulting in a graph-based representation of the collection.

Similar to email collections, newsgroups provide message, author and thread structure. Xi et al. [129] looked at message retrieval in this setting, employing a learning-to-rank approach to combining various content and structural features in a ranking function. They found that textual features of the message, its thread, and other responses in the thread are the most useful in ranking.

The majority of the work in search over email or newsgroups is concerned with combining evidence from different parts of the message, such as the title, body or replies, as well as using non-textual features such as the number of replies. None of the previous work, however, is concerned with leveraging more complex collection structure, for example through modeling the expertise of the author when ranking messages. Additionally, in contrast to the work presented in Chapter 6 all of these studies have looked at retrieving single messages, rather than a message thread.

### 2.7.3 Online Message Boards and Forums

Online messages boards support user discussion in a structured environment, with messages, threads, and frequently higher-level classifications of those threads. There are many thousands of public online forum websites and they typically cater to a specific topic of discussion such as health issues, movie reviews, programming languages or politics. Typically message boards contain several topical *sub-forums*, which in turn contain the message posts and post replies. The message post/response structure are displayed either flat or hierarchically and most frequently ordered according the time of posting. Message boards also frequently provide users with the ability to cultivate somewhat of a online personality with pictures and personal biographical profiles. Some message boards provide additional features such as galleries, recipe databases, explicit "social networking" features such as marking connections between other message board users.

Seo and Croft have studied the task of thread retrieval in online forums, with a specific focus on the reconstruction and use of hierarchical message-reply structure within the thread [108, 113]. The authors predict message-response relations with features such as text similarity, message depth, temporal gap between messages, co-authorship and author mentions. A high accuracy of reconstructing the message-response relations is reported, and features based on this hierarchical thread structure are use in thread-ranking algorithms. The authors define "contexts" within the thread — messages grouped by their relation to other messages — and combine retrieval scores from various contexts via the Pseudo-Cluster Selection model described above. Using some of the reconstructed thread information, the authors show significant improvement over models ignoring this hierarchical structure [108].

Several studies have looked at knowledge extraction in online forums, for example identifying question-answer pairs [36] or identifying responses that provide context and an answer to a previous question in the thread [43]. Somewhat similar to a retrieval task, Feng et al. [51] developed a "discussion-bot", which

responds to new forum posts with automatically identified related questions and answers. The question-matching component of this system retrieves likely answers with a vector-space TF-IDF ranking formula.

### 2.7.4 Community Question Answering

Community Question Answer (CQA) services such as Yahoo! Answers[1] resemble online message boards in structural organization. On CQA sites, initial posts take the form of a question, and participants can post an answer in response. They are similar to online message boards in their organization into topical "threads" with contributions from different users. However, in CQA sites, users typically only provide a single answer in response to a question, whereas online forums provide more of a conversational structure, with users potentially contributing a series of posts responding to different messages in the tread at different times. CQA sites also have several features not present in online forums. On these sites, users accumulate "points" when their answers are judged best by the original asker, or voted on by others in the community.

Yahoo! Answers has been extensively studied by Agichtein et al. [2, 21, 20, 75, 74, 62, 68]. Their work explores various prediction tasks with this dataset, primarily focusing on feature design and engineering. Several of the tasks studied include: automatically identifying "high quality" questions and answers [2], predicting if askers are satisfied by the answers provided [75] and identifying malicious voting in community judgement system [20].

Several studies have looked at CQA archives as data sources for factoid question answering. Bian et al. [21] used a Yahoo! Answers dataset and Xue et al. [130] use a similar dataset from Wondir[2] for QA evaluations. In these studies, questions from several TREC QA datasets were used as queries and answers were identified from the TREC answer patterns [123, 124]. In the first study, answers were ranked via a learning-to-rank algorithm, using several content-based, community-based and statistical features from the dataset. In the second, a statistical translation model was adapted to retrieving questions and answers.

Some of these studies model answerer expertise in prediction or ranking. In the Agichtein studies, this is done through gross measures of expertise, rather than query- or question-specific measures. Some of the expertise measures used in those studies includes the number of "points" an answerer has received or several graph-based authority measures calculated over the user-question-answer graph [2]. This study reports that some of the most effective features in predicting the quality of an answer are community judgements on the previous answers provide by the same user.

[1] http://answers.yahoo.com
[2] http://www.wondir.com

# Chapter 3

# Conversational Social Media Collections

In this chapter we discuss the collections used throughout the thesis experiments. First, we briefly present the BLOG06 collection, used for both the blog post search and blog feed search experiments discussed in Chapters 4 and 5. Next, we describe the construction of two online forum collections. These collections were built for the purposes of studying thread search in online forums, and those experiments are presented in Chapter 6.

## 3.1 BLOG06 Dataset

The BLOG06 collection has been used at the Text REtrieval Conference (TREC) for several years [82, 100, 102], as well as in the experiments presented in Chapters 4 and 5. The process of building the collection has been described in detail by Macdonald and Ounis [78] and in this section we give a brief overview of the collection statistics.

This is a collection of all the posts published at 100,649 blogs over the course of 11 weeks in the end of 2005 and beginning of 2006. The collection consists of three components: the feeds are XML documents (mostly RSS and ATOM format) fetched weekly describing the most recent posts in the blog, the permalinks are HTML documents containing the content of each post in the collection, and the homepages are the main landing page of the blog at the time the feed was originally collected. In order to allow reader comments to accumulate on the blog post, several days delay was introduced between post discovery in the feed XML and crawling the permalink HTML page. In total, this collection contains over 3.2 million permalink documents, 750,000 feed fetches, and 320,000 homepage documents. An effort was made to include mostly popular blogs in the collection (about 70%), but also intentionally include some spam bogs (18%) and other popular websites (12%).

Further details on how this collection is used experimentally are presented in Chapters 4 and 5.

## 3.2 Online Forums

Online forums, or message boards, contain a wealth of user generated content over a wide range of topics: from computer hardware manufacturers[1], to movies reviews and commentary[2], to genealogy research[3] to gardening[4]. The contributors to online forums are often domain experts and these social information

[1]MacRumors Forum: http://forums.macrumors.com/
[2]IMDB Forum: http://www.imdb.com/boards/
[3]Ancestry.com Forum: http://boards.ancestry.com/
[4]Gardenweb Forum: http://forums.gardenweb.com/forums

spaces host in some cases many millions of archived messages. Access to this historical information, however, is often rudimentary, providing the most basic of browsing interfaces and simple keyword message-searching facilities.

In the remainder of this chapter, we first present the structure of online forums, discussing why forums are interesting testbeds for investigating the use of collection structure in retrieval algorithms. Next, we describe the construction of two online forum test collections. We present the MacRumors.com test collection, with relevance information extracted via mining of usage behavior present within the collection itself. Finally, we present the Ancestry.com test collection, a much larger document collection with more queries and annotations by relevance assessors.

#### Research Contributions

The major contributions of this chapter are:

- The development of a novel method for mining relevance information *in situ* from a conversational document collection. We use this method to build the MacRumors.com Forum IR test collection, leveraging the social interaction between participants in the online forum community to identify information needs and relevant documents.
- The construction and release to the research community of the Ancestry.com Forum test collection[5]. This is the first publicly released IR test collection for online forum thread search, and to our knowledge the largest dataset of its kind to be studied in an academic setting.

## 3.3 The Structure of Online Forums

Online forums are highly structured, often with different levels of organizational granularity and different axes of organization. Messages are grouped into *message threads*, representing a single conversation between a group of contributors. A message thread has a single *start message* contributed by the *thread starter* and zero or more *response messages* contributed by *respondents*. Message threads are frequently displayed chronologically in a "flat" structure where each message in the thread has at most one response. Some online forums support hierarchical organization, where each message may have more than one response. Figure 3.1 shows this thread-level organization and Figure 3.2 shows an example thread from the MacRumors forum with these different structural elements indicated.

Message threads are often grouped into *sub-forums*, which usually represent sub-topics discussed in the online forum. Sub-forums can, in turn, be organized hierarchically, with child sub-forums representing more specific topics. Non-topical organizational axes could also be represented in the subforum hierarchy as well, such as the geographic organization present in the Ancestry.com forum discussed below. An example of the MacRumors forum hierarchy is shown in Figure 3.3.

The thread starter selects a sub-forum when composing the start message, and thus these sub-forums can be thought of as an author assigned thread classification. In the case of the MacRumors.com forum these sub-forums are focused on specific hardware or software products. In the Ancestry.com forum geographic, alphabetical, and topical organizations are present in the subforum hierarchy. Figure 3.4 shows this subforum-level organization.

Another axis of organization is time. Messages in the online forum are posted at a specific time, which is recorded in the message metadata. Online forum interfaces frequently order threads by the date of their most recent contribution, allowing users to browse conversations recently active. As can be seen in Figure

[5] See `http://www.cs.cmu.edu/~jelsas/data/ancestry.com` for distribution details.

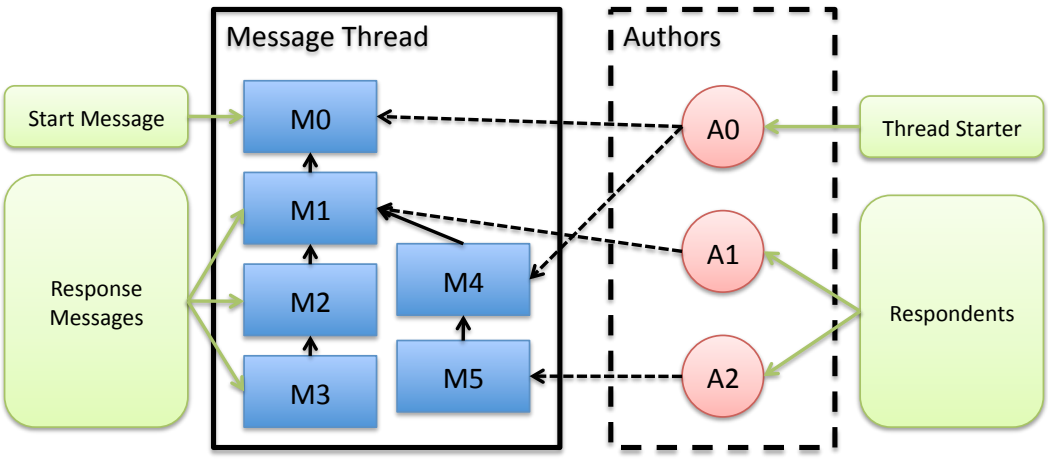



**Figure 3.1:** Online Forum Message Thread Organization, showing a simple hierarchical message organization.

3.7, some threads are active for a very short time, while others can have contributions to the thread over the course of several years.

From an information retrieval perspective, the rich organization present in online forums provides several interesting challenges.

1. The first challenge is that there are several possible units of retrieval. Depending on the task, an information seeker may be interested in retrieving individual messages, entire message threads, or even an entire sub-forum. We may also envision a situation where, like the expert finding tasks described in Chapter 2.6.3, information seekers are interested in finding authors with a high level of expertise on a particular subject. The unit of retrieval is primarily dictated by the task, and in this work we focus on retrieving message threads. We believe this to be a generally useful unit of retrieval, and further discussion of the retrieval granularity is given below.
2. A second challenge in information retrieval over this type of collection is that of leveraging this structure within ranking algorithms, such as those discussed in Chapter 2. In the task of message thread ranking, an information seeker may be more interested in threads residing in a particular sub-forum that is closely related to their query. Or, users may be interested in threads with significant contributions form authors who have a high level of expertise with respect to their information need. To leverage this type of structure in ranking, we must develop algorithms that can model the expertise of authors and the affinity of the query to the different sub-forums.

In the following sections, we describe the construction of two information retrieval test collections for ad-hoc thread retrieval in online forums.

## 3.4 MacRumors.com Forum Dataset

Some of the work presented in the following chapters uses a crawl of a technically-oriented, online message board, the MacRumors.com Forum[6]. This is a large online forum dedicated to the discussion of news and opinion relating to the computer manufacturer, Apple, Inc. This section describes the construction of an IR test collection with the MacRumors.com Forum dataset and follows the outline below:

- In **Section 3.4.1** we describe the crawl of the online forum website and the resulting document

[6] http://forums.macrumors.com

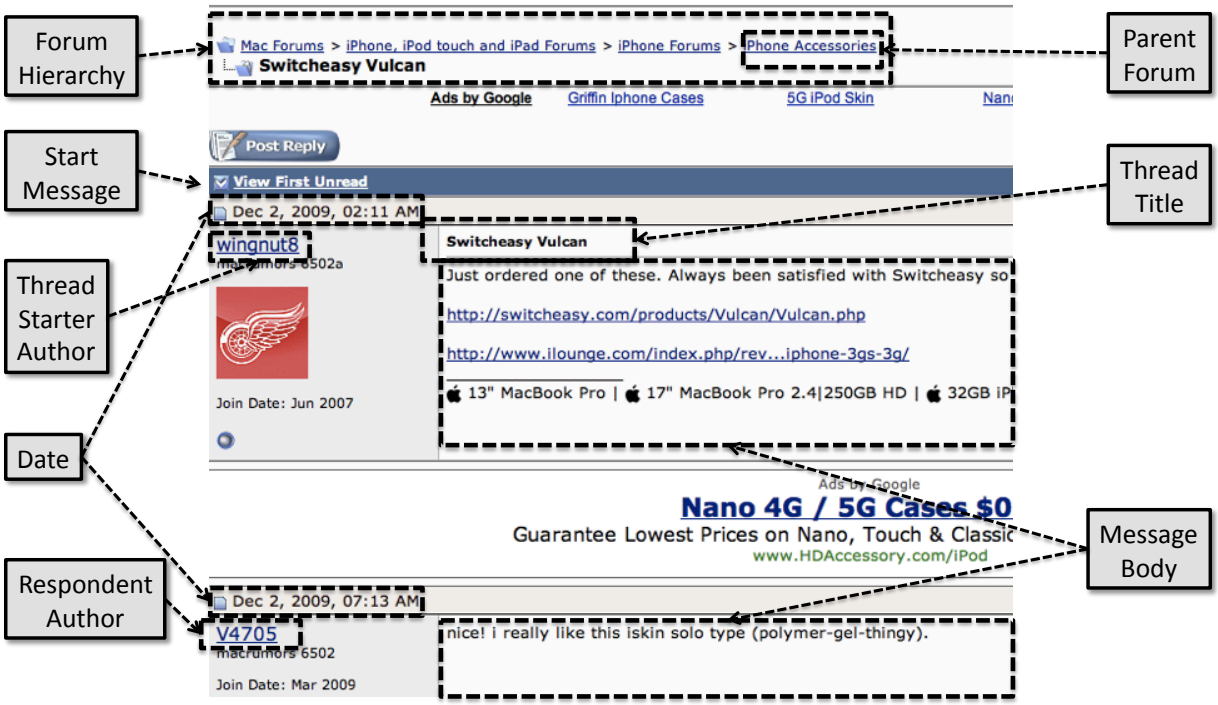



**Figure 3.2:** Example thread from the MacRumors Forum showing different structural elements of the thread.

collection.

- In **Section 3.4.2** we discuss a method for discovering information needs and relevance information *within the collection itself*, rather than using an external query log and relevance assessment.
- In **Section 3.4.3** we discuss the benefits and drawbacks to the method used to build the collection, and contrast this method to traditional methods for creating IR test collections.
- In **Section 3.4.4** we describe the process used to generate keyword queries from the identified information needs in the collection.

### 3.4.1 Collecting the Macrumors.com Forum Documents

In March of 2008, an exhaustive crawl of the message thread pages was conducted. The resulting dataset and contains over 3 million messages, over 370,000 threads, with contributions from over 85,000 authors. Detailed dataset statistics are given in Table 3.1. Figure 3.5 shows the message and thread volume distributions. Figure 3.6 shows evolutionary statistics of the forum over time. To conduct this crawl, each HTML page corresponding to a single thread was downloaded. Threads are numbered sequentially and this unique thread identifier is part of the URL used to view the thread (e.g. `forum.com/showthread.php?t=123`). Through this simple programmatic access of forum threads, it is possible to make an exhaustive crawl of all threads on the forum.

The thread HTML pages downloaded in this manner contain the contents of the start message and the first twenty-four messages written in response to that message. These messages are extracted from the HTML, retaining structural information such as thread titles, message titles, user IDs and message time stamps. Note that although an exhaustive thread crawl was performed, not all forum messages were extracted. Because only the first 25 messages for each thread were downloaded, 9.8% of the threads in the

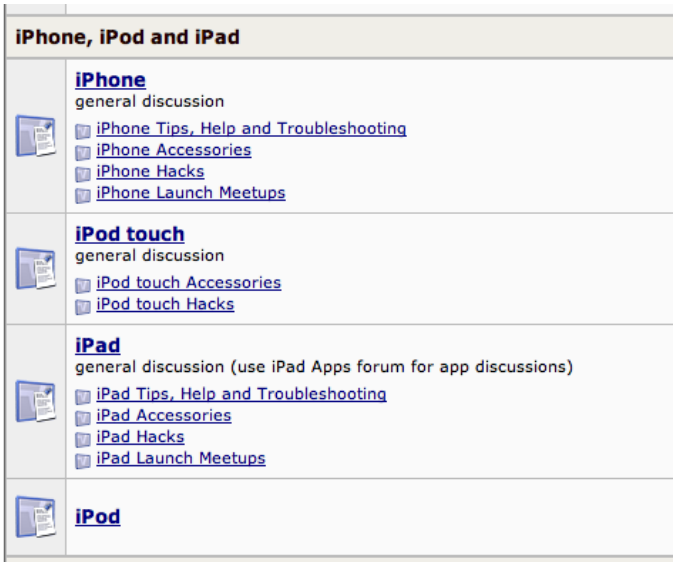



**Figure 3.3:** Excerpt fo the forum hierarchy from the MacRumors Forum showing the overall topical organization under the "iPhone, iPod and iPad" node.

| | |
|---|---|
| Number of Messages | 3,108,244 |
| Number of Threads | 373,639 |
| Number of Sub-forums | 86 |
| Number of Unique Users | 85,298 |
| Message Date Range | April 2001 - March 2008 |
| Mean (Standard Deviation) Message Length | 81.8 (86.1) tokens |
| Mean (Standard Deviation) Thread Size | 12.4 (49.3) messages |

**Table 3.1:** MacRumors.com Forum Dataset Statistics.

crawled collection are incomplete. Although this is a relatively small fraction of incomplete threads, these missing messages do account for a sizable portion of the entire forum: roughly 30% of the total message on the forum at the time of the crawl are not included in our dataset.

### 3.4.2 Identifying Information Needs and Relevant Documents

As stated above, these online forums host an enormous archive of historical information, over 3 million messages in the case of the MacRumors.com forum. Search-based access to this information, however, is often rudimentary. Because of the difficulty in accessing the forum archives, users often post questions to messages boards that may have been answered in previous threads in the archive. Commonly, when this happens, another user responds to that question with a link to a previous discussion possibly containing the answer. We can leverage this interaction between online forum users to build an information retrieval test collection. The original question can be considered a query and the linked-to thread a relevant document.

We make the assumption that the typical useful unit of retrieval for message board search is the message *thread*. Although this is certainly not always the case — sometimes a single message may fully answer an information need — the thread provides useful conversation context and discussion. When viewing the message board, a thread-view is typically most convenient. Additionally, of all the intra-

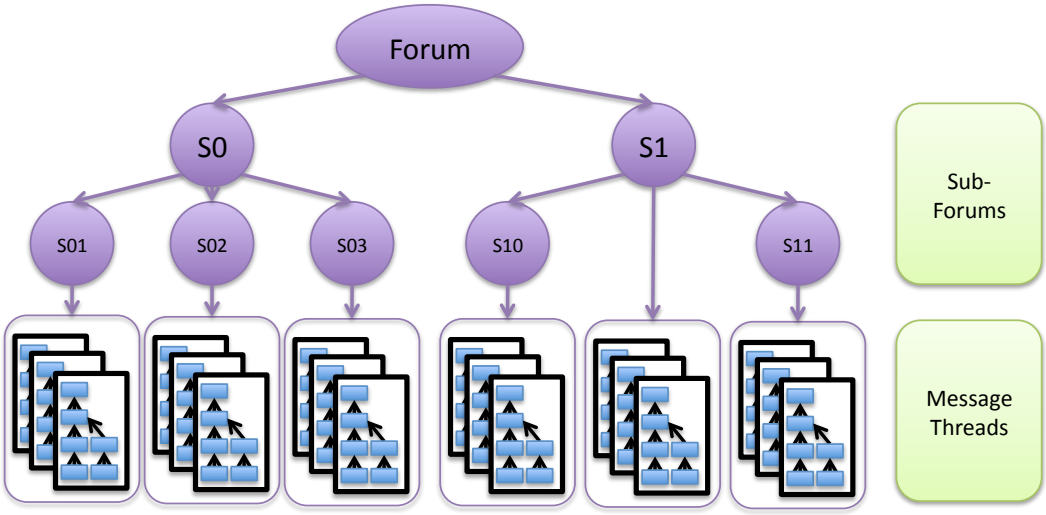



**Figure 3.4:** Online Forum Sub-Forum Organization. Shows two top-level sub-forums ($S0$ and $S1$) and with various child sub-forums. Message threads can belong to sub-forums at any level of the hierarchy.

forum linking in the MacRumors.com forum, 86% of the links refer to other threads, rather than messages or other possible units of retrieval. Table 3.2 shows the volume of linking to different possible units of retrieval in the MacRumors.com forum.

| Link Target | Volume | Percent of Total Link Volume |
|---|---|---|
| Thread | 27,068 | 86.43% |
| User profile page | 2,071 | 6.61% |
| Message | 1,140 | 3.64% |
| Subforum | 688 | 2.20% |
| Search result page | 350 | 1.12% |

**Table 3.2:** Volume of intra-forum linking in the MacRumors.com Forum.

To build a thread-retrieval test collection, we isolated all response messages that contained a link to other threads in the same collection. These candidate "answer messages" may be an instance of a forum user answering a start message question with a link to a previously posted relevant thread. Over 17,000 candidate answer messages were identified, and a random sampling of 550 of the corresponding threads were manually annotated for containing a question/link-answer pair. A thread contains a question-answer pair when the following conditions hold true:

1. A response message provides a hyperlink to a previous thread in the message board.
2. The start message in this thread contains a question that is answered by the linked-to thread.
3. Subsequent response messages in the thread do not indicate the linked-to thread is irrelevant to the original question.

Many instances of within-forum linking are common, and do not necessarily indicate a question/answer interaction. Often links indicated a forum user was experiencing similar problems as the thread starter (but providing no answers), links to a news topic that may have been previously discussed, or links advertising items for sale by forum users. Because of the volume and diversity in link types in the forum

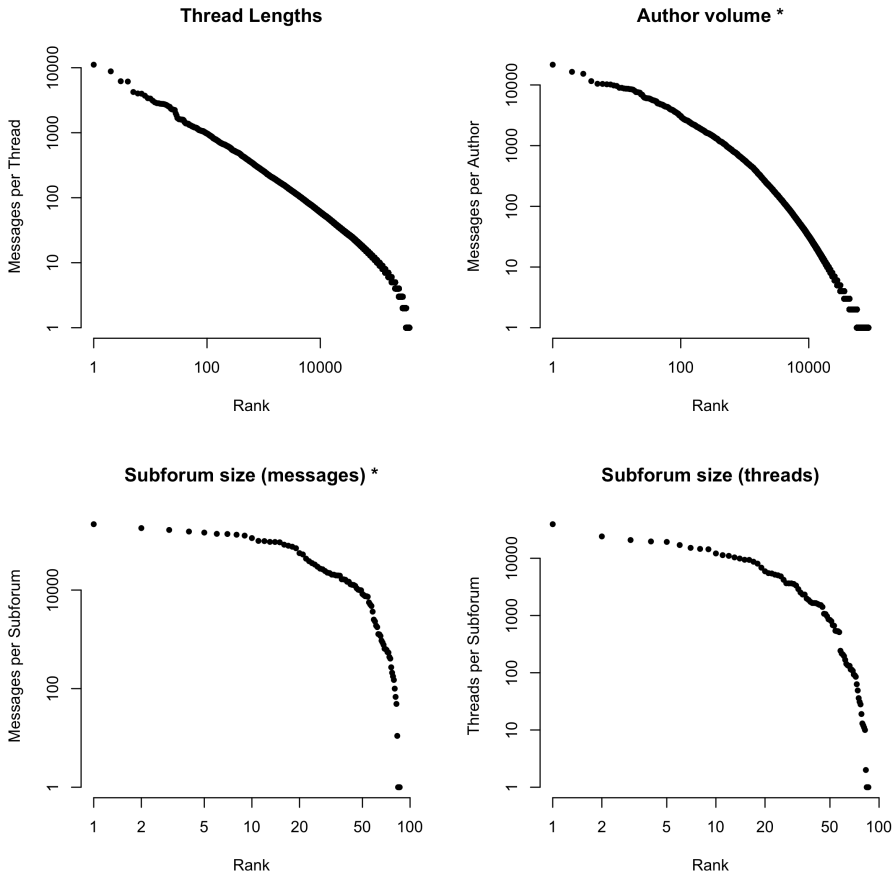



**Figure 3.5:** MacRumors.com Forum size distributions. Top: thread length and author volume distributions. Bottom: subforum size distributions (messages & threads). Plots marked with an asterisk (*) are calculated from only the first 25 message in each thread.

data, less than 10% of the sampled links indicated a forum thread that directly answered the thread starter's question. In total, 48 question-answer pairs were identified in this test collection.

Previous work has shown that message length is a leading indicator in predicting question and answer quality in community question answer sites [2]. Based on these observations in other collections, it may be the case that that relevant threads in our collection are longer than the typical forum message. This hypothesis is borne out in our data, with the mean message size in answer threads being 93.7 tokens as compared to 81.8 tokens in the entire collection (see Table 3.1). Additionally, answer threads tend to be longer, containing on average 43.5 messages compared to the average of 12.4 messages per thread in the collection. Both of these differences are statistically significant with a 1-tailed t-test ($p < 0.001$ and $p < 0.005$ respectively).

Figure 3.7 shows the temporal distribution of the question message (black dots) and the answer thread activity (black lines). While many questions messages are composed shortly after the answer thread is started, this is not necessarily always the case. Some answer threads are active for several years before the question is posted. The answer threads are also active for a wide variety of durations, reflecting the overall popularity of those threads. Some of the more popular answer threads have many messages that are contributed over the course of two or more years. Others are very short lived, where messages may

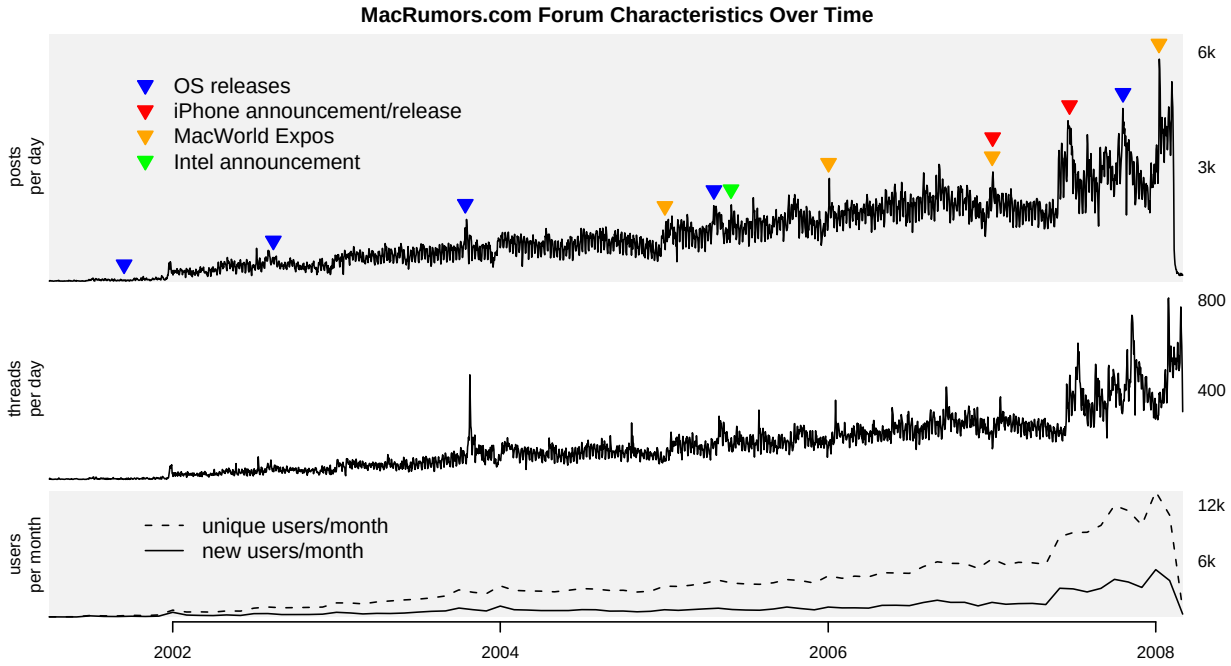



**Figure 3.6:** MacRumors.com forum characteristics over time. From top to bottom: Daily message volume, with significant events indicated with colored triangles; Daily thread volume; Monthly user volume Note: Volume drop in early 2008 is an artifact of the crawl method, where the most recent threads in the collection are incomplete.

only be contributed on a single day.

### 3.4.3 MacRumors.com Dataset Discussion

This type of test collection creation does have some distinct advantages over other typical retrieval test collections. First, the queries represent real information needs of real users of the online forum. These information needs are also much more verbose than typical keyword queries on a web search engine, providing a retrieval system more evidence with which to use in relevance scoring. The "relevance judgement", provided by another forum user linking to a previous thread, also presents in-situ relevance information — sensitive not only to the original question, but also to the overall nature of the forum and the time when the question was asked.

There are several drawbacks inherent in this type of corpus creation, most importantly with regard to the exhaustiveness of the relevance assessment. Typically in TREC-style collection development [125], ranked results from several retrieval systems are pooled and those pooled documents are assessed for relevance. When the systems' output is sufficiently diverse and relevance assessment is sufficiently deep, this produces a reasonably complete relevance assessment for each query — if a relevant document is in the collection, it would most likely be retrieved by one of the systems and be judged by being admitted into the pool. The method of collecting relevance judgements presented here, on the other hand, will not produce anything close to an exhaustive set of relevant threads. In the great majority of cases, only a single thread is linked to in a subsequent reply message. There is no guarantee that this thread is the best or only relevant thread in the collection. For this reason, we must take care in evaluating thread retrieval algorithms with this dataset, and not to assume non-judged threads are necessarily irrelevant.

One additional facet of this test collection is significantly different than IR test collections typically

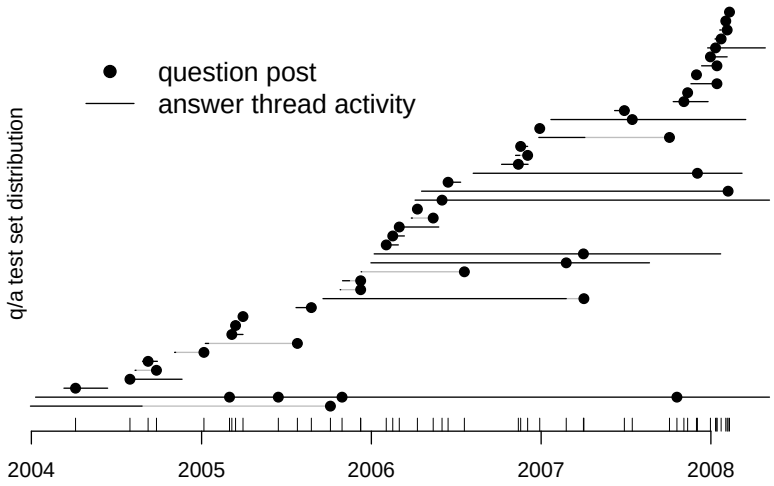



**Figure 3.7:** MacRumors.com forum date distribution of question message and answer thread activity. Each horizontal line represents a single answer thread, the solid black portion indicating the message posting activity. Each solid circle indicates the time the corresponding question message was posted. Some answer threads (eg. second from the bottom) were identified as relevant to more than one question.

used. The online forum data is temporal, with threads and messages constantly being added to the collection. For this reason, any evaluation with this data must take into account the time at which a question is asked, and only consider messages created prior to that time for evaluation.

### 3.4.4 Generating Queries

In order to use this dataset as an IR test collection, we must convert the question messages into keyword queries. In preliminary testing, we evaluated several methods for generating keyword queries from the original text of the question message. The goal of generating queries from the question messages is to capture all the pertinent information in the question while eliminating as much of the spurious text as possible. We evaluated several manual methods for generating queries, ranging from using the entire text of the question message, to only a select few words from the message title. All of the query generation methods used only words in the message, never adding unseen tokens to the query. Initial testing showed that focusing only on the title tends to omit too many highly useful query terms, but some considerable cleaning of the message text is necessary to form effective and concise queries. Automatic methods for generating queries, such as those proposed by Bendersky and Croft [18], were not evaluated in this work. We leave this as a possible direction for future work.

To generate queries form the question messages, we first extract the text from the message body and title, eliminating any HTML, quoted text or "signature" text. Then, we manually clean the original message text, eliminating any text unrelated to the central question of the message. The goal of this cleaning is to remove any conversational introduction comments by the poster, phrases like "Hello", "Thank You" or "Any suggestions", and general commentary not related to the topic of the question. Preliminary testing indicated that this method consistently produced effective queries. All tests on the MacRumors.com Forum reported below use this cleaned version of the query set. Table 3.3 shows summary statistics of the queries used in the experiments with this dataset.

| Queries | Mean (St. Dev.) Query Length | Mean (St. Dev.) Relevant Per Query |
|---|---|---|
| 48 | 5.90 (2.94) | 1.79 (2.22) |

**Table 3.3:** MacRumors.com forum text collection query statistics.

## 3.5 Ancestry.com Forum Dataset

In this section we describe a second online forum dataset from the website Ancestry.com. Ancestry.com is an online resource for historical and genealogical research. The associated forum provides users of the site with a mechanism to discuss their research, share findings, and collaborate. As with the MacRumors.com forum dataset, we build this dataset for evaluating the thread retrieval task. In this case, however, rather than rely on the sparse relevance information present in the linking behavior of the forum participants, we collect relevance annotations from assessors.

The dataset presented here is almost one order of magnitude larger than the MacRumors.com dataset, in terms of message volume, with more than three times more queries that test collections previously studied for this task [108]. The Ancestry.com document collection, queries and relevance assessment are publicly available[7], as well as the software tools developed to collect the assessment and perform the evaluation. To our knowledge, this dataset is the only publicly available IR test collection for studying thread retrieval in online forums.

Our discussion of the construction of an IR test collection with the Ancestry.com Forum dataset is organized as follows:

- In **Section 3.5.1** we present an overall description of the document collection, including the dataset size and a description of the major organizational dimensions.
- In **Section 3.5.2** we analyze the Ancestry.com query log and discuss sampling queries from that log to assess for the IR test collection.
- In **Section 3.5.3** we discuss our process of document pooling in order to create a diverse set of potentially relevant documents for later assessment.
- In **Section 3.5.4** we present the process of assessing document pair preferences, the decisions that were made when considering different assessment protocols, and we present the algorithm used to select document pairs for assessment.
- In **Sections 3.5.5** and **3.5.6** we present the results of a pilot assessment, analysis of inter-annotator agreement, and refinements in the assessment process for a further round of assessment.
- In **Section 3.5.7** we present the evaluation measures used in our evaluation based on the pairwise preference assessments collected.

### 3.5.1 Document Collection

The Ancestry.com document collection was gathered in collaboration with the company. The online forum has been active for well over ten years, and contains more than 22 million messages. This dataset is significantly larger than the MacRumors.com dataset, and is organized similarly with messages organized into threads, and threads organized into hierarchical subforums. A full copy of the messages database was generated in July of 2010, containing all messages accessible on the forum at that time. The following metadata is included with the messages:

[7] See `http://www.cs.cmu.edu/~jelsas/data/ancestry.com` for distribution details.

- Unique message identifier, containing thread membership information
- Author name
- Unique numeric author identifier (or 0 if missing)
- Publication date (or 01-01-1900 if missing)
- Subforum name
- URL of the original document
- Message title
- Message body

Note that some of the messages have missing metadata fields: 102,219 messages (0.46%) have no author identified, and 279 (0.001%) messages have no date specified.

Table 3.4 gives dataset size statistics. Figure 3.8 shows the user and message volume over time. Figure 3.9 shows the message and thread volume distributions.

| | |
|---|---|
| Number of Messages | 22,054,728 |
| Number of Threads | 9,040,958 |
| Number of Sub-forums | 165,358 |
| Number of Unique Users | 3,775,670 |
| Message Date Range | December 1995 - July 2010 |
| Mean (Standard Deviation) Message Length | 94.0 (155.9) tokens |
| Mean (Standard Deviation) Thread Size | 2.4 (4.8) messages |

**Table 3.4:** Ancestry.com Forum Dataset Statistics.

As opposed to the MacRumors.com Forum dataset, in which the subforum hierarchy is largely a topical organization, the subforum hierarchy in the Ancestry.com Forum is organized in three primary dimensions. First, a location-based subforum organization provides a mechanism for authors to associate a geographical location with their threads. Second, a surname-based subforum organization allows authors to associate their threads with an associated surname. Third, a smaller topical organization exists, with topics such as `religion`, `military service` and `adoption`. Several example nodes of the hierarchy are shown below:

```
Localities > North America > Canada > British Columbia > Kootenay
Localities > Western Europe > Belgium > Brussels
Localities > Middle East > Kuwait > General
Surnames > Abigail
Surnames > Efman
Surnames > Graal
Topics > Cemeteries & Tombstones > Europe > France
Topics > Military > American Revolution > Delaware
Topics > Photographs > Vintage Photograph Identification
```

Table 3.5 shows the sizes of these largest three subforum organizations in the collection. Although there are many more `surnames` subforums, they tend to contain many fewer messages and threads. The `localities` and `surnames` each account for slightly less than half the entire online forum thread volume. This primary organization into both a geographic and surname listing occasionally leads to an author posting a question in multiple subforums, referring to two aspects of their question. Although it leads to some redundancy in the collection, this type of organization does correspond to the primary ways users access the data, as we will show below when discussing the query set.

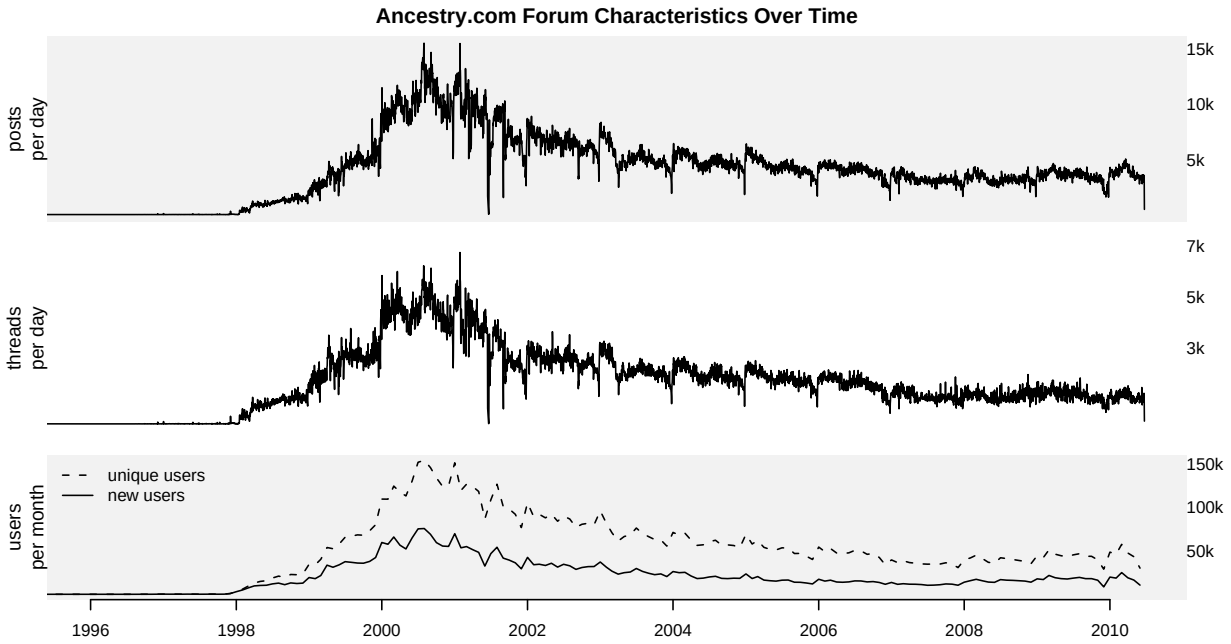



**Figure 3.8:** Ancestry.com forum characteristics over time. From top to bottom: Daily message volume, Daily thread volume, Monthly user volume. Increase in forum activity around 2000 corresponds to Ancestry.com's purchase of RootsWeb (www.rootsweb.ancestry.com), inheriting its online forum and combining the two sites' user bases. The steady decline in forum activity after the peak around 2001 reflects Ancestry.com's focus on providing more genealogical content and site functionality outside of the online forum, and users' decreased reliance on the online forum for performing research. Periodic annual volume dips correspond to December holidays.

The Ancestry.com Forum is significantly larger than other online forum test collections studied for thread search. Table 3.6 shows a side-by-side comparison of the online forum test collections, and similar collections, in previously published studies. The Ancestry.com collection here contains seven times more messages than the next largest collection (the MacRumors.com Forum, described above) and approximately three times more queries than used in other studies for thread search in online forums. Note that some of these collections have been studied for *message search* rather than thread search, but we include them here for purposes of comparison.

### 3.5.2 Query Set

In addition to the document collection, Ancestry.com also provided a set of 10,000 search queries likely to have relevant documents in this collection. The search queries are extracted from Ancestry.com's primary search engine query log. Although this search engine retrieves documents from across the site, not just limited to the online forum, the queries are likely to be representative of the type of information needs that could be served by the online forum documents.

The queries issued to the main Ancestry.com search engine[8] are structured queries, containing different fields for names, locations, dates, or other fields present in their data. Figure 3.10 shows the internal representation of a typical query entered on the main Ancestry.com search engine. Many of these fields refer to the rich annotations generated by Ancestry.com on some parts of their document collection, such as

[8] http://search.ancestry.com/search/

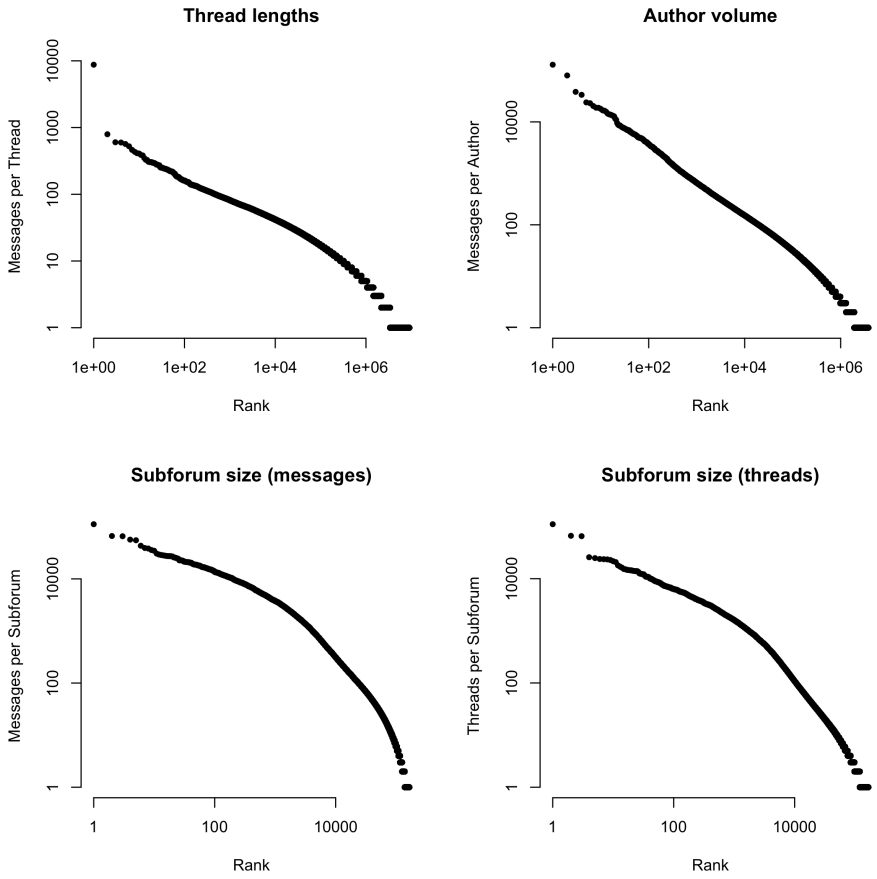



**Figure 3.9:** Ancestry.com Forum size distributions. Top: thread length and author volume distributions. Bottom: subforum size distributions (messages & threads).

census or military records which may have locations, dates and names identified. The different structural fields present in the data are visible in the query, as well as weights placed on those fields in Ancestry.com's internal ranking algorithm. Ancestry.com does not annotate those fields in the online forum documents, and for this reason, we convert the structured queries to keyword queries by extracting only textual information. The query shown in Figure 3.10 would be converted to the keyword query *[martha kekahuna yvonne oahu hi]*. All attempts were made to retain a sensible ordering of query terms, so that order and proximity query operators continue to be effective.

Although we do not directly use the fields in the Ancestry.com queries, we can inspect the query structure to learn about the types of information needs users of the system typically have. Table 3.7 shows an analysis of the 10,000 queries in our query set, and Figure 3.11 shows the frequency of field combinations in the query set. The vast majority of queries (99%) contain a name, and approximately half (49%) contain a name and some other information, either a location or keywords. Although the date field is relatively common, the online forums collection does not have this field annotated and we cannot make use of this information in our experiments. Additionally, it is unlikely that date information would be as useful in the forum collection rather than other documents such as census or military records where publication dates are a primary organizational facet. We sample queries for assessment to conform to this observed distribution, with 97 containing only a person name, and 94 containing a name as well as

| Root node | Num. Subforums | Num. Threads | Num. Messages |
|---|---|---|---|
| `Localities` | 4632 (2%) | 4098253 (45%) | 9151570 (41%) |
| `Surnames` | 158350 (96%) | 4409776 (48%) | 11840781 (53%) |
| `Topics` | 1550 (1%) | 461472 (5%) | 870326 (4%) |
| other | 826 (0.5%) | 74234 (0.8%) | 192051 (0.8%) |

**Table 3.5:** Ancestry.com subforum sizes, showing number of subforums, threads and messages under the largest three top nodes. Percent of total collection shown.

| Collection | # Messages | # Threads | # Subforums | # Authors | # Queries |
|---|---|---|---|---|---|
| Ancestry.com | 22,054,728 | 9,040,958 | 165,358 | 3,775,670 | 191 |
| MacRumors.com | 3,108,244 | 373,639 | 86 | 85,298 | 48 |
| WOW | 1,373,525 | 16,274 | – | – | 30 |
| CANCUN | 529,165 | 58,150 | – | – | 30 |
| W3C | 151,649 | 72,214 | – | – | 110* |
| Microsoft.public | 973,948 | – | – | – | 343* |

**Table 3.6:** Size comparison of the Ancestry.com test collection and other online forums datasets used to study thread retrieval algorithms. WOW and CANCUN statistics reported by Seo and Croft [108], W3C reported by Craswell et al. [39] and Seo and Croft [108], and Microsoft.public reported by Xi et al. [129]. Not all studies report all statistics. Asterisk (*) indicates message search, not thread search.

additional query terms such as location or keyword. We refer to these sets of queries as the *name* and *name+* queries respectively. Table 3.8 shows the query length statistics.

### 3.5.3 Document Pooling

This section describes the process of pooling for identifying those documents to judge with respect to each query. The pooling process has two steps: First, a set of diverse retrieval systems must produce retrieval runs, or document rankings for each query. Second, using those retrieval runs, the most likely relevant documents must be selected for assessment.

| Field Type | Ancestry.com query fields | Query Frequency |
|---|---|---|
| Name | `surname` | 9860 |
| | `given-name` | 9112 |
| | `secondary-surname` | 3864 |
| | `secondary-given-name` | 4255 |
| Location | `place` | 6970 (4642 usable) |
| Other Keywords | `keyword` | 560 (504 usable) |
| Date | `date` | 6923 |

**Table 3.7:** Ancestry.com query set analysis. Note: some field values refer to numeric location identifiers, or other information not usable by our system. These values are ignored for our purposes.

```
(rank
  (given-name
    (weight 100 (field 80004002 martha))
    (weight 90 (field 80000002 martha)))
  (surname
    (weight 100 (field 80004003 kekahuna))
    (weight 90 (field 80000003 kekahuna)))
  (secondary-given-name
    (weight 40 (field 80014002 yvonne)))
  (place 500
    (location 82004010 (place-text oahu, hi))
    (location 82004220 (place-text oahu, hi))
    (weight 80 (location 82004000 (place-text oahu, hi)))
    (weight 60 (location 82000000 (place-text oahu, hi)))))
```

**Figure 3.10:** Ancestry.com sample structured query.

| | Query Length | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| Number *name* Queries | 10 | 52 | 32 | 3 |
| | Query Length | | | |
| | 3-4 | 5-6 | 7-8 | 9-14 |
| Number *name*+ Queries | 15 | 26 | 23 | 30 |

**Table 3.8:** Ancestry.com query length statistics.

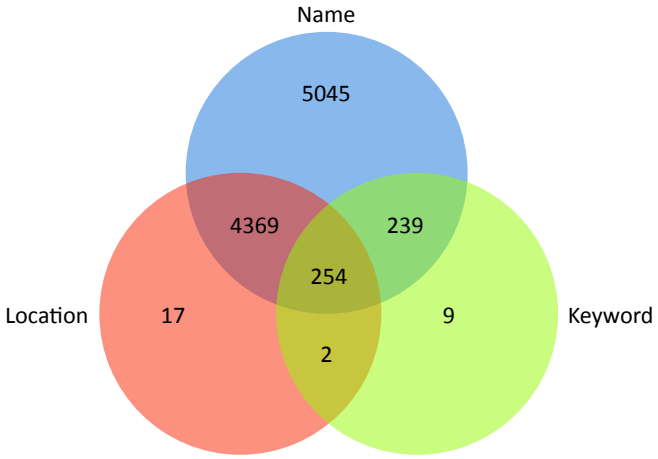



**Figure 3.11:** Ancestry.com Query Field Frequency. Frequency of different query fields in the sample of 10,000 Ancestry.com search queries. Note: only usable field values included in counts.

### Retrieval Runs

Typically in TREC evaluations (for example the Blog track [100, 82, 102]), many different teams with different underlying retrieval systems submit a variety of retrieval runs. With a diverse set of underlying systems and a diverse set of techniques, the resulting rankings are likely to contain a sufficient diversity for a reliable evaluation.

In the absence of a set of runs contributed by different TREC participants, we must simulate a document pool. Our goal in the creation of retrieval runs is as follows:

- First, we aim to diversify the underlying system software in order to avoid overly biasing results towards one indexing or ranking method.
- Second, we aim to include *highly effective* runs and select systems and algorithms that have a history of performing well at TREC.
- Third, we aim to include diverse features of ranking algorithms. For example, we include runs that use only a bag-of-words retrieval model, as well as runs that include fielded retrieval models.

For our retrieval runs, we use four different retrieval system in a variety of configurations, for a total of eight different message ranking algorithms. The systems and message retrieval algorithms used are listed below:

- **Indri**[9] using a variety of query formulations:
  - **Bag-of-words** queries, eg.

    ```
    #combine(john stephen manley)
    ```

  - **Dependence Model** (DM) queries [86], using the suggested model weights, eg.

    ```
    #weight(
      0.8 #combine(john stephen manley)
      0.1 #combine(#1(john stephen) #1(john stephen manley)
                   #1(stephen manley))
      0.1 #combine(#uw4(john stephen) #uw4(john manley)
                    #uw4(stephen manley) #uw8(john stephen manley)))
    ```

  - **Fielded query** with **linear combination**, eg.

[9] Indri version 2.12 (Lemur version 4.12), http://lemurproject.org/

```
#wsum(
  0.4 #weight(
    0.8 #combine( john.(post_title) stephen.(post_title) manley.(post_title) )
    0.1 #combine( #1( john stephen ).(post_title)
                  #1( john stephen manley ).(post_title)
                  #1( stephen manley ).(post_title) )
    0.1 #combine( #uw4( john stephen ).(post_title)
                  #uw4( john manley ).(post_title)
                  #uw4( stephen manley ).(post_title)
                  #uw8( john stephen manley ).(post_title) ) )
  0.4 #weight(
    0.8 #combine( john.(text) stephen.(text) manley.(text) )
    0.1 #combine( #1( john stephen ).(text)
                  #1( john stephen manley ).(text)
                  #1( stephen manley ).(text) )
    0.1 #combine( #uw4( john stephen ).(text)
                  #uw4( john manley ).(text)
                  #uw4( stephen manley ).(text)
                  #uw8( john stephen manley ).(text) ) )
  0.2 #weight(
    0.8 #combine( john.(subforum) stephen.(subforum) manley.(subforum) )
    0.1 #combine( #1( john stephen ).(subforum)
                  #1( john stephen manley ).(subforum)
                  #1( stephen manley ).(subforum) )
    0.1 #combine( #uw4( john stephen ).(subforum)
                  #uw4( john manley ).(subforum)
                  #uw4( stephen manley ).(subforum)
                  #uw8( john stephen manley ).(subforum) ) ) )
```

  - **Fielded query** with **loglinear combination**, similar to the above formulation, but with an outer `weight` instead of a `wsum`

- **Terrier**[10] [99] using two retrieval models, $PL2$ and $InL2$ with the default parameters.
- **Zettair**[11] using the default Okapi BM25 ranking algorithm [60].
- **Ancestry.com**[12] The ranked boolean system used on the main search site of Ancestry.com.

The output of these systems is a message ranking, which must be converted to a thread ranking for our assessment and evaluation. We apply three different aggregation methods to convert each message-ranking to a thread ranking for each system, resulting in a total of 24 different thread rankings in our pool. These aggregation methods are listed below and described in more detail in Section 2.4.2:

- **Mean**: Thread score is the mean score of the retrieved messages, identical to the *document mixture* model with uniform document weights.
- **Max**: Thread score is the max score of the retrieved messages, identical to the *best document* model.
- **Pseudo-Cluster Selection** (PCS): Thread score is the geometric mean of the top-$k$ retrieved messages. In these experiments, $k = 5$.

These aggregation methods were selected as representative of both *inclusive* and *selective* aggregation methods, as well as the methods found superior in preliminary studies with the MacRumors.com Forum dataset. See Chapter 6.2.1 for further details on the performance of these aggregation methods on that

[10] Terrier version 3.0, http://terrier.org/
[11] Zettair Version 0.9.3, http://www.seg.rmit.edu.au/zettair/
[12] Provided by Ancestry.com based on the simplified keyword queries, not the original structured queries.

dataset.

For each system, we retrieved 1000 messages per query, then retained the top 100 threads after converting to a thread ranking. Taking a union of these thread rankings across retrieval runs results in 374 unique threads per query on average.

#### Pooled Document Selection

Give the retrieval runs described above, we then must select those documents most likely to be relevant for judgement. Simple methods for document selection are typically employed at TREC, such as selecting all the top-$k$ documents from each run to judge [125]. This results in a maximum of $n \times k$ documents per query to judge, where $n$ is the number of systems submitting to the pool. At TREC, frequently $k$ is set to 100.

In our case, we do not have the resources to judge a great number of documents, and we prefer to perform a more selective method for choosing documents for the pool. We opt to apply a simple meta-search algorithm, Borda count [8], to prioritize documents retrieved by the systems. This algorithm assigns "votes" to each document based on the rank at which it was retrieved by each system. Borda count assigns a single score to each thread, with documents receiving higher scores if more systems tend to rank them highly.

Given these pooled document scores, we can then prioritize our document selection while assessing in order to favor those more likely to be relevant. Section 3.5.4 describes the algorithm for selecting document pairs for assessment and how this score is used during that selection.

### 3.5.4 Assessment Process[13]

In this section, we describe the process for collecting relevance assessment. Ancestry.com provided personnel to perform the assessment of forum threads. These assessors, although familiar with genealogical research, did not have experience in relevance assessment. Many of the decision made during the relevance assessment process were in an effort to simplify and streamline the assessment task for these assessors who did not have prior experience as information analysts or relevance assessors.

Several approaches to document assessment for retrieval evaluation have been proposed, and for this collection we choose to collect pairwise preferences [31]. Initial research into the collection pairwise preferences shows that collecting the pairwise approach offers several advantages: (1) there is no need to assign an arbitrary ordinal scale to judgement levels, (2) preference capture more document ordering information than coarse absolute judgement levels, (3) assessors can assign preferences faster than absolute judgments, (4) agreement across assessors on preferences is as high or higher than on absolute judgements. The higher levels of agreement and faster assessment times are also an indication that the preference assessment task may be easier for assessors previously unfamiliar with annotating relevance. Additionally, traditional retrieval evaluation measures such as Mean Average Precision and Precision at cutoffs adapted to pairwise preferences correlate extremely highly with the traditional counterparts [29].

Our approach to preference collection is similar to that described by Carterette et al. [31], presenting side-by-side document pairs $(L, R)$ and collecting the assessments:

- Document $L$ is preferred to document $R$ ($L > R$)
- Document $R$ s preferred to document $L$ ($L < R$)
- Document $L$ and document $R$ are duplicates ($L = R$)

[13]The code for the assessment system and document pair selection algorithm is available to download here: https://github.com/jelsas/django-assessment.

- Document $L$ is *bad* (i.e. completely non-relevant)
- Document $R$ is *bad* (i.e. completely non-relevant).

These preferences are essentially a binary judgement of which document is better. We allow assessors to identify *bad* documents and duplicates in order to avoid needlessly collecting redundant preferences or preferences on documents unrelated to the query.

In order to ensure comparability of our collection to ones previously built with pair preferences, our assessment tool is modeled after the one presented by Carterette et al. [30]. See Appendix 7.4.4 for a screenshot of the interface. The assessors who performed the assessment task were provided by Ancestry.com, were trained to use the assessment interface and given the assessment guidelines shown in Appendix 7.4.4.

In addition to the document-pair preferences we collect, we can also use these preferences to infer binary relevance judgements. If a document is ever preferred to any other document for a query, we assume the document is at least minimally relevant. If a document is judged *bad*, we assume the document is non-relevant. Duplicate documents retain the judgement of the other document in the pair, and no assumption is made about documents that are presented to the assessor but never preferred. These assumptions are consistent with the assessor instructions. We will refer to these as *inferred binary judgements* in the discussion below.

### Document Pair Selection Algorithm

The naïve approach to preference assessment would require a quadratic increase in the number of assessments to collect in order to achieve a complete labeling of the documents. Carterette et al. [31] describe several methods for reducing the number of preferences necessary to collect: (1) collecting *bad* judgements and dropping those documents from further assessment, (2) assuming preference transitivity, and (3) ordering preferences to collect based on their utility in distinguishing rankings given an evaluation measure and stopping collection early. The first two methods result in sub-$O(n \ln n)$ judgements per query for a complete labeling. The third method results in an approximately linear number of judgments per query for a complete labeling, but increased complexity in the selection algorithm. We integrate the first two of these suggestions into our pair selection algorithm, but not the third.

We develop an algorithm for active selection of document pairs for assessment adhering to the above principles and several others listed below. These are based in part on learnings from previous work on annotating pairwise preferences [30, 19].

1. After each preference is collected, the preferred document of the pair should be kept in a fixed location in the interface if possible. Fixing a document's location and indicating this to the assessor eliminates the need for the assessor to re-read the document after each judgement.
2. The assessor should be given the opportunity to view the best document for the query as soon as possible, either by exposing the assessor to the entire collection or by showing "better" documents first.
3. Avoid showing the same document to an assessor too many times to avoid assessor fatigue and collect preferences over more diverse documents.

The method developed for active pair selection is presented in Algorithm 1, 2, 3 and 4. Throughout these algorithms, we assume the list of documents $D$ is maintained in some fixed order, with documents more likely to be preferred occurring earlier in the list. We initially order $D$ by the Borda count [8] of the document from the retrieval pool. To avoid overly biasing towards the original retrievals, we then divide $D$ into bins of size 5 and randomize within the bins.

Algorithm 1, AnnotateDocumentPairs(), presents the general framework for active selection of document pairs. In this procedure, we build a list of preference judgements $(L, R, p)$ with $L$ and $R$ documents

**Algorithm 1** AnnotateDocumentPairs(): General document selection strategy for pairwise preference assessment.

**Input:** List of documents $D$ of length at least 2, target number of preferences to collect $m$, the maximum number of appearances of a document $k$ (default $\infty$)
**Output:** List of document pairs and preferences: $\{(L, R, p)|L, R$ documents, $p$ preference$\}$

```
P ← []
while |P| < m do
    if P empty then
        (L, R) ← FreshPair(D, P)
        Collect preference p on pair (L, R)
        Push (L, R, p) onto P
    else
        D' ← {d|d ∈ D, d has been seen ≤ k times and d has not been marked bad}
        (L, R) ← NextPair(D', P)
        if (L, R) = nil then
            return P
        end if
        Collect preference p on pair (L, R)
        Push (L, R, p) onto P
    end if
end while
return P
```

presented on the left and right of the interface, and $p$ a preference value, described above. This algorithm eliminates documents from the assessment pool that have been presented more then a fixed number of times $k$ or have been marked bad.

Algorithm 2, AssessedWith(), calculates the set of documents that already have a preference assigned with a given document. The algorithm is sensitive to a parameter indicating whether we are assuming

**Algorithm 2** AssessedWith(): Algorithm to calculate which other documents have been judged with the given document. Note: this algorithm handles the assumption of transitive or intransitive preferences.

**Input:** A document $d$, and previously collected preferences $P$
**Output:** A set of documents that have been judged with $d$.

```
if Collecting transitive preferences then
    Graph G = {d_i → d_j | d_i preferred to d_j or duplicates}
    Graph G' = {d_j → d_i | d_i → d_j ∈ G}
    Find all reachable nodes in G and G' from node d (eg. via Djikstra's algorithm [38])
    return The resulting set of reachable nodes.
else
    return {d'|(d, d', p) or (d', d, p) ∈ P for any p}
end if
```

preference transitivity. If assuming transitivity, the preference graph and reverse preference graph are constructed and Djikstra's algorithm is run to identify all other documents reachable from the current

document. Due to the use if Djikstra's algorithm, this procedure runs in $O(n^2)$ each time it is called, where $n$ is the number of documents seen so far. It may be advantageous in some cases to pre-compute all paths via the Floyd-Warhsall algorithm [38] ($O(n^3)$) after each preference collection, but we do not investigate that modification here. If not assuming transitivity, the algorithm runs in $O(|P|)$ time.

Algorithm 3, FreshPair(), selects a new unseen document pair for assessment when neither the $L$ or $R$ documents should be held constant from the previous assessment.

**Algorithm 3** FreshPair(): Algorithm to select a fresh pair of documents from the given list of documents and previously assessed preferences.

**Input:** List of documents $D$ of length at least 2, previously collected preferences $P$
**Output:** A pair of documents $(L, R)$ or *nil* if no pairs are available.

```
for all L ∈ D do
    D' ← D \ AssessedWith(L, P)
    if D' not empty then
        return (L, D'[0])
    end if
end for
return nil
```

Algorithm 4, NextPair(), attempts to select a new document pair when either the $L$ or $R$ documents are held fixed from the previous assessment. If no new pair can be identified, this procedure falls-back to calling the FreshPair() procedure (Algorithm 3).

### 3.5.5 Pilot Assessment & Assessor Analysis

In order to validate our approach to preference assessment, and to test whether previous findings on preference assessment hold for our assessors and test collection, we conduct a pilot assessment of 50 queries each annotated by two assessors. We do not assume transitivity in the document pair selection algorithm for this study in order to test the hypothesis that judgements are transitive. We collect up to $m = 100$ document pairs for assessment in this pilot. We also do not place a limit on the number of times a document can be shown to the assessor, letting $k = \infty$ in Algorithm 1 above.

We evaluate three questions with the pilot assessment:

1. What level of *external* agreement for pairwise preferences exists between assessors? I.e. if one assessor expresses a preference, does the other assessor agree?
2. What level of *external* agreement for inferred binary relevance exists between assessors?
3. What level of *internal* agreement exists among a single assessors preferences? I.e. does an assessor produce transitive preferences?

The agreement shown in Tables 3.9, 3.10 and 3.11 aim to answer these questions.

In Table 3.9, we show for each pair of documents $(A, B)$, the preference label assigned by the two assessors. Only explicit preferences are considered in this table, and no transitivity is assumed. We do make the assumption that documents that have ever been preferred for that query are preferred to any marked *bad*. The overall agreement between the two assessors (excluding the "no pref" row and column) is 22.6%. This, however, counts equally all judgement decisions, whether they are for an expressed preference or an assessor deemed the two documents not relevant for the query. One assessor, corresponding to the column counts, is clearly much more aggressive in identifying *bad* documents. If the two assessors

**Algorithm 4** NextPair(): Algorithm to select the next pair of documents for assessment.

**Input:** Document list $D$, non-empty previously collected preference list $P$
**Output:** Pair of documents $(L, R)$ for assessment, or *nil* if no more documents pairs are available.

```
(L, R, p) ← peek at last item in P
if p = L preferred or R bad or L and R duplicates then
    if L has been seen ≥ k times then
        return FreshPair(D, P)
    end if
    D' ← D\ AssessedWith(L, P)
    if D' empty then
        return FreshPair(D, P)
    else
        return (L, D'[0])
    end if
else // In this case, p = R preferred or L bad
    if R has been seen ≥ k times then
        return FreshPair(D, P)
    end if
    D' ← D\ AssessedWith(R, P)
    if D' empty then
        return FreshPair(D, P)
    else
        return (D'[0], R)
    end if
end if
```

| | | Assessor 1 | | | | |
|---|---|---|---|---|---|---|
| | | $A < B$ | $A > B$ | $A, B$ bad | $A = B$ | no pref |
| Assessor 0 | $A < B$ | 381 | 126 | 1078 | 2 | 856 |
| | $A > B$ | 143 | 304 | 986 | 10 | 0 |
| | $A, B$ bad | 0 | 0 | 1 | 0 | 0 |
| | $A = B$ | 2 | 0 | 1 | 0 | 1 |
| | no pref | 185 | 241 | 1744 | 2 | 342 |

**Table 3.9:** Preference inter-assessor agreement for all document pairs $A$, $B$ summed across all 50 queries used in the pilot assessment. Note: Due to active selection strategy, not all pairs shown to both assessors and a document pair may be presented in opposite order.

have differing thresholds for declaring a document *bad*, as we see here, a preference between documents stated by one assessor may appear as two bad documents to another assessor.

We can look at this problem of pairwise agreement between assessors from an analytical perspective and calculate a bound on the maximum preference agreement two assessors will have if one is more aggressive in in marking *bad* documents. Assume, for example, both assessors perfectly agree on the total orderings of $n$ documents. One assessor has a more conservative view of the information need and marks some fraction $s$ of documents *bad*, $0 \leq s \leq 1.0$, while the other assessor marks no documents as bad. If $s = 0$, then agreement is 100%, but if $s > 0$, pairwise agreement will suffer even though both assessors agree on the total ordering of our documents. The overall agreement between these two assessors is given by:

$$\text{Agreement} = \frac{\binom{n}{2} - \binom{ns}{2}}{\binom{n}{2}} = \frac{n(n-1) - ns(ns-1)}{n(n-1)} = 1 - \frac{ns^2}{n-1} + \frac{s}{n-1}$$

where we have $\binom{n}{2}$ total document pairs and $\binom{n \times s}{2}$ document paris marked both *bad* by one assessor. As $n \to \infty$ the agreement between our assessors approaches $1 - s^2$ (and is quite close to $1 - s^2$ for values of $n \approx 40$ as in our dataset). In our case, one assessor assigns effectively no documents a *bad* label, and the other marks *bad* roughly 64% of the document seen (see Table 3.10). This places an upper bound on their agreement at 59%, assuming they perfectly agree on the absolute ordering of all the document seen. The observed agreement of 22.6% is roughly 39% of this upper bound.

Although this discrepancy exists between our assessors' willingness to assign a *bad* label, it primarily indicates the assessors had differing interpretations of the specificity of the information need. Our primary concern is agreement between assessors when they both express a definite preference, shown in the shaded cells in the table. When only considering these cells of the table the agreement is much higher at 71.8%. This is an acceptable level of agreement for preferences, and comparable to previous levels of agreement observed for preference assessment [31]. The discrepancy in assessors' willingness to assign a *bad* judgement was addressed in further assessor training and clarification of the assessment guidelines.

Table 3.10 shows agreement on inferred binary judgements. Based on the differences observed above in how the assessors label *bad* documents, we expect to see a lower level of agreement when looking at inferred binary relevance. This is the case here, where we see a 36.2% overlap in relevance judgements (size of the intersection of relevant documents divided by the union). This level of agreement is at the lower end of the range of agreement levels for absolute judgements observed in other studies using more experienced assessors [31, 122], and well below the level of agreement we observed on preferences.

Finally, we must validate whether the assessors produce internally consistent judgements – that is, whether the assessors' preference judgements are transitive. To calculate the fraction of an assessor's

| | | Assessor 1 | |
|---|---|---|---|
| | | relevant | non-relevant |
| Assessor 0 | relevant | 333 | 591 |
| | non-relevant | 2 | 3 |

**Table 3.10:** Inter-assessor agreement on inferred binary judgements, counts summed across queries. Those documents that have ever been preferred are considered relevant, those documents marked *bad* are considered non-relevant. No assumption is made on documents that are presented but never preferred.

judgements that are transitive, we compute for each (assessor, query) pair a statistic $t$, the fraction of document triples that the assessor judged transitively:

$$P = \{(A, B, C) | \text{ Assessor prefers } A > B \text{ and } B > C\}$$
$$T = \{(A, B, C) | (A, B, C) \in P \text{ and Assessor preferrers } A > C\}$$
$$t = \frac{|T|}{|P|}$$

The $t$ values, macro-averaged across queries, are shown in Table 3.11. Additionally, we show the number of queries queries that each assessor assessed perfectly transitive, i.e. with $t = 1.0$. In this table, we can

| | **Macro-Average $t$** | **Queries Perfectly Transitive** |
|---|---|---|
| Assessor 0 | 0.991 | 24 |
| Assessor 1 | 0.999 | 48 |

**Table 3.11:** Internal consistency of assessors' judgements over 50 queries. $t$ measures the fraction of document triples assessed transitively.

see that the assessors' judgements are almost perfectly transitive. Again, this finding agrees with previous work [31] and indicates that we can assume preferences are transitive when collecting assessment on more queries.

### 3.5.6 Additional Assessment

Based on learnings from the pilot study, we conduct an additional assessment of 141 other queries, each assessed by one assessor. We adjust the parameters of the pair selection algorithm based on the following findings of the pilot study:

- Preference annotations are almost perfectly transitive.
- Annotating 100 document pairs per query leads to assessor fatigue.
- If a highly preferred document was presented early in the assessment, this document tended to be presented as one item of the pair in all the remaining pairs.

Based on these findings, we assume transitivity in the document pair selection algorithm, and collect up to $m = 60$ document pairs to be assessed per query. We also limit the number of times a single document can be presented, setting $k = 5$ in the selection algorithm. Table 3.12 shows the assessment statistics for the *name* and *name+* queries.

| Query Set | Num. Docs Seen | Pairs Assessed | Num. Bad | Num. Duplicate |
|---|---|---|---|---|
| *name* | 44.10 | 69.16 | 9.68 | 0.29 |
| *name+* | 46.09 | 69.79 | 20.89 | 0.24 |

**Table 3.12:** Ancestry.com assessment statistics per query set. Statistics shown averaged across queries in each set.

### 3.5.7 Pairwise Preference Evaluation[14]

The collection of pairwise preferences necessitates evaluation measures computed over those preferences, rather than traditional absolute judgements. Previous work has proposed several analogues to absolute evaluation measures computed over preferences [29]. We use most of these measures in our evaluation, with some modifications to $APpref$, described below.

In all these measures, we assume the set of preferences, $P$, is given:

$$P = \{(A, B) | A, B \text{ documents s.t. } A \text{ is preferred to } B\}. \tag{3.1}$$

This set of preferences is either explicitly given by the assessors, or implicitly through an assumption of transitivity, for example. Each document in the set of preferences has an associated rank assigned by the ranking algorithm, $\rho(A)$. We can then define the set of *correctly ranked* preferences:

$$P_{\text{corr}} = \{(A, B) | (A, B) \in P \text{ and } \rho(A) < \rho(B)\} \tag{3.2}$$

where the ranking of the documents corresponds with the preferred ordering.

The analogue to Precision at a cutoff ($P@k$) is *ppref@k*, the fraction of correctly ordered preferences where at least one documents in the pair is ranked above $k$. Formally, this is given by:

$$ppref@k = \frac{|\{(A, B) | (A, B) \in P_{\text{corr}} \text{ and } \min(\rho(A), \rho(B)) \leq k\}|}{|\{(A, B) | (A, B) \in P \text{ and } \min(\rho(A), \rho(B)) \leq k\}|}. \tag{3.3}$$

Similarly, an analogue to Recall at cutoffs ($R@k$), *rpref@k*, is defined as the fraction of *all* preferences that meet the criteria above. Formally, this is given by:

$$rpref@k = \frac{|\{(A, B) | (A, B) \in P_{\text{corr}} \text{ and } \min(\rho(A), \rho(B)) \leq k\}|}{|P|}. \tag{3.4}$$

An analogue to Average Precision (AP) over preferences, *APpref*, has also been proposed. Carterette et al. define this as "ppref [...] averaged over ranks at which rpref increases" [29]. This definition, however, is somewhat problematic. First, *ppref* and *rpref* values change at different ranks, so some *ppref* values may not get factored into the calculation. Second, because some *ppref* values may be ignored, it is possible to construct a ranking with some incorrectly ordered pairs but achieves a "perfect" *APpref* of 1.0. Third, the original definition does not state how to treat unranked pairs, whereas typically Average Precision calculations factor in a minimum precision value for all unranked relevant documents [84].

We propose an alternative formulation of *APpref* that captures all *ppref* values regardless of whether *rpref* changes at that rank level. First, we define the set of *preferred* documents $P^+$ as those that ever been preferred to any other document:

$$P^+ = \{A | (A, B) \in P\}. \tag{3.5}$$

[14]The code for evaluation of retrieval system output with pairwise preferences is available for download here: https://github.com/jelsas/Pairwise-Preference-Evaluation

Note that the ranks of documents in $P^+$ are the only ranks where *rpref* can change, although it does not necessarily change at all those ranks. We then average $ppref$ values over the ranks of all preferred documents to calculate $APpref$:

$$APpref = \frac{\sum_{A \in P^+} ppref@\rho(A)}{|P^+|} \tag{3.6}$$

If we define $\rho(A) = \infty$ for those documents unranked by the retrieval system, excluding those pairs from the set $P_{\text{corr}}$, this definition of $APpref$ behaves similarly to Average Precision for unranked documents.

## 3.6 Conclusion

In this chapter, we presented the collection used in the thesis experiments. First, we briefly presented the BLOG06 collection. Next, we discussed online forum collections and present our work building collections for use in the experiments in following chapters. We presented the structure of online forums, and gave insight into why these are interesting dataset for information retrieval evaluations. We also presented the construction of two IR test collections to study thread retrieval in online forums: the MacRumors.com Forum and the Ancestry.com Forum. These datasets share many structural characteristics, and lend themselves to the study of how collection structure can be used in retrieval algorithms.

This chapter presented two major research contributions. First, we developed a novel technique for identifying information needs and relevant documents from within the collection itself. In the case of the MacRumors.com Forum, linking across forum threads is a frequent occurrence. In many cases, one author will reply to a question with a link to another thread containing the answer. This question message/answer thread pair can be viewed as an information need and a relevant document. We exploit these relationships to build the MacRumors.com Forum test collection.

Second, we present another IR test collection for studying thread retrieval in online forums, the Ancestry.com Forum. This dataset differs from the MacRumors.com Forum in several ways: (1) it was collected with cooperation of the website owners, (2) it is significantly larger, (3) the associated queries are sampled from a query log rather than manually extracted from messages, and (4) relevance judgements were collected from assessors rather than extracted from interactions in the collection. To our knowledge, this dataset is the largest online forum IR test collection ever used in the academic setting. We plan on releasing this test collection to the academic community, which will make it the only publicly available test collection for studying information retrieval in online forums.

# Chapter 4

# Blog Post Search

In this chapter, we turn our attention to the task of blog post search. Blog collections are interesting from the perspective of collection structure because they provide a single hierarchical dimension of organization. All blog posts belong to a unique blog, and blogs are typically written by a single author, which we refer to as the blogger. Thus, this task gives us a mechanism to evaluate the utility of blogger-level evidence as a feature in post retrieval algorithms.

Although we don't have statistics for how many blogs in our collection have contributions from multiple authors, research on other collections indicates that 10% or less of blogs have more than one author [57]. Based on this information, we may find that our findings with regard to blogger-level features in blog search generalize to other collections where authorship information is available.

### Chapter Roadmap

Our exploration of the blog post search task in this chapter is organized as follows:

- In **Section 4.1** we describe the Blog Post Search task.
- In **Section 4.2** we present an overview of our approach, which focuses on how to incorporate evidence from the text of the blog as a whole into the post scoring. We specifically focus on *expert bloggers* who have contributed a disproportionate number of posts judged relevant. We first present our baseline algorithm. Then we develop and analyze an *oracle model* that learns optimal blog-level weights to maximize post-ranking performance. In this section we show that, while the expert bloggers tend to have a high weight in the oracle model, the general trend is for more prolific bloggers to receive a negative weight. This insight highlights the risk-reward tradeoff in incorporating blogger-level content into ranking and informs the development of post ranking models.
- In **Section 4.3** we present methods to leverage blog-level evidence in the language modeling retrieval framework, focusing on blog post smoothing or document-expansion methods. We analyze the performance of these methods along several dimensions: overall performance, the diversity of the retrieved blogs, and the effect of these models on posts written by the expert bloggers. We find that language modeling approaches to incorporating blogger-level evidence, while improving the ranking of posts written by expert bloggers, fail to improve overall post retrieval performance.
- In **Section 4.4** we look at integrating blog-level evidence in feature-based retrieval models, and introduce models that can incorporate *negative evidence* from the blogger. We also develop a more flexible feature-based model that differentially weights bloggers based on whether or not they have been identified as an expert. We find that these feature-based models can consistently and significantly improve post retrieval performance.

- In **Section 4.5** we aim to explain *why* the retrieval models presented in this chapter behave the way they do. To do this, we elaborate on the risk-reward tradeoff when incorporating blogger-level evidence in post ranking, and give specific examples of the queries that are helped and hurt as well as the bloggers that are most affected by these models.
- In **Sections 4.6** and **4.7** we present related work and conclude the chapter, giving directions for future work.

#### Research Contributions

The major contributions of the work presented in this chapter are as follows:

- We present the first thorough exploration of using evidence from the text of the blog as a whole in post ranking.
- We develop an effective feature-based model incorporating evidence from the blog as a whole that provides significant and consistent improvements over a strong baseline blog post ranking model.
- We analyze and discuss the role of diversity of blogs in post ranking, particularly with regard to mitigating the risk of favoring prolific bloggers.

## 4.1 Task Description

The blog post search task has been studied in the context of TREC for three years [82, 102, 100]. This is a tasks of retrieving blog *posts*, typically corresponding to a single editorial written by a *blogger*. This task differs from the blog feed retrieval task also studied at TREC (see Chapter 5 for details), which looked at retrieving entire blogs, i.e. the collection of all posts written by a blogger.

The document collection used at TREC, and throughout the experiments here, is the BLOG06 collection [78] described in Section 3.1. All experiments presented below use an index made of only the permalink (HTML) portion of the collection, with no other preprocessing of the documents. This approach is common among the participants in the TREC evaluations, but does have several drawbacks. First, in addition to the blog post text, the reader-submitted comments are also included in the index. Second, the index includes posts from non-English and Spam blogs, which are likely judged non-relevant by the assessors. Third, we perform no content analysis of the blog posts, so that our index includes boilerplate content such as site navigational text, copyright footers, and "blogroll" links repeated on each post of a blog. Although these factors likely affect blog retrieval performance, we take a straightforward approach to indexing the collection that is in keeping with most previous work at TREC. We discuss the effect of some of these factors on retrieval performance below in Section 4.5.

The blog post retrieval task at TREC is designed as a forum for evaluating opinion finding and polarity classification algorithms in addition to ad-hoc post search. The query set includes topics likely to attract opinionated discussion online, mostly involving politics, products, or cultural events. Fifty queries were assessed in each of the three years of the TREC Blog Track, and we will use all 150 of these queries in the experiments below. In the 2008 Blog Track, the last year this task was run at TREC, the systems were evaluated against the 50 new queries for that year, and against all 150 queries from the three years together. For this reason, we report results on each year's queries separately, and on the average performance across all 150 queries.

TREC provides multiple levels of evaluation with the blog post retrieval task data, including the judgement levels *non-relevant*, *relevant*, *relevant and positively opinionated*, *relevant and mixed opinionated* and *relevant and negatively opinionated* [101]. This tiered judgement scheme allows multiple uses of the

resulting test collection. For our purposes, we are only interested in the topical retrieval aspect of the task, and will ignore the opinion and polarity document annotations. We treat all documents labeled as *relevant* (whether opinionated or not) as relevant for our purposes.

## 4.2 Overview of Our Approach

We look at the task of blog post search solely for the purposes of evaluating the efficacy of blogger-level evidence in blog post ranking. In order to do this, we evaluate several techniques to integrate this evidence into the ranking function.

There are two primary goals of the inclusion of blogger-level evidence in post ranking. First, we may want to promote posts from bloggers who seem to be generally high-quality authors or particularly knowledgeable about the topic of the query. Second, we may want to demote posts from authors who may not be experts, or generally produce low-quality posts.

### 4.2.1 High-Quality Bloggers

To understand the implications of blog-level evidence in ranking, we look at the distribution of relevant posts across blogs in the collection. Figure 4.1 shows the distribution of relevant posts over blogs for a selection of queries. As we can see, there is a very skewed distribution, with some bloggers contributing a disproportionally high fraction of the relevant posts and most bloggers contributing very few. We have indicated those bloggers who have contributed more than 5% of the relevant posts for a query with the circled datapoints.

When considering bloggers to favor in post ranking, those bloggers at the top ranks in Figure 4.1 (to the left) may be good candidates. If we can infer that they are likely to be good sources of relevant documents, favoring their posts may lead to improved post retrieval performance. We can consider the task of identifying high-quality bloggers as similar to two other tasks studied at TREC: expert finding and blog feed retrieval. See Chapters 2 and 5 for details on those tasks.

Throughout the experiments below, we show specific analysis of those bloggers who have written more than 5% of the relevant posts for a query. In keeping with similar previous research, we refer to these as "expert bloggers", and one of our hypotheses is that favoring these bloggers will result in an improved post retrieval performance. Across all years of TREC queries, approximately 70% of the queries have at least one blog that contributes more than 5% of the total relevant posts.

Although we use the term "expert" in this chapter to refer to these preferred authors, we do not necessarily consider the number of relevant posts written as a true measure of expertise. Other features may be important to identify true experts, such as a deeper topical analysis of the posts written, evaluation of how those posts are referenced or linked to by other bloggers, and possibly a consideration of the author's experience or education. In this work, we use the term "expert" only to identify those bloggers who have written a particularly large number of relevant posts.

### 4.2.2 Baseline Model

Throughout the following experiments we use a strong baseline retrieval model based on the language modeling framework. Our baseline model is Metzler's full dependence model, using the n-gram and term window weights as suggested in the literature [86]. See Section 2.1.1 for the details of this model. The baseline uses only the topic titles, which are most similar to short keyword queries typical in a web search scenario.

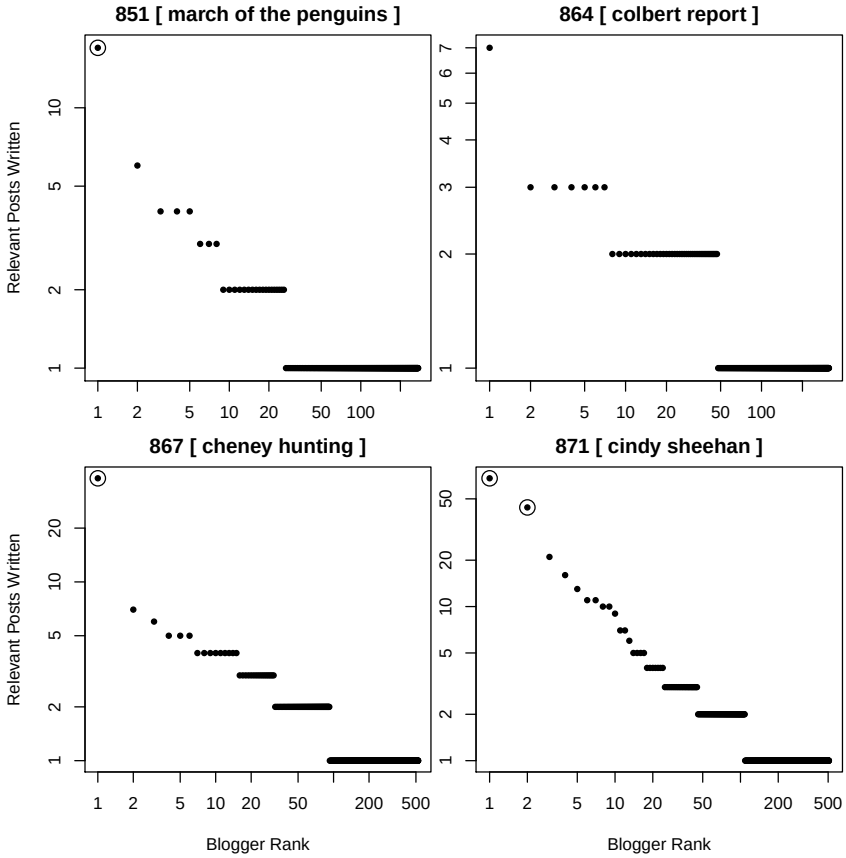



**Figure 4.1:** Distribution of relevant posts across bloggers for a sample of four queries from the TREC 2006 blog track. Shown on a log-log scale. Circled datapoints represent bloggers who have contributed more than 5% of the total relevant posts for that query.

The performance of this model with Indri's default smoothing parameters across all the three years of the TREC Blog Track is shown in Table 4.1. Although the baseline model we use does not always perform as well as the *best* submissions at TREC, its well above the median run performance and does not use additional topic fields, query expansion, or specialized collection preprocessing techniques.

### 4.2.3 Oracle Experiments

One assumption with this investigation into blog post search is that there exist *some* features specific to the blogger that are beneficial to post ranking. To validate the existence of these features, we perform a series of oracle experiments using a simple linear model that adjusts the baseline retrieval model score with a blog-specific score. This oracle model for a single query is given by the following formulation (using matrix-vector notation):

$$\mathbf{f}_Q = \mathbf{y}_Q + \mathbf{B}\mathbf{a}_Q \tag{4.1}$$

where $\mathbf{y}_Q$ is a vector of the original query scores for each post, $\mathbf{B}$ is a blog-by-post binary affinity matrix indicating which posts belong to each blog such that $\mathbf{B}_{ij} = 1$ if post $P_i$ belongs to blog $B_j$, and $\mathbf{a}_Q$ is a vector of blog weights to be learned. Equation 4.1 shows the model for a single query, and below we

| TREC Year | Baseline MAP | TREC Best MAP | TREC Median MAP |
|---|---|---|---|
| 2006 | 0.3405 | 0.2959 | |
| 2007 | 0.4058 | 0.6382* | 0.3340* |
| 2008 (50 new queries) | 0.3967 | 0.4954** | 0.3529* |
| 2008 (all 150 queries) | 0.3810 | 0.5847*,** | 0.3551*,** |

**Table 4.1:** Baseline model performance over three years of TREC Blog Search queries, compared to the best topical retrieval performance of title-only run from that year's TREC submissions. Note: Title-only run performance is not always reported in TREC proceedings. Performance marked with an asterisk (*) are the best/median overall run performance, using any topic field. Performance marked with two asterisks (**) utilized extensive corpus preprocessing and pseudo-relevance feedback. Best and median TREC results reported in the 2006 [100], 2007 [82] and 2008 [102] overviews.

extend the model to many queries, learning weights for each blog independently on every query. This allows a single blog to be favored on one query and not on another, for example.

We treat the learning of this vector of blog weights $\mathbf{a}_Q$ as a simple learning-to-rank problem. For this analysis, we use the entire set of 150 queries and relevance judgements from TREC 2006-2008 as our "training" data. Because we are interested in showing the existence of some blog-level feature weight that improves performance, we do not consider a test set and focus only on training set performance.

Using RankSVM [59], we train a linear model to minimize the number of times a relevant document is ranked below a non-relevant document in the training set. Minimizing the number of swapped pairs is known to maximize a lower-bound on MAP and many other common IR performance metrics [47]. The regularization parameter $C$ we set to a low value, eg. $C = 1$, to let the model focus on minimizing swapped document pairs and not restrict the range of the learned parameters. In effect, we are intentionally over-fitting the model to the training data to evaluate the potential effectiveness of blog-level features.

It is important to note that the training procedure will always place a relatively high weight on blogs with very few posts that are all judged relevant to the query. There is never a risk of assigning a high weight to a blog with only one relevant post, but there can be considerable risk in assigning a higher weight to a blog with a mix of relevant and non-relevant (or unjudged) posts retrieved. This issue of risk with respect to the blog post search task is a central theme in the analysis of the retrieval models discussed in the sections below.

To train the model, we construct a training set with posts-query pairs as training instances. Each training instance has exactly two non-zero features: the baseline retrieval score, and a binary feature indicating which blog the instance belongs to. Note that this second feature is unique per-query, so that if the same post is retrieved multiple times for different queries, it will not only appear multiple times in the training set, but also have completely distinct feature values.

Formally, these training vector $\mathbf{x}_{qi}$ for each query $q$ and retrieved document $i$ are given by:

$$\begin{aligned}\mathbf{x}_{qi} = \langle & y_{qi}, b_{i1}^{q_1}, b_{i2}^{q_1}, \ldots, b_{iM}^{q_1}, \\ & b_{i1}^{q_2}, b_{i2}^{q_2}, \ldots b_{iM}^{q_2}, \\ & \ldots, \\ & b_{i1}^{q_{150}}, b_{i2}^{q_{150}}, \ldots b_{iM}^{q_{150}} \rangle\end{aligned}$$

where $y_{qi}$ is the baseline retrieval score on document $i$ for query $q$ and $b_{ij}^{q_k}$ is a binary feature:

$$b_{ij}^{q_k} = \begin{cases} 1 & \text{if post } i \text{ belongs to blog } j \text{ and } q = q_k \\ 0 & \text{otherwise} \end{cases}$$

The training matrix is illustrated for two queries in Figure 4.2. Although each training instance contains

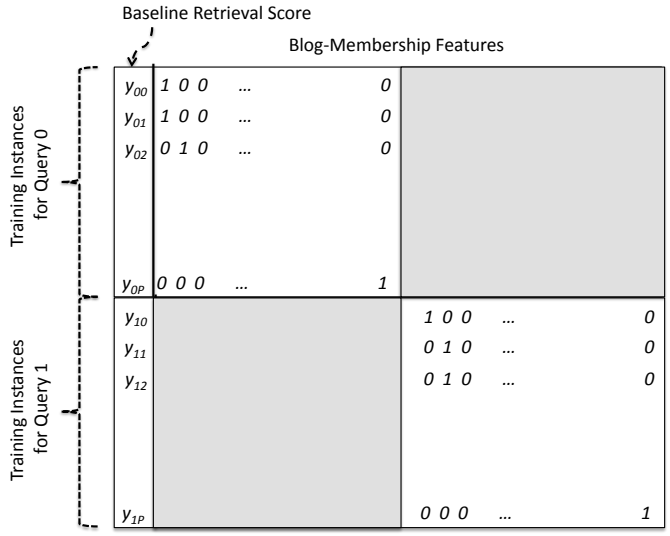



**Figure 4.2:** Illustration of training data matrix for the oracle model. Training data for two queries are shown for simplicity. Shaded areas indicate all zero-valued features.

only two non-zero features, the final training set is extremely sparse and contains 55644 total features. Note that these features *b only* indicate blog membership, and do not describe any interesting features of the blog itself or the content of the blog.

The results of the oracle model are given in Table 4.2. These results show 80-90% improvement in

| TREC Year | MAP | % Improvement over Baseline |
|---|---|---|
| 2006 | 0.6333 | 85.99% |
| 2007 | 0.7233 | 78.24% |
| 2008 (50 new queries) | 0.7688 | 93.80% |
| 2008 (all 150 queries) | 0.7085 | 85.95% |

**Table 4.2:** Results of the oracle model over all TREC Blog Track years. All improvements are significant at the $p < 0.0001$ level.

performance over the baseline retrieval model. This validates that, given perfect knowledge of relevant blog posts, there exists some blog-level feature that significantly improves retrieval performance.

Looking more closely at the oracle model, we may be interested in which types of blogs are favored or disfavored with the learned weights. Figure 4.3 shows the learned blog weight (i.e. the values of the $\mathbf{a}_q$ vector from formula 4.1) for a selection of queries. The pattern of the blog weights for these queries is typical across most queries in the dataset. As expected, the general trend is towards a decreasing (negative) weight on blogs with more retrieved posts, even when those blogs have some relevant posts. This reflects the fact that there is no risk in favoring short blogs with mostly relevant posts, but a greater risk in favoring blogs with a mix of retrieved relevant and non-relevant posts.

The more interesting blogs are those with more posts *and* higher weight. These are blogs in which the blogger may have written more than one relevant post, and could be considered an expert on the topic of the query. Because of the larger amount of data produced by the blogger and retrieved, we may be able to

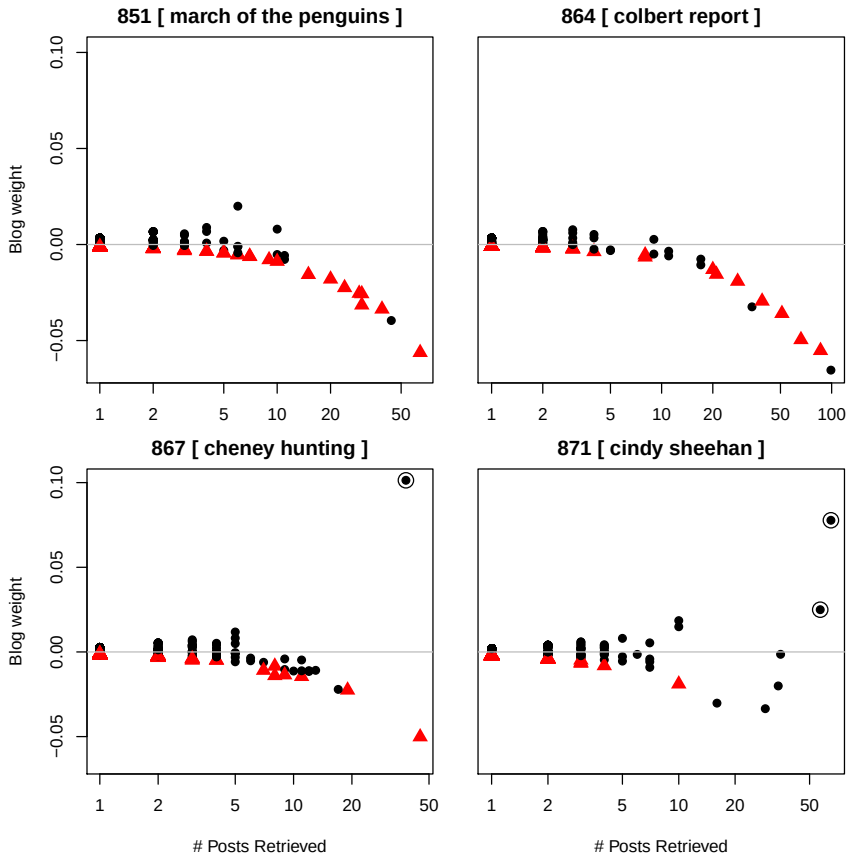



**Figure 4.3:** Oracle model blog weight vs. the number of posts retrieved in the baseline model for a sample of queries. X-axis in log-scale to show the low-frequency distribution. Red triangles are blogs with zero judged relevant posts. Circled datapoints represent blogs which contain greater than 5% of the total relevant posts for that query.

deduce from the content of those blogs that the blogger is an expert on the topic. All the queries shown in Figure 4.3 have some blogs that meet this criteria. Although the overall trend is to disfavor blogs with more retrieved posts, there is a small positive blog weight for some blogs with between 2 and 10 posts.

The bottom two queries in the figure also have a handful of blogs with more than 20 retrieved posts and a considerable positive weight on the blog. In fact, these highly weighted blogs are the "expert bloggers" we defined above, who have contributed more than 5% of the total relevant posts for the query. We indicate the expert bloggers by the circled data points in the figure.

The above observations give us insight into how we may want to identify blogs to favor or disfavor when ranking posts. On one hand, the blogs with an exceptionally high weight and a large number of likely relevant posts may represent "expert" bloggers and favoring those bloggers could yield some performance improvements. On the other, the general blog-weighting trend is towards disfavoring blogs with more retrieved posts. These two competing factors in the blogger weights seen in the oracle model summarize the potential risk and reward of favoring prolific bloggers.

## 4.3 Blog Structure in Language Modeling Retrieval Models

One natural method to introduce blog-level evidence in language modeling framework is through two-level smoothing models. Several two-level smoothing models have been proposed throughout the IR literature and applied to problems such as field retrieval in structured documents [137] and cluster-based retrieval [65]. Typically, smoothing is viewed as a method to improve probability estimation, but in the context of post retrieval we can view two-level smoothing as a document expansion technique. In this context, some of the probability mass from the blog is used to estimate the post-level term generation probabilities.

As shown above, some bloggers write a large number of relevant posts. To the extent that the blogger language model can identify these "expert bloggers", incorporating this evidence into the post language model may improve performance. This section explores this hypothesis by applying smoothing models known to be effective for other retrieval tasks to the task of blog post retrieval.

### 4.3.1 Smoothing with Blog Structure

We consider two different models of two-level smoothing in the language modeling framework. These models, two-level Jelinek-Mercer and two-level Dirichlet smoothing, are discussed in detail in Section 2.1.2, and we apply them as described here. In this case, the document aggregate is the blog, comprised of individual blog posts. Both of these models have two parameters that need to be fit based on performance on training data. Unless otherwise noted, we fit these parameters with three-fold cross validation via a grid search to maximize Mean Average Precision (MAP), using the natural annual query sets to define the test folds and evaluating approximately 150 parameters settings for each model. Both of these models gracefully degrade to standard collection-level smoothing models with adjustment of the smoothing parameters. In Jelinek-Mercer smoothing, as the blog-level weight $\lambda_B \rightarrow 0$, evidence from the blog is ignored. Similarly in two-level Dirichlet smoothing, as $\mu_C \rightarrow +\infty$, the influence blog-level evidence is diminished.

### 4.3.2 Experiments

In this section we describe two experiments applying the smoothing models to blog post search. First, we validate the assumptions made by the two-level smoothing models, that representing the posts from the blog as a single large document is an effective method to identify "expert bloggers". Next, we apply the two-level smoothing models to the task of post retrieval.

#### Validation of Blogger Retrieval

Some bloggers generate a large fraction of the relevant documents for some queries. In fact, almost 70% of the queries have at least one blogger who generates more than 5% of the relevant documents. In this section, we investigate whether standard language modeling techniques can effectively identify those bloggers.

The two smoothing models presented above treat the blog as a single bag-of-words and estimate the blog-level evidence from this language model. Work on blog feed retrieval, described in Chapter 5, proposes several methods of modeling a blog as a collection of documents rather than treating the blog as a single large document. The work described there, and in other work on blog feed retrieval has found that this large-document approach is effective [15, 45]. In order to validate that in the case here, we evaluate the effectiveness of expert blogger ranking with a variety of these models.

To do this, we construct a dataset with blog-level relevance information from the post-level judgements, considering a blog to be relevant if at least 5% of the relevant posts belong to that blog. This task is similar to the Blog Feed Search task discussed in Chapter 5, but here we are concerned about the quality of the blog ranking only as an intermediate step towards our final post ranking.

In Table 4.3 we show the results of this evaluation using several blog search models for this task. The models used are all described in detail in Section 2.4.2. Here, we use the following document aggregate ranking methods in order to explore a variety of methods found to be effective for other tasks:

- A large document model (**LD**), identical to the document concatenation model,
- A linear mixture model over documents (**SD**$_{sum}$), identical to Balog's Model 2,
- A log-linear document combination (**SD**$_{prod}$), identical to the Pseudo-Cluster Selection model, and
- A model that scores blogs by a single post score (**SD**$_{max}$), identical to the best-document model.

| TREC Year | Number of Queries | LD | SD$_{sum}$ | SD$_{prod}$ | SD$_{max}$ |
|---|---|---|---|---|---|
| 2006 | 33/50 | **0.2974** | 0.2564 | 0.2564 | 0.2164 |
| 2007 | 39/50 | **0.3313** | 0.3083 | 0.2627 | 0.2527 |
| 2008 (50 new queries) | 32/50 | 0.3263 | **0.3319** | 0.3006 | 0.2433 |
| 2008 (all 150 queries) | 104/150 | **0.3190** | 0.3089 | 0.2724 | 0.2383 |

**Table 4.3:** Blog retrieval performance, Mean Average Precision for a variety of blog retrieval models (see Chapter 5 for model details). Evaluation considers blogs containing at least 5% of the relevant posts to be relevant for the queries. Note: not all queries have at least one blog meeting this criteria.

In this table, we can see that the majority of the queries have at least one blogger that contributes at least 5% of the relevant posts. The large-document model (**LD**) which treats the blog as a single bag-of-words performs best at this task across almost all query sets. This result justifies our treatment of blog language model as a single bag-of-words in the two-stage smoothing models.

### Two-Level Smoothing

Now, looking at the performance of the two-level smoothing models on the blog post retrieval task, we evaluate the effect of sweeping the parameter values varying the influence of the blog evidence on ranking. Before fitting the parameters to a training set and testing on a hold-out set, we must evaluate the effect of these parameters on a training set alone. For each year of TREC queries, we evaluate the performance of the two smoothing models across a range of smoothing parameters, paying particular attention to the performance of the models as the amount of blogger-level evidence is increased.

Figures 4.4 and 4.5 show the performance of the two-level Jelinek-Mercer and Dirichlet models respectively. In these figures, we can see that, somewhat surprisingly, increasing the amount of blog-level evidence in fact *degrades* retrieval performance for both retrieval models. The best performance in both cases occurs when only standard (one-level) Dirichlet and Jelinek-Mercer smoothing is employed, i.e. when $\mu_C = \infty$ and $\lambda_B = 0$. These results represent best-case performance on a training set, and do not consider performance on a hold out set. This finding is contrary to our hypotheses that expanding the post language model with the blog language model would be beneficial.

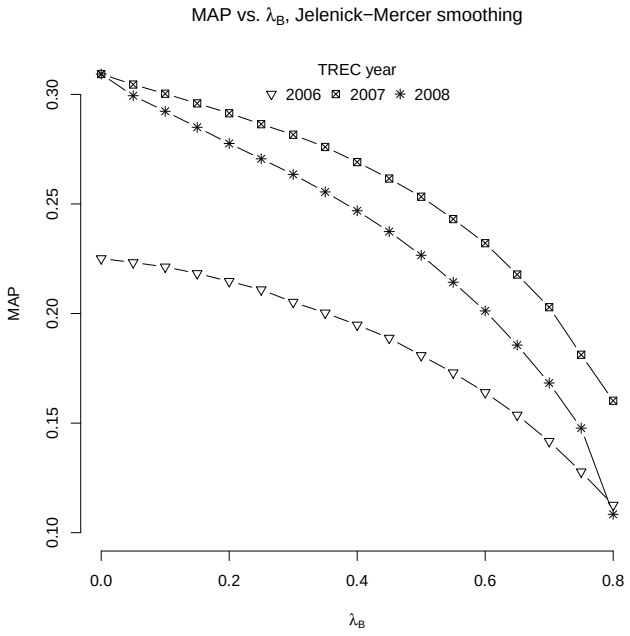



**Figure 4.4:** Training set performance of two-level Jelinek-Mercer smoothing. Figure shows the best performing parameter setting over a range of $\lambda_P$ values while varying the weight on the blog $\lambda_B$, for each year of TREC topics.

### 4.3.3 Analysis

As shown above, although the large-document model seems to be an effective retrieval model for finding "expert" bloggers, smoothing the post language model with the blog always hurts performance. The results presented in the last section show average performance, and performance on individual queries is similar – very few queries are helped at all by blog-level smoothing and those that are helped are only improved by a negligible amount.

These results invalidate our original hypothesis, and raise the question of why the two-stage smoothing models do not work for this task. In the following sections, we explore the effect of the smoothing models on the "expert bloggers" defined above, as well as diversity and the fraction of judged posts in the ranked list.

#### Smoothing's Effect on Expert Bloggers

The primary goal of these smoothing methods is to increase the retrieval scores of posts from bloggers who have written other relevant posts. By incorporating information from the entire blog language model, the hypothesis was that if a critical mass of relevant posts existed in the blog, all of those posts would benefit from blog-level smoothing. In particular, we would like to increase the post scores from those "expert bloggers" who have written more than 5% of the relevant posts.

In order to measure the effect of the smoothing models on the rank of posts from those expert bloggers, we look at the reciprocal rank of the first post written by any expert blogger. A higher expert Mean Reciprocal Rank (MRR) value indicates a model which is more effective at favoring expert bloggers. The expert bloggers in this analysis are the same ones identified above, in Figure 4.3 and Table 4.3. Figure 4.6 shows expert blogger reciprocal rank (Expert MRR) for the two-level Dirichlet smoothing model as $\mu_C$

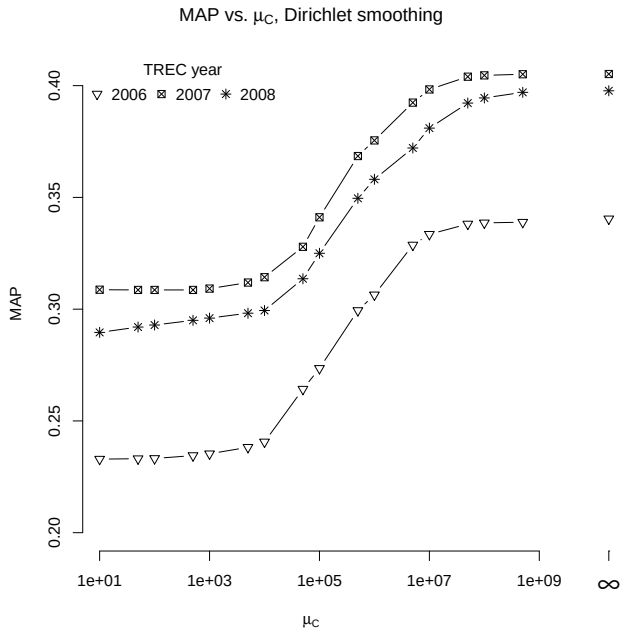



**Figure 4.5:** Training set performance of two-level Dirichlet smoothing. Figure shows the best Mean Average Precision (MAP) across a range of $\mu_D$ values as $\mu_C$ varies for each year of TREC topics. As $\mu_C \to \infty$ the amount of blog-level smoothing decreasing, resulting in collection-only smoothing at the rightmost point in the figure. As $\mu_C \to 0$ the amount of collection-level smoothing decreases, resulting in blog-only smoothing.

varies. This figure shows that for a range of $\mu_C$ values, the ranking of posts from expert bloggers are in fact improved. However, based on the results shown in Figure 4.5, this doesn't result in a corresponding increasing in overall MAP for the post retrieval task.

### Diversity of Represented Bloggers

One side-effect of incorporating blog-level evidence into the post language model is that it will decrease the diversity of the blogs in the ranking. As more information from the blog is incorporated into the post language model, all the posts' language models from the same blog tend to converge into a single blog language model. This will inevitably lead to an increased uniformity of post scores for those posts from the same blog, and a grouping together of those posts in the final ranking.

To measure the blogger diversity in the post ranking, we use entropy of the represented blogs at a cutoff $k$. This is given by:

$$H_k(P_1, P_2, P_3, \ldots, P_N) = -\sum_B P(B|P_1, \ldots, P_k) \log P(B|P_1, \ldots, P_k) \tag{4.2}$$

$$= -\sum_B \frac{\sum_{i=1}^k \mathbf{I}(b(P_i) = B)}{k} \log \frac{\sum_{i=1}^k \mathbf{I}(b(P_i) = B)}{k} \tag{4.3}$$

where the $P_i$ is the post at rank $i$, $\mathbf{I}(\bullet)$ is 1 when $\bullet$ holds true, and we let $p \log p = 0$ when $p = 0$.

Figure 4.7 shows the blogger entropy at $k = 10$ ($H_{10}$) for the Dirichlet two-level smoothing model with a fixed $\mu_D = 5000$ as $\mu_C$ varies. As expected, the diversity of the bloggers represented at top ranks

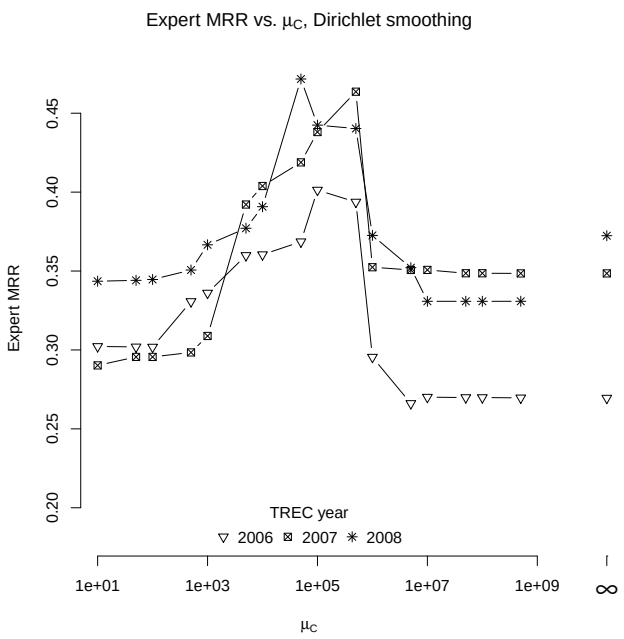



**Figure 4.6:** Best Mean Reciprocal Rank (MRR) of the first post by an expert blogger across a range of $\mu_D$ values as $\mu_C$ varies for each year of TREC topics. The left of the graph ($\mu_C \to 0$) the model performs blog-only smoothing and the right of the graph ($\mu_C \to \infty$) the model performs collection-only smoothing. We see that for a range of $\mu_C$ values, $10^4 \leq \mu_C \leq 10^6$, incorporating some blog-level evidence into the post language model improves the ranking of posts written by expert bloggers.

decreases as higher levels of blog-level smoothing are applied (towards the left-side of the figure).

Although the TREC task definition doesn't specify diversity as an evaluation criteria, the judged document pool is considerably more diverse than the documents retrieved by the baseline model used here. Macro-averaged across all 150 queries, the blogger entropy of the entire set of judged documents is 5.7075, whereas the entropy of the standard one-level Dirichlet retrieval model at 1000 documents (roughly the size of the judged document pool) is 4.7295. This considerable difference between the diversity of the retrieved and judged documents is likely due to the pooling process, where many different retrieval systems used different criteria for retrieval, possibly filtering blogs on language or spam-based features.

### Judged Documents

Because the diversity characteristics of our retrieved documents differ from the judged document pool, we may be concerned that there are a large number of unjudged documents in our retrieval runs. To investigate this, we look at the fraction of judged documents at top ranks. We define the fraction of judged documents at rank $k$ as

$$J_k(P_1, P_2, P_3, \ldots, P_N) = \frac{1}{k} \sum_{i=1}^{k} \mathbf{I}(P_i \in \mathcal{J}) \tag{4.4}$$

where the $\mathcal{J}$ is the set of judged documents.

Figure 4.8 shows the judged documents at rank $k = 10$ ($J_{10}$) for the Dirichlet two-level smoothing model with a fixed $\mu_D = 5000$ as $\mu_C$ varies. Again, we see a drop in the fraction of judged documents

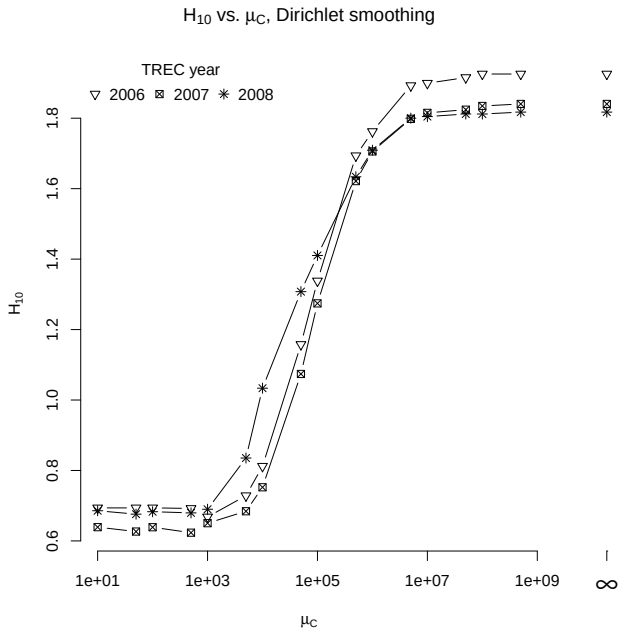



**Figure 4.7:** Blogger Entropy at cutoff $k = 10$ ($H_{10}$) for the two-level Dirichlet smoothing with a fixed $\mu_D = 5000$ as as $\mu_C$ varies. Higher $H_{10}$ values indicate a more diverse set of bloggers represented in the top $k$ retrieved posts. As more blog-level evidence is included into the post language model (decreasing $\mu_C$), a monotonic decline in Blogger Entropy at the top ranks is observed, resulting in a less diverse set of blogs represented by the top retrieved posts.

as the blog-level smoothing is increased (towards the left of the figure). In this figure we see that roughly 10% of the top-10 documents are unjudged at $\mu_C = 10^4$. These unjudged documents are treated as non-relevant by the standard TREC retrieval measures, in particular Mean Average Precision used as our primary evaluation measure here. This may be part of the reason we are seeing a drop in retrieval performance as blog-level smoothing is increased, although Figure 4.5 shows performance beginning to decline at a higher level of $\mu_C$ (roughly $\mu_C = 10^7$).

Several evaluation metrics have been proposed to be robust to the presence of unjudged documents in the ranked list. Binary Preference (BPref) [24], for example, penalizes a ranking based on the number of judged non-relevant documents ranked above a judged relevant document. Addition of unjudged documents in the ranking do not affect BPref, and BPref correlates highly with MAP when full judgements are available. Looking at an the performance with regard to BPref, we see a similar trend. Figure 4.9 shows BPref as a function of varying $\mu_D$. In this figure, we see a performance degradation almost identical to the MAP results in Figure 4.5. This indicates that the presence of more unjudged documents high in the ranked list is not the *only* reason for the two-level smoothing models hurting retrieval performance.

### Conclusions

The experiments presented above investigate the utility of incorporating blog-level smoothing into the the probabilistic language modeling framework. Our hypothesis was that by smoothing posts with the blog language model, we could improve the ranking of posts written by "expert bloggers", and therefore improve the overall post retrieval performance. Although we have some evidence that the ranking posts

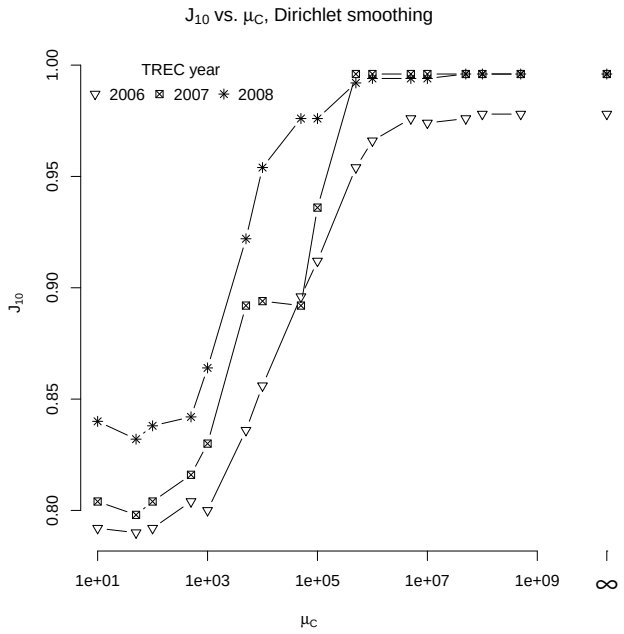



**Figure 4.8:** Fraction of judged posts at cutoff $k = 10$ ($J_{10}$) for the two-level Dirichlet smoothing with a fixed $\mu_D = 5000$ as as $\mu_C$ varies. As more blog-level evidence is included into the post language model (decreasing $\mu_C$), a nearly monotonic decline in the fraction of judged document is observed. The increasing fraction of unjudged documents likely contributes to the performance decrease observed in Figure 4.5.

written by expert bloggers can in fact improved with blog-level smoothing (shown in Figure 4.6), this does not result in an overall increase in post retrieval performance. In fact, as shown in Figures 4.4, 4.5 and 4.9, any amount of blog-level smoothing tends to *degrade* post retrieval performance.

We observe an increasing fraction of unjudged documents retrieved at top ranks when more blogger-level evidence is included in the model. An increased fraction of unjudged documents certainly has a detrimental effect on performance when using evaluation measures that treat unjudged documents as not relevant. However, the performance with respect to BPref, a measure designed to be robust to unjudged documents in the ranked list, shows a similar performance trend. For this reason, the presence of unjudged documents is likely not the *only* reason performance suffers.

The increase in blogger-level evidence is also coupled with a decrease in the diversity of blogs represented by the top retrieved posts. Although diversity of the result set is not a criteria specified in the task description and relevance assessment guidelines, a more diverse results set has the effect of mitigating some of the risk associated with favoring posts from very few bloggers. Not all bloggers with a large number of retrieved posts are experts, and the risk of favoring bloggers based on criteria like their query score appears to outweigh the reward in this case. We discuss this risk-reward tradeoff in more detail below, with a detailed analysis of which blogs are likely to be affected by models such as those presented here.

Before that analysis, we take a departure from the language modeling approach to incorporating blogger-level evidence. Although the probabilistic language modeling retrieval models are theoretically appealing to work with, the experiments above show some shortcomings of this approach. In order to remedy these problems, we will turn towards feature-based retrieval models. As we will show in the next section, feature-based models provide some tools, such as incorporation of negative evidence in the

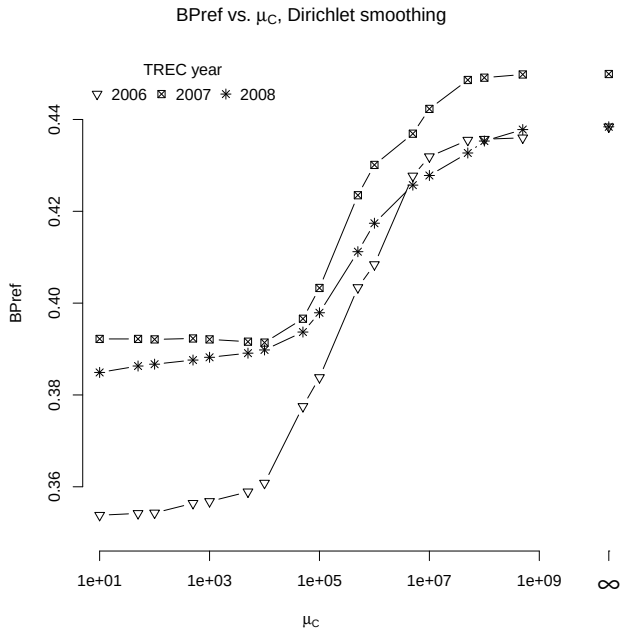



**Figure 4.9:** Training set performance of two-level Dirichlet smoothing. Figure shows the best Binary Preference (BPref) across a range of $\mu_D$ values as $\mu_C$ varies for each year of TREC topics.

ranking function, that are useful for this task.

## 4.4 Blog Structure in Feature-Based Retrieval Models

Feature-based retrieval models are a general class of ranking functions that incorporate a potentially large number of diverse features into a single scoring function. Many probabilistic language models can be viewed in this context, such as Metzler's Dependence Model [86] described in Chapter 2 which is a log-linear combination of unigram, n-gram and term window features. However, feature-based retrieval models have several potential advantages over probabilistic language models. First, it can be much easier to incorporate many types of features into a single model. For example, probabilistic language models often include a document prior $P(D)$ to favor documents with certain characteristics independent of the query. But, we are often concerned with many query-independent features of a document: graph-based quality measures such as PageRank or the number of in-links [23], the document length, spam classification confidence scores, and usage-based quality measures such as click-count. Second, feature-based models can easily incorporate negative evidence into the ranking function, which is particularly difficult in probabilistic retrieval models.

This section explores including blog-level features in feature-based retrieval models. We will focus exclusively on linear models here in order to aid interpretability of the models, but we expect the findings to generalize to non-linear models as well.

Unlike the previous section, where we focused on training-set performance as a means to evaluate the effectiveness of the approach, in this section we present three-fold cross-validation results. In these experiments, parameters are fit via a grid search on two years of TREC queries, and tested on the third, and this is repeated for all three years of data. Unless otherwise noted, results presented are from three-fold

cross validation.

### 4.4.1 Two-feature Models

The first feature-based models we may consider are simple score combination models, mixing the blog post retrieval scores with blog retrieval scores. In the probabilistic retrieval model setting, simple mixture models fall into this category:

$$P_{mix}(Q|P) = (1-\alpha)P(Q|P) + \alpha P(Q|b(P)) \tag{4.5}$$

$$= (1-\alpha)\prod_{q\in Q} P(q|P)^{n(q,Q)} + \alpha \prod_{q\in Q} P(q|b(P))^{n(q,Q)} \tag{4.6}$$

where $0 \leq \alpha \leq 1$ controls the amount of weight on the blog score. Note that this differs from the Jelinek-Mercer smoothing presented above in Section 4.3.1. In the smoothing models above, the blog-level evidence was included in the estimation of term generation probabilities $p(t|D)$, whereas here the blog-level evidence is incorporated outside of the probability estimation.

More generally, we will investigate two-feature models of the form:

$$\mathbf{f}_Q = \mathbf{y}_Q + \alpha \mathbf{B}\mathbf{b}_Q \tag{4.7}$$

where $\mathbf{b}_Q$ are blog scores for the given query using the **LD** blogger retrieval model that performed well in identifying expert bloggers above. As before, $\mathbf{y}_Q$ is a vector of post scores for the query $Q$, and $\mathbf{B}$ is the binary matrix of blog-post membership. Our goal is to learn an unrestricted weight $\alpha \in (-\infty, +\infty)$ that maximizes post retrieval performance. Throughout these experiments, we use a "shift and scale" normalization on our score vectors $\mathbf{y}$ and $\mathbf{b}$ so that the scores all lie in the range $[0, 1]$:

$$\mathbf{y}'_{Qi} = \frac{\mathbf{y}_{Qi} - \min \mathbf{y}_Q}{\max \mathbf{y}_Q - \min \mathbf{y}_Q} \tag{4.8}$$

#### Experiments and Analysis

Our first set of experiments uses the baseline retrieval scores from above, Section 4.2.2, for the post score vector $\mathbf{y}$, and the **LD** model scores, Table 4.3, for the blog score vector $\mathbf{b}$. Figure 4.10 shows the training-set performance on one training fold as we vary the $\alpha$ parameter. In this figure, we see a consistent decrease in MAP as $\alpha$ increases above zero. This finding agrees with the finding in the smoothing experiments above (Figure 4.5), where more blog-level evidence degrades retrieval performance. But, interestingly, we see an *increase* in MAP of about 6.5% as $\alpha$ becomes slightly negative.

Comparing the other measures we investigated above, we see similar trends as more blogger level evidence is incorporated into the ranking function. Figure 4.11 shows these other measures. At the left in the figure we can see the two-feature model is capable of improving the rank of posts written by expert bloggers, with Expert MRR almost monotonically increasing as $\alpha$ increases. But, as before, this is not result in an overall improvement in post retrieval performance in terms of MAP. In the middle of the figure, we see a monotonically decreasing blogger diversity as measured by $H_{10}$ across this range of $\alpha$ values. The increase in diversity as $\alpha \to -1.0$ is coupled with a slight increase and plateau in the fraction of judged documents at rank 10, $J_{10}$, shown at the right of the figure.

Across all three cross-validation folds, training with the same grid-search shown in Figure 4.10, we see a test set MAP of 0.4057, a 6.5% improvement over the baseline. Complete results are shown in Table 4.4. Figure 4.12 shows the change in Average Precision per-query for each of the test sets. In this figure, we can clearly see that the majority of queries (79.7%) are helped by this model.

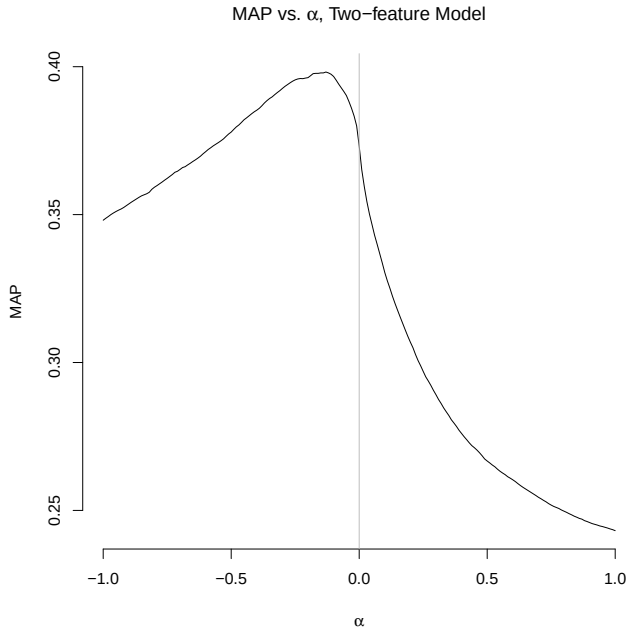



**Figure 4.10:** One fold training set performance of the two-feature model, Mean Average Precision (MAP) vs. $\alpha$. Vertical gray line indicates no blog-level evidence in the post ranking at $\alpha = 0$. To the right of the vertical line, a decrease in performance is observed as more blogger-level evidence is incorporated, similar to the smoothing model in Figure 4.5. To the left of the vertical line, an increase in performance is observed as a small amount of *negative* evidence is incorporated.

### 4.4.2 Expert-Sensitive Two-Feature Model

In the two-feature model presented above, we determined that by placing a small negative weight on blog score, we can realize a significant performance improvement in blog post retrieval. When considering the oracle model from Section 4.2.3, this negative weight likely captures the trend towards the negative weight on blogs as the number of retrieved posts increases (see Figure 4.3). However, for some queries, strong positive weights are learned by the oracle model on those "expert bloggers" who contribute a large fraction of judged relevant posts. A negative weight in the two-feature model above ignores this effect.

In this section, we explore the a variant of the two-feature model that is sensitive to expert bloggers. The bloggers who receive the highest weight by the oracle model (positive or negative) are those that have a large number of posts retrieved. For this reason, we consider a model that only weighs bloggers if they have written more than ten retrieved posts for a query. We can classify these bloggers as *GOOD* or *BAD* based on the relevance judgements of the written posts. If the majority of retrieved posts are relevant, the blogger is considered *GOOD*, otherwise the blogger is considered *BAD*. Given these classes, we develop a more flexible model for weighting blogs, allowing different weights for the different blogger models. This model is given as follows:

$$\mathbf{f}_Q = \mathbf{y}_Q + \alpha_G \mathbf{B}_G \mathbf{b}_Q + \alpha_B \mathbf{B}_B \mathbf{b}_Q \tag{4.9}$$

again, where $\mathbf{b}_Q$ are blog scores for the given query. In this case, rather than use the full blog-post affinity matrix $\mathbf{B}$, we use class-specific matrices $\mathbf{B}_G$ and $\mathbf{B}_B$ which only contain non-zero elements for those blogs identified as *GOOD* and *BAD* respectively. This model allows us to learn two weights $\alpha_G$ and $\alpha_B$

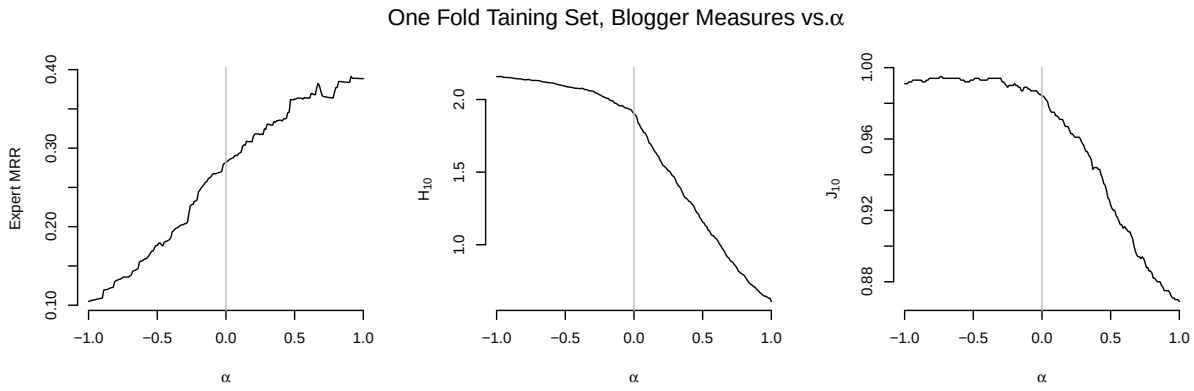



**Figure 4.11:** One fold training set performance of the two-feature model. Left to right: Expert Blogger Mean Reciprocal Rank (Expert MRR), Blogger Entropy at rank 10 ($H_{10}$), and Fraction Judged Documents at rank 10 ($J_{10}$) vs. $\alpha$. Vertical gray lines indicate no blog-level evidence in the post ranking at $\alpha = 0$. To the right of the vertical gray lines, $\alpha > 0$, we see a similar trend to the smoothing experiments (Figures 4.6, 4.7 and 4.8), with decreasing diversity ($H_{10}$) decreasing fraction of judged documents ($J_{10}$), but some improvement in the rank of posts written by expert bloggers (Expert MRR).

| TREC Year (test fold) | Two-Feature Model MAP | % Over Baseline |
|---|---|---|
| 2006 | 0.3539 | 3.9%* |
| 2007 | 0.4420 | 8.9%** |
| 2008 (50 new queries) | 0.4213 | 6.2%** |
| Average | 0.4057 | 6.5%** |

**Table 4.4:** Two-feature retrieval model performance. Performance marked with an asterisk (*) indicates a significant improvement over the baseline at the $p < 0.005$ level and performance marked with two asterisks (**) indicates significance at the $p < 0.001$ level with a one-sided paired t-test.

to differentially weight the different blog classes. Our goal is is to learn these weights that maximize post retrieval performance. Our expectation in these experiments is that we will learn $\alpha_G > 0$ and $\alpha_B < 0$, thereby addressing both the positive and negative weights learned through the oracle model experiments in Section 4.2.3.

### Gold-standard Blogger Identification

Given the above classification of bloggers, we need to understand the potential benefit to retrieval performance of this model. If we assume we know the gold-standard blogger labels (*GOOD* and *BAD*) so that we can construct the matrices $\mathbf{B}_G$ and $\mathbf{B}_B$ perfectly, we can train our model as before to learn the weights $\alpha_G$ and $\alpha_B$. This oracle model uses perfect knowledge of the relevance labels for the blogger labels, but cross-validation to train the $\alpha$ weights. Figure 4.13 shows the training-set performance of the expert-sensitive model as we vary $\alpha_G$ and $\alpha_B$. One the left (right) we show the *maximum* performance achieved for a given value of $\alpha_B$ ($\alpha_G$) over the range of $\alpha_G$ ($\alpha_B$) values. In this figure, as expected, we see an increase in performance as $\alpha_B$ decreases, plateauing around $\alpha_B = -0.4$. We do see a slight increase in performance as $\alpha_G$ increases, reaching a maximum at $\alpha_G = 0.1$.

Table 4.5 shows the full three-fold cross validation performance with learned $\alpha$ weights and gold-

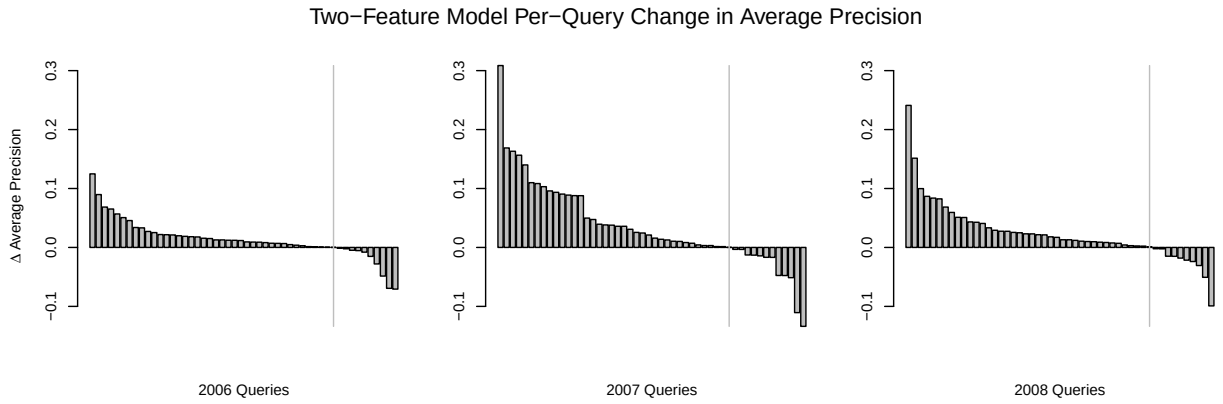



**Figure 4.12:** Change in Average Precision per-query between the baseline model and the two-feature model on each fold's test set. Vertical line divides queries with a positive and negative change in Average Precision. The two-feature model helps in 118 out of 150 queries (79.7%) across all three test folds.

standard (oracle) blogger labels. In this table, we see a sizable additional performance boost over the

| TREC Year (test fold) | Expert-Sensitive Model MAP | % Over Baseline |
|---|---|---|
| 2006 | 0.3788 | 11.2% |
| 2007 | 0.4632 | 14.1% |
| 2008 (50 new queries) | 0.4476 | 12.8% |
| Average | 0.4299 | 12.8% |

**Table 4.5:** Expert-sensitive retrieval model performance. Blogger labels are from an oracle blogger classifier, and model weights ($\alpha_G$ and $\alpha_B$) are learned through cross-validation.

baseline, approximately twice the performance gain realized with the two-feature model (Table 4.4). This result is an indication that if we can identify those *GOOD* and *BAD* bloggers, rather than just uniformly weighting the bloggers, we can realize further retrieval performance improvements.

### Automatic Blogger Identification

In this section, we evaluate the efficacy of automatically classifying bloggers as *GOOD* or *BAD* for a given query. We expect this task to be easier than classifying documents for relevance for two reasons: First, we have more data associated with the blogger than a single document. These bloggers have written at least ten blog posts that are retrieved for this query. Second, our tolerance for errors may be high. We do not know what accuracy of blogger classification is required to realize a performance improvement in blog post ranking. Its possible that a weak blogger classifier may be sufficient for this task.

We take a straightforward approach to blogger classification, generating features based on various aspects of posts belonging to that blog, the baseline post retrieval and the blog retrieval score. Table 4.6 gives a summary of the features used for this classification task. These features are designed to describe several aspects of the quality of the blog posts and the blog in general. The comment count features should capture the general popularity of the blog posts, the post count and length features should capture the prolificacy of the blogger, retrieval scores capture the degree of match between the query and blog posts, and link features generally capture the diversity and quantity of linking in the blog.

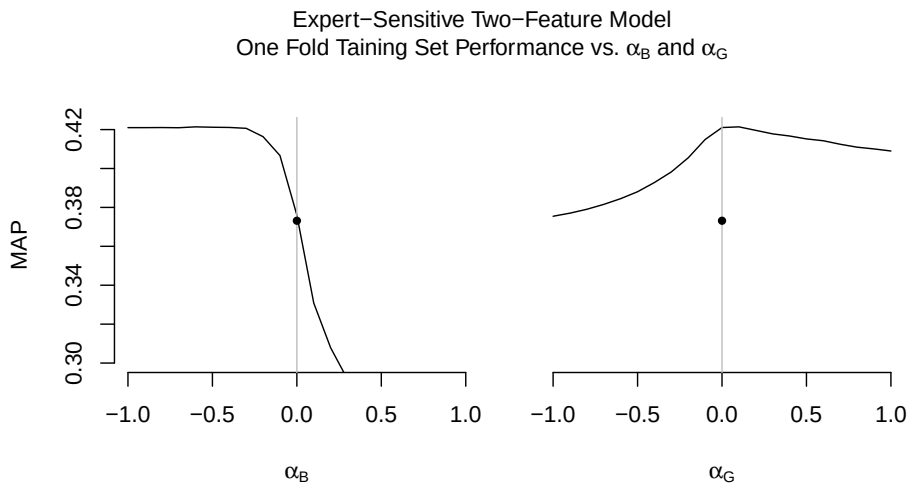



**Figure 4.13:** One fold training set performance of the expert-sensitive two-feature model. On the left, best performance (MAP) for range of $\alpha_B$ values. On the right, best performance (MAP) for range of $\alpha_G$ values. Vertical gray lines indicate no blog-level evidence in the post ranking at $\alpha = 0$. Solid circles indicate performance when $\alpha_B = \alpha_G = 0$. The performance effect of adjusting $\alpha_G$ and $\alpha_B$ largely correspond to our intuition — dis-favoring prolific bloggers who do not write relevant posts, and slightly favoring prolific bloggers who write relevant posts both lead to an increase in performance.

We use an $L2$-regularized logistic regression classifier, fitting the regularization parameter on 5-fold cross-validation within each training fold [49]. Using the output of this trained model, we can fit the same $\alpha_B$ and $\alpha_G$ parameters used in the model above via grid search.

The classifier itself achieves approximate 80% accuracy in classifying *GOOD* vs. *BAD* bloggers, a slight improvement over the majority class classifier which achieves 77% accuracy. But this level of accuracy is inadequate to result an improvement significantly better than the two-feature model above. Results for the model with learned *GOOD*/*BAD* blogger classes are given in Table 4.7. These results show an improvement over the baseline comparable to the simpler two-feature model.

### Discussion & Analysis

In this section we explore the differential weighting of bloggers based on the fraction of relevant posts they have written. Given perfect knowledge of whether these bloggers have written more relevant than non-relevant posts, we can realize a sizable improvement over the baseline post retrieval performance of more than 12%. The efficacy of this approach, however, hinges on the ability to automatically identify those bloggers that have written a significant number of relevant posts. Our classification approach achieves approximately 80% accuracy, a slight improvement over predicting the majority class. Although this accuracy is insufficient to yield any improvement over the simpler two-feature model, the analysis here provides an interesting opportunity for future work. Given further refinements of the blogger classification model here, a more sizable performance improvement could be achieved.

Although we developed models in this section that show incorporating blog-level evidence in post ranking can significantly improve post retrieval performance, the learned parameter weights are somewhat contrary to our initial expectation and motivations. The dominant trend in these results is to dis-favor bloggers who have a high query score, a large number of retrieved posts, but few relevant posts. Figure 4.13 shows that the performance gain in dis-favoring these *BAD* bloggers is much greater than the gain from favoring the *GOOD* bloggers. In an effort to explain why these models perform the way they do, the

| Feature Class | Feature Name |
|---|---|
| Comment Count | Over all posts (comment.count.all) |
| | Over retrieved posts (comment.count.retr) |
| | Average over all posts (comment.per.post.all) |
| | Average over retrieved posts (comment.per.post.retr) |
| | On the top retrieved post (top.post.comments) |
| Post Count | All posts (post.count.all) |
| | Retrieved Posts (post.count.retr) |
| Post Length | Summed over all posts (post.length.all) |
| | Summed over retrieved posts (post.length.retr) |
| | Averaged over all posts (post.length.per.post.all) |
| | Averaged over retrieved posts (post.length.per.post.retr) |
| | Of the top retrieved post (top.post.length) |
| Retrieval Score | Blog **LD** model Score (score) |
| | Baseline model score of top retrieved post (top.post.score) |
| Link Count | All links from blog (num.links) |
| | Links to different hosts (num.ext.links) |
| Link Features | Entropy of anchor text (link.text.entropy) |
| | Entropy of link URLs (url.entropy) |
| | Entropy of link URL hosts (url.host.entropy) |
| | Entropy of external link anchor text (ext.link.text.entropy) |
| | Entropy of external links (ext.url.entropy) |
| | Entropy of external link URL hosts (ext.url.host.entropy) |
| Other | Reciprocal rank of top post (top.post.rr) |

**Table 4.6:** Features used for *GOOD*/*BAD* blogger classification.

following section explores aspects of those high-scoring blogs that may be detrimental to the post retrieval performance in the following section.

## 4.5 Analysis of Blog-Level Evidence

The previous sections present two methods for incorporating blog-level features into post ranking algorithms. In both models, when incorporating more positive evidence from the language of the blog as a whole, either through document expansion or through score combination, performance of the post ranking degrades. But, with the ability of the feature-based model to incorporate negative evidence in the post scoring, we see a significant improvement when dis-favoring high-scoring blogs.

This finding is largely contradictory to our original motivations for this work, and in this section we aim to explain why this is the case. To fully understand why the feature based models are capable of improving performance when dis-favoring high-scoring blogs, we look at the queries that are helped and hurt most by these models and the blogs that are most affected by this weighting. Through this analysis identify several common reasons why blogs receive a high score, and why these blogs may not be useful blogs to favor in post ranking.

Our process for analyzing the effect of this blog-level evidence is as follows:

| TREC Year (test fold) | Two-Feature Model MAP | % Over Baseline |
|---|---|---|
| 2006 | 0.3542 | 4.0% |
| 2007 | 0.4427 | 9.1% |
| 2008 (50 new queries) | 0.4157 | 4.8% |
| Average | 0.4042 | 6.1% |

**Table 4.7:** Expert-sensitive retrieval model performance, using learned blogger labels and weights.

1. We take a sample of queries, including those examples above [1] and the queries helped and hurt most by the best-performing two feature model [2].
2. We identify the top-scoring 3-4 blogs for each of these queries with respect to the **LD** model used in the two-feature models. The posts belonging to these blogs are most affected by the blog weighting, and their change in rank likely contribute most to the performance change when a non-zero blog weight is used.
3. We inspect those blogs and identify the dominant reasons they received a high query score.

In this process, many cases were identified where the top scoring blogs are in fact excellent resources, producing large numbers of relevant posts. The two queries that are hurt the most by the best performing feature based model both have several high-scoring blogs that contain many relevant posts for the query. The top three scoring blogs for *[Blackberry]* (862) together have over 200 retrieved posts in the baseline post ranking, and over 80% of these blogs' judged posts are judged relevant. The entertainment and celebrity gossip site Egostatic[3] ranks highly for the query *[Natalie Portman]* (880) and contains several relevant posts.

Favoring these predominantly relevant blogs would be beneficial for these queries. But, based on the learned parameters for the feature-based model above, on average the reward for favoring the top blogs for these queries does not outweigh the risks associated with favoring top scoring blogs for other queries. Although these queries are hurt considerably (-16% and -30% change in MAP respectively), this is the minority case in our dataset.

Several recurring reasons for a predominantly non-relevant blog receiving a high query score are also identified. These reasons highlight the risks of favoring highly scoring blogs in post ranking, and generally point to a mismatch between the intent of the query and the content of the blog.

### 4.5.1 Query Intent Mismatch at the Blog-Level

The negative weight on the blog-level retrieval score learned by the two-feature model is an indication that query matches at the blog level may not be a good indicator of relevance. In these query mismatches, although the query terms occur in the blog text, the intent of the query is not served by that document. Four primary types of query mismatch were identified in our analysis.

#### Word Sense Mismatch

We observe some cases of blogs being scored very highly by the retrieval system, but the dominant word sense on the blog is different than the intended word sense in the query. The top scoring blog for the query

[1] *[March of the Penguins] (851), [Colbert Report]* (864), *[Cheney Hunting]* (867), *[Cindy Sheehan]* (871)
[2] *[Blackberry]* (862), *[Natalie Portman]* (880), *[Heineken]* (883), *[Jim Moran]* (892)
[3] http://www.egotastic.com

*[Colbert Report]* (864) falls into this category. The site "BrentColbert.com"[4] contains many occurrences of the relatively rare query term "Colbert". Although one post from this blog is judged relevant, the dominant meaning of this term on this blog is different than the sense in the query. In this case, penalizing the 34 blog posts retrieved form this blog is clearly a beneficial strategy. Although these posts are dense with occurrences of a reasonably discriminative query term, they are rarely relevant.

The query *[March of the Penguins]* (851) has two high scoring blogs with word sense mismatches to the query. Both of the blogs "Twelve Happy Penguins"[5] and "About.com Animals"[6] mention the term "penguin" with some frequency, but never in reference to the movie referred to by the query. In this case, 22 posts were retrieved from these blogs combined and none are judged relevant.

This issue of word sense mismatch is a clear example of the risks involved when favoring some bloggers over others. As we showed experimentally above, favoring blogs in the post ranking tends to reduce the diversity of blogs in the ranked list. In these cases, a single word sense is dominant throughout the blog. If there is a word sense mismatch with the query, and these blogs are favored, the errors incurred with this mismatch will compound. If, on the other hand, diversity of the blogs represented by the post ranking is increased, a system can hedge its bets against a one blog's possibly differing word sense dominating the post ranking.

### Blog Post Listing Mismatch

A common navigational feature on blogs is a listing of recent posts in the blog's sidebar. Figure 4.14 shows an example of a blog post from the technology blog "Gizmodo"[7]. In this figure, we see the most recent few posts, and further down on the page approximately 100 titles of recent posts are listed. Although published on the same blog, these posts are not necessarily on the same topic as the post displayed. When the post listing is displayed on every post from a blog, the query term matches will compound and the blog will receive a high score.

This injection of non-relevant content into every post belonging to a blog happens with some frequency in our experiments. In several cases in the top retrieved blogs, a small number of relevant posts were written by the blogger, and those posts' titles contained query term matches. But, the repetition of the title across all subsequent posts in the collection leads to spurious query term matches. For the query *[Cheney Hunting]* (867), the blog "About.com Political Humor"[8] published three relevant posts. But, the titles of these relevant posts also appeared in in fourteen other non-relevant posts. Similarly, the blog "Pennsylvania: Hunt Fish Shoot"[9] published only one relevant post, but this post's title text was repeated across nine other retrieved posts. The presence of query term matches in many of the posts published by this blog led to the blog receiving a high score for this query, even though only a small fraction of those posts were relevant.

### Boilerplate Mismatch

In addition to the blog post listing, bloggers often include a "blogroll" with links to other blogs or websites of interest to the blogger and likely to the readers. This content is most frequently included in the blog's post template or *boilerplate*. Similar to the post listing, this text is repeated on each post document from

[4] http://brentcolbert.com
[5] http://chai.blogs.com/twelve_happy_penguins
[6] http://animals.about.com
[7] http://gizmodo.com
[8] http://politicalhumor.about.com
[9] http://pahuntfishshoot.com/

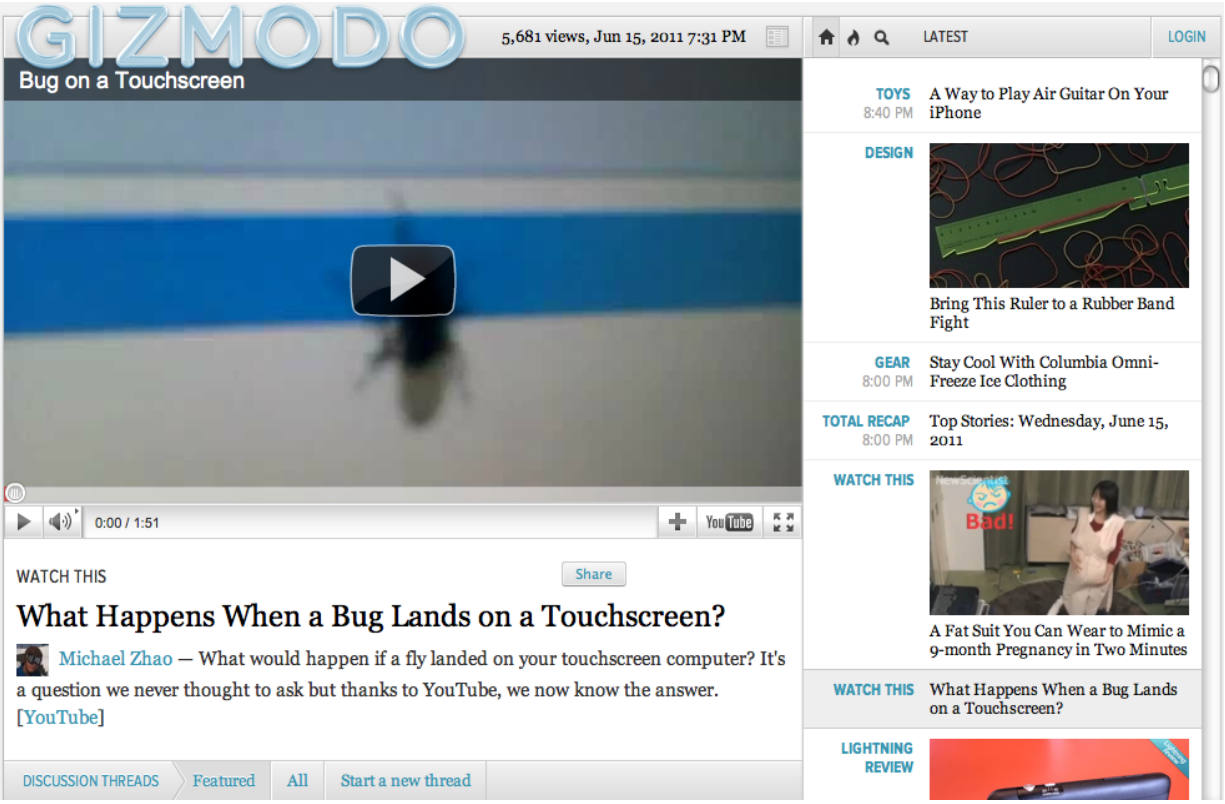



**Figure 4.14:** Blog post from the technology blog Gizmodo (`http://gizmodo.com`). Listing of titles of recent posts can be seen along the right side and scrolling reveals many more titles.

that blog in the collection, however the blogger may not ever write a blog post relevant to this text. If there are query term matches with this boilerplate content, they will match every post belonging to that blog.

The query *[Jim Moran]* (892) is an example of a query susceptible to boilerplate matches. The person referred to in the query was running for political office at the time of the blog dataset collection. Bloggers wishing to proclaim their political affiliation linked to his website, and the link text matching the query was repeated on each post. For this query, the top three scoring blogs, all politically oriented, have over 100 posts retrieved. But, none of these posts are are judged relevant to the query.

Both the boilerplate mismatches and the post listing mismatches are associated broadly with navigational or "sidebar" content on the blog. But, we consider these different types of mismatch for two reasons. First, in contrast to the post listing mismatch where the blogger often does write a relevant post, the boilerplate mismatches do not necessarily point to possibly relevant content elsewhere on the blog. Second, the post listing changes each time an new blog post is published and listed, whereas the boilerplate tends to be static across all published blog posts. Because of these reasons, the appropriate techniques for eliminating these type of mismatch may be different.

### Spam

Blogs provide a web publishing mechanism with a low barrier to entry. Because of this, blogs are particularly susceptible to spam, generally in the form of link farms. In these spam blogs, a large number of links to a target website are published, all containing text likely to match search queries. The goal of these sites

is to bias a search engine's internal representation of the linked-to web pages, causing them to rank higher for those query terms or inflate their PageRank or other link-based quality measure [64, 23].

In one of the queries evaluated, *[Natalie Portman]* (880), spam blogs were scored highly by the blog scoring algorithm. This blog clearly was designed to drive traffic to a pornography website, and down-weighting or filtering blogs like this is likely improve performance. Several techniques have been proposed to automatically identify spam blogs, and application of those techniques to this task is likely to improve performance [91, 64].

### 4.5.2 Discussion

The discussion above presents several reasons that predominantly non-relevant blogs receive a high retrieval score. In all of these cases, the presence of query term matches across most or all posts from the blog compound the mismatch problem, inflating the query score of the blog as a whole despite the blog having none or very few relevant blog posts. The balance between favoring blogs that truly contain a large number of relevant posts and blogs that receive a high score because of these spurious query term matches presents a risk-reward tradeoff when incorporating blog-level text features in post ranking. In the retrieval experiments above, the reward is outweighed by the risk, and thus the two-feature retrieval model learns a negative weight on blog-level retrieval score. In other tasks or datasets, this may not necessarily be the case.

The method we present to deal with this problem, discounting posts by the blog-level retrieval score, is extremely lightweight and surprisingly effective. In the great majority of queries, almost 80%, performance is improved. We believe the reason for this performance increase is twofold. First, based on our analysis in this section, blogs with spurious query term matches tend to be scored highly because of the compounding effect of the repetition across posts. Second, this method has a tendency to diversify the blogs represented in the post ranking, thereby hedging against these mismatches.

While the technique we applied to mitigate the mismatch problem is effective, other approaches that are designed to address each mismatch type individually may be more effective and provide more control over the blog post ranking. When comparing the mismatch types, the boilerplate and post listing mismatches were the most common in our analysis. In the six queries we analyzed that were helped by this negative weighting technique, all had at least one blog that received a high score based primarily on boilerplate and post listing mismatches. Spam and word sense mismatches were the reasons for high scores on blogs retrieved by two queries. For these reasons, investigation into content analysis algorithms is likely to be the most beneficial to this task.

One relatively straightforward algorithm to separate boilerplate content, DiffPost, has been applied to the task of blog post retrieval using the same collection studied here [94]. This algorithm looks at differences in the textual content of two temporally adjacent blog posts from the same blog, and retains only the lines of text in a post that are not present in the previous post from that blog. The reported performance of this method varies by year, but is roughly on par or slightly below the performance of our two-feature model discussed in Section 4.4.1. It is unclear whether the method presented here is slightly superior due to the stronger baseline retrieval model or to other factors.

Although content analysis and spam classification algorithms could alleviate most of the mismatch problems observed above, they will not address the word sense mismatches. In this case, the dominant word sense used on a blog is different than the word sense intended by the query. In the case of this type of mismatch, is likely that deeper natural language processing is needed to completely alleviate the problem. In the absence of that linguistic processing, favoring a diverse set of blogs in the top retrieved posts is a sensible strategy to hedge against a single, possibly incorrect word sense dominating the results.

## 4.6 Related Work

The work presented in this chapter looks at the task of blog post search, a well-studied IR task in recent years through TREC evaluations [100, 82, 102]. We are taking a somewhat different focus than the main aim of the blog track, which was to retrieve relevant and *opinionated* blog posts and classify polarity. In this section, we present work on blog post search that, like our work, integrates features from the blog-level into post search. A discussion of the work on opinion detection is out of the scope of this thesis.

The techniques employed above are similar to techniques applied to cluster-based retrieval and score regularization. We also present and contrast some of those methods closest to our approach.

### 4.6.1 Using Blog-Level Features in Post Search

Weerkamp and de Rijke's work is the closest to the work here, evaluating a variety of "credibility indicators" calculated over the blog posts and the blog as a whole. The authors find that some of these measures are effective at improving blog post retrieval [127]. The set of credibility indicators derived from blog level evidence include spam classification, the number of comments across all posts on the blog, the regularity of the time interval between posts, and the overall topical consistency of the posts in the blog. Of the blog-level credibility measures studied, only the spam filtering provided consistent improvements. Unlike the blog query score used in the work above, none of these blog-level measures are query-specific. Rather, they are all computed independently of the query, possibly as a preprocessing step before retrieval.

Several other studies have also assessed the effect of spam on blog post search, and methods to automatically identify those spam blogs. Spam is treated as a feature of the blog, not the post, and the automatic identification of spam is to some degree using blog-level features to inform the post ranking. Macdonald and Ounis looked specifically at how opinion retrieval performance is affect by the presence of spam blogs, and find that in TREC evaluations approximately 10% of posts retrieved by participating systems are from spam blogs [83]. Their analysis shows that overall system performance could be improved by the identification and removal of spam blogs. This result is supported by the Weerkamp work described above [127], as well as by other studies [91]. Spam blog identification has also been evaluated independently of retrieval tasks, with a variety of methods proposed based on textual and temporal features of a blog's posts [69, 63].

Content analysis methods that attempt to separate useful document text from extraneous boilerplate or template text also typically use blog- or collection-level features. As mentioned above, Nam et al. developed the DiffPost content analysis algorithm for the task of eliminating boilerplate content in a blog post index and apply this technique blog post search [94]. This algorithm uses textual differences across subsequent blog posts from the same blog to identify new post content and eliminate the repeated content. The authors report significant performance improvements over their baseline model on the 2006 and 2007 queries, as well as a considerable reduction of the index size. Other more sophisticated content analysis algorithms to detect text-reuse have been applied to blog collections, finding large amounts of duplicate text across blog posts [111].

These methods, while improving performance by using blog-level features, do not use the text of the blog to the degree that we attempt in our experiments. None of these techniques use query-specific features from the blog in post ranking.

### 4.6.2 Cluster-Based Retrieval and Inter-Document Similarities

Techniques used in cluster-based retrieval methods are similar to some of the methods explored in this chapter. Chapter 2 gives an overview of cluster-based retrieval approaches, and we describe how several

of these approaches relate to our work here. These approaches to document retrieval broadly attempt to leverage inter-document similarities to improve retrieval performance. To the extent that blogs define clusters of documents, or blog membership defines post similarity, we can think of the work in this chapter in the same light.

Liu and Croft evaluate several approaches to utilizing document clusters in the language modeling retrieval framework [71]. In their work, document clusters are generated either statically prior to retrieval time, or using only those top retrieved documents. The authors find that smoothing the documents' language model with the language model of the corresponding cluster is a generally effective approach. Their methods are identical to our approach in Section 4.3.1, representing the cluster as one large document and applying the two-stage Jelinek-Mercer smoothing model. In contrast to our results, which showed a consistent degradation in retrieval performance with increasing smoothing from the blog-level, they found that smoothing the document language models with their corresponding cluster consistently and significantly improves performance across a range of collections.

Kurlund and Lee present a series of more complicated cluster-based retrieval models also in the language modeling framework [66]. Their model, described in Section 2.5, linearly combines the document query likelihood score $P(Q|D)$ with scores from related clusters, generally calculated via a large-document representation concatenating the cluster's blog posts. If we consider blogs analogous to clusters, this model shares some similarity with the two-feature model discussed in Section 4.4.1. In our case, the related clusters includes only the blog corresponding to the post document $d$, and the associations $P(d|c)$ is constant. Their model has a strict probabilistic interpretation, and does not allow for negative weights on the cluster score.

Diaz's score regularization model, presented in Section 2.5 can also be viewed as a similar approach to our own [42]. This regularization approach is quite similar to our two-feature retrieval model when restricting the weight parameter $\alpha$ in Equation 4.7 to be positive. If we define the diffusion operator in the Diaz model $\mathbf{S}$ as representing blog membership, letting the elements $s_{ij}$ be defined as

$$s_{ij} = \begin{cases} \frac{1}{|b(D_i)|} & \text{if } b(D_i) = b(D_j) \\ 0 & \text{otherwise} \end{cases}$$

then these two models are roughly equivalent, using a small-document representation for the blog score here, and a large document representation in our work above.

Although taking a mathematically similar form, our two-feature model cannot be considered a form of score regularization. In the two-feature model, all the post scores from the same blog are adjusted by the same amount which does not affect the local score consistency. The Diaz model, in contrast, uses a true similarity metric to construct $\mathbf{S}$ rather than just indicating blog or cluster membership. Because of this, it is unlikely that document scores would be adjusted by the same amount in any document neighborhood. Additionally, the two feature model performs best with a small *negative* value for $\alpha$. This is not supported by the Diaz model.

## 4.7 Conclusion and Future Work

In this chapter, we explore the task of blog post search, and specifically evaluate the efficacy of integrating text content from the blog-level into the post scoring. To motivate this work, we analyze the relevant document distribution across bloggers, and observe that a small number of bloggers often contribute a disproportionately large number of relevant posts. Our initial hypothesis is that by incorporating language from the blog as a whole into the post scoring functions, the posts written by these "expert bloggers"

will be favored in the document ranking. To test this hypothesis, we apply several retrieval models to the task of blog post search: first we explore document smoothing or expansion models, and next we explore feature-based score combination models. While we find that smoothing with the blog-level language model is generally harmful to post retrieval performance, inclusion of the blog-level query score as *negative* evidence can significantly improve performance by 6.5%. We show that further performance improvements, up to 12.8%, could be achieved by identifying expert bloggers and differentially weighting their posts, but the automatic method presented here fails to achieve a sufficient level of accuracy.

The finding that incorporating positive evidence from the blogger language model hurts performance is contrary to our initial motivation for exploring this task. Upon performing an in-depth analysis of the blogs that are most affected by these retrieval models, we identify several types of blog-level query mismatch that lead to largely non-relevant blogs receiving a high query score: spam, boilerplate content matches, and word sense mismatches. When combining the content of all posts in a blog, these mismatches compound, and as a result these blogs receive a high retrieval score. Because of the high score on these non-relevant blogs, our feature-based model learns a negative weight on the blog-level query score.

Down-weighting blogs based on their query score seems to mitigate the effect of many of these undesirable properties of high-scoring blogs. This lightweight, although naïve, approach improves performance on almost 80% of the queries in our test set. Although many queries are helped, this down-weighting of the blog-level score is at the expense of penalizing posts from the handful of expert bloggers that do produce a large number of relevant posts. This highlights the risk-reward tradeoff in favoring some blogs over others when ranking blog posts. In the case of this dataset and this task, the reward for favoring high-scoring relevant blogs is offset by the risk of mistakenly favoring blogs with spurious query term matches.

These insights open several doors to future work in blog feed search.

1. **Spam and Content Analysis**: Our analysis in Section 4.5 finds that the blog post retrieval is negatively affected by the presence of spam blogs and boilerplate content when author-level content is included. This result indicates that by eliminating spam blogs and boilerplate content the performance characteristics of the smoothing models would more closely match our original expectations. As near-term future work, we propose investigating the efficacy of the presented models after these types of collection preprocessing methods are applied.
2. **Expert Blogger Classification**: In Section 4.4.2, we present a retrieval model that differentially weights posts from expert bloggers and other prolific bloggers. We find that, if these experts can be identified a substantial ($> 10\%$) performance over the baseline can be achieved. Although the automatic classification methods presented above does not perform accurately enough to achieve this performance improvement, the analysis here presents an opportunity for substantial performance gains.
3. **Risk and Reward Models**: The idea of risk and reward with respect to information retrieval tasks has been formalized and evaluated by Collins-Thompson [35]. While their work focused on selecting terms for query expansion and re-weighting, an analogous model could be developed for the task discussed here. As we have described, incorporation of blog-level evidence in the blog post search task naturally lends itself to analysis as a risk-reward tradeoff.

# Chapter 5

# Blog Feed Search[1]

## 5.1 Introduction

Blog feed search is an information seeking task in which someone has an ongoing interest in a topic and plans to follow blogs discussing that topic on a regular basis, possibly through their feed reader. Several commercial blog search engines exist (`blogsearch.google.com`, `technorati.com/search`). Most of these present a feed search service in conjunction with blog post searching and some are closely integrated with feed reading services.[2]

Several characteristics of this task distinguish blog retrieval from typical ad hoc document retrieval. First, blog retrieval is a task of ranking document collections, or aggregates, rather than single documents. In this respect, blog feed search bears some similarity to resource ranking in federated search — the collection can similarly be viewed at multiple levels of granularity, and questions of which level is the most effective representation apply. Second, as opposed to other tasks ranking document aggregates, the goal of this task is to find feeds with a "principal, recurring interest" in the topic of the query [82]. This guideline implies that the individual blog posts may bear some topical relationship to each other, or to the blog as a whole. Other tasks of ranking aggregates, like resource selection in federated search or cluster-based retrieval, are often a means to find a document collection that can then be subsequently searched. When the ultimate goal is to retrieval the aggregate document, is not clear that methods developed for those tasks are applicable.

Previous work has generally shown that in the language modeling framework, a large-document model is superior for this task. Although highly performant, there are several practical drawbacks to large document models. First, blog posts are published periodically, not all at once. If the blog feed is indexed as a single document, discovery of new posts would require updating the feed in the index, something not supported by inverted indexes, or re-indexing the entire feed including the new content. For this reason, it is advantageous to index a blog as individual posts, adding new posts as they are discovered. Second, an index containing posts rather than entire blogs could be used for both blog feed search as well as post search, two common tasks with blog collections. Finally, indexing individual posts could lead to more flexibility in the retrieval model. As we will show, the ability to differentially weight posts when ranking feeds can lead to significant improvements.

For all of these reasons, we are interested in developing a small-document model for blog feed search

[1] This chapter presents work originally published by Elsas et al. [45].

[2] In this work, we refer to blogs (the collection of HTML web pages) and feeds (the XML syndication format version of the blog) interchangeably as there is a one-to-one correspondence between the two. Likewise, we refer to a blog post or permalink document (the HTML page) and a feed entry (an XML element within a feed) interchangeably.

that performs at least as well, and preferably better than the baseline large document. In this chapter, we present a series of probabilistic retrieval models for blog retrieval that do exactly that. Through these models, we investigate the relationship between the topicality of individual entries and the blog as a whole, and we investigate the appropriate unit of representation for this task — whether it is the entry or the feed. These models extend a state-of-the-art approach previously developed for federated search to blog retrieval, the ReDDE algorithm proposed by Si and Callan [114]. Contrary to previous work in feed retrieval [110, 7, 44], we show that a federated search model with entries as the unit of retrieval can outperform a large document model that treats the whole feed as the unit of retrieval.

### Chapter Roadmap

Our exploration of the blog post search task in this chapter is organized as follows:

- In **Section 5.2** we present several probabilistic language modeling retrieval models for blog feed search. These models are inspired by the expert search and resource ranking models described in Chapter 2, but adapted to this task by modeling the topical *centrality* of a post to a blog.
- **Section 5.3** presents experiments using those proposed models on the blog feed search task. We show that, while a large-document model performs well for this task, a small-document model that is capable of differentially weighting and sensitive to the size of the blog feed can achieve significant improvements.
- In **Sections 5.4** and **5.5** we presented related work and conclude the chapter.

### Research Contributions

The major contributions of the work presented in this chapter are:

- A comparison of two general methods for representing blog feeds in search, the "large document" approach that treats a blog feed as a single monolithic document, and a "small document" approach that scores blog posts individually and aggregates those scores into a feed score.
- The development of a novel method for measuring the "centrality" of a blog post to the feed, for use in re-weighting blog posts in the feed scoring.
- The demonstration that two factors are critical for the success of a small document model: favoring those posts that are central to the topic of the feed, and normalizing for feed length. Utilizing both of these in the small document model results in significant improvements over the baseline large document model.

## 5.2 Probabilistic Retrieval Models for Feed Search

Previous research into feed search models has drawn analogies between this task and several other well-studied retrieval tasks: expert finding [15], cluster-based retrieval [109] and resource selection in distributed information retrieval [45, 110]. All of these tasks share the common goal of ranking document aggregates rather than single documents. Refer to Chapter 2 for a discussion of these other areas of research. In the following sections we present a series of probabilistic retrieval models for feed search based on ones previously proposed for ad hoc retrieval and resource ranking in federated search.

### 5.2.1 Large Document Model

The first and simplest model treats each feed as a single monolithic document, ignoring any distinction between individual entries within those feeds. This baseline model is a simplified version of the unexpanded large document model presented by Arguello et al. [7, 44], the best performing retrieval model without query expansion in the TREC 2007 Feed Distillation task. Keeping the same naming convention, we refer to this model as the *large document* model.

In comparison to previous work on resource ranking in federated search, this model is similar in spirit to the CORI algorithm, which creates pseudo-documents for each collection using collection term frequency statistics [26, 27], as well as Liu's "concatenation" cluster representation model [73]. Pseudo-documents are then ranked by their similarity to the query. Our large document model uses a similar approach, representing each feed by a concatenation of all its entries. We derive the large document model as follows, ranking feeds by their posterior probability given the query

$$\begin{aligned} P_{LD}(F|Q) &= P_{LD}(Q,F)/P(Q) \\ &\overset{\text{rank}}{=} \underbrace{P(F)}_{\text{Feed Prior}} \quad \underbrace{P_{LD}(Q|F)}_{\text{Query Likelihood}} \end{aligned} \tag{5.1}$$

where the query likelihood component is estimated with dependence models and Dirichlet smoothing (Equations 2.3 and 2.1).

The feed prior component, shared between this model and the small document models introduced below, is used to incorporate query independent features into the ranking algorithm. See Section 5.2.2 for a detailed explanation of feed priors and how they are used in our models.

### 5.2.2 Small Document Models

The next set of models treat blog feeds as collections of individual documents — the blog's constituent entries. Retrieving information sources as collections rather than single entities has been an effective approach in federated search, cluster based retrieval and expert finding [114, 72, 12]. Additionally, decomposing our retrieval task in this way enables us to model the relationship among entries or between the entry and the feed, measuring how "central" the entry's language is to that of the entire feed.

There are several reasons we may want to decompose our retrieval task in this way:

1. Although the unit of retrieval in this task is an entire blog or feed, the unit of consumption is an individual post. Recognizing this in the retrieval model has potential to improve retrieval performance.
2. The varying size of blog posts may lead to an imbalanced representation of the feed's language model. Without recognizing distinctions between entries and appropriately controlling for their varying length, larger than average entries would dominate the language model of the feed and potentially skew retrieval results.
3. The goal of this feed ranking task is to find feeds with a "central and recurring interest" on a given topic. By looking at individual entries instead of the entire feed, we can determine how "central" the entry's language is to that of the entire feed.

Keeping these concerns in mind, our small document model is derived as follows, again ranking feeds

by the posterior probability of observing the feed given the query

$$
\begin{aligned}
P_{SD}(F|Q) &= \frac{1}{P(Q)} \sum_{E \in F} P_{SD}(Q, E, F) \\
&\overset{\text{rank}}{=} P(F) \sum_{E \in F} P(Q|E, F) P(E|F) \\
&\overset{\text{rank}}{=} \underbrace{P(F)}_{\text{Feed Prior}} \sum_{E \in F} \underbrace{P(Q|E)}_{\text{Query Likelihood}} \underbrace{P(E|F)}_{\text{Entry Centrality}}
\end{aligned} \tag{5.2}
$$

where the last line holds if we assume queries are conditionally independent of feeds given the entry. This model scores feeds as a weighted average of their constituent entry scores. The entry weighting in this model is defined by the centrality component, described in more detail below.

The model above extends the one proposed in [7, 44], which is loosely based on the ReDDE federated search algorithm [114] and similar to Balog's Expert Search Model 2 [12]. Both of these models are described in more detail in Chapter 2.4. The ReDDE model favors large collections, a desirable property when ranking by the expected number of relevant documents. But in our task, high traffic blogs may not necessarily be more relevant than infrequently updated blogs. The ReDDE analog of our centrality component, $P(D|C)$, is uniform for each document aggregate. We extend this to a true measure of centrality rather than simply a means to balance collections of different sampled sizes.

Each component of our small document model is explained below.

### Query Likelihood

The query likelihood component of our small document model is estimated similarly to the large document model, using the same full dependence model query features and smoothing via with two-level Jelinek-Mercer smoothing. This allows combination of evidence from the entry, feed and collection. See Section 2.1.2 for details on this smoothing model. Although the small document model cannot be completely expressed in the Indri query language, the query likelihood scoring is identical to a dependence model query, retrieving entries rather than feeds.

### Entry Centrality

The entry centrality component of our model serves two purposes. First, because we want to favor relevant entries that are also representative of the entire feed, the centrality component measures how closely the language used in the entry's text resembles the language of the feed as a whole. This has the effect of down-weighting the influence of an outlier entry that happens to be relevant to the query.

The second purpose of the $P(E|F)$ component is to balance the scoring across feeds with varying numbers of entries. Without this balancing, the summation in the small document model, Equation 5.2, would favor longer feeds.

Our entity centrality component is proportional to some measure of similarity between the entry and the feed, $\phi$, normalized to be a probability distribution over all the entries belonging to this feed

$$
P(E|F) = \frac{\phi(E, F)}{\sum_{E_i \in F} \phi(E_i, F)}. \tag{5.3}
$$

In general, any measure of similarity could be used here, for example, K-L divergence or cosine similarity. In our experiments we evaluated two centrality scoring functions. As a means to assess the effect of the

centrality component of our model, our first scoring function is uniform, i.e. no centrality computation

$$\phi_{CONST}(E, F) = 1.0$$

and the centrality component of our model using this scoring function only serves to normalize for feed size. The second scoring function computes a centrality measure based on the geometric mean of term generation probabilities, weighted by their likelihood in the entry language model

$$\phi_{GM}(E, F) = \prod_{t_i \in E} P(t_i|F)^{P(t_i|E)} = \left( \prod_{t_i \in E} P(t_i|F)^{\frac{n(t_i,E)}{|E|}} \right) \tag{5.4}$$

where we estimate the feed language model as follows, again taking care to control for varying entry lengths

$$P(t_i|F) = \frac{1}{N_F} \sum_{E_j \in F; j=1}^{N_F} P_{MLE}(t_i|E_j).$$

This scoring function is similar to the un-normalized entry generation likelihood from the feed language model.

In our implementation, the product in Equation 5.4 is only performed over the query terms, thereby providing a topic-conditioned centrality measure biased towards the query. Additionally, significant efficiency improvements can be realized by only taking the product over the query terms rather than the entire entry vocabulary.

In some sense, the entry centrality term in our model is similar to Hannah et. al.'s blog cohesiveness measure [55]. However, our centrality measure is more appealing in several ways: (1) it has a direct probabilistic interpretation in the model, (2) it gives an entry-specific score instead of a global feed score, and (3) as described above, this score can be conditioned on the query, providing a query-specific centrality measure.

The formulation of our centrality measure, Equation 5.3, has the tendency to inflate the scores of entries belonging to shorter feeds. This can be seen in the distribution of the length of retrieved feeds by the large document and baseline small document models, shown in Figure 5.2. In this figure, we see that the feeds retrieved by the small document model (in red) are on average smaller than those retrieved by the large document model (in blue). Discounting this normalization may be one way to control for this, however in this work we chose to use the feed prior as a means to favor feeds based on their size. This in in keeping with previous work on fielded retrieval and other document aggregate ranking tasks [95, 80], and separates the centrality and feed size components of our feed ranking model.

### Feed Prior

The feed prior component, $P(F)$, provides a way to integrate query-independent factors into the feed ranking. Previous uses of document priors in the Indri retrieval framework include favoring documents with shorter URLs in homepage finding tasks or higher PageRank values in web search tasks [95, 23]. In this work we use the feed prior to favor longer feeds, which without any knowledge of the query are more likely to contain relevant entries. This also has the effect of controlling for the overly-optimistic centrality scoring for short feeds.

We evaluate two feed priors in this work: one which grows logarithmically with the feed size, $P_{LOG}(F) \propto \log(1 + N_F)$, and a uniform feed prior that does not influence the document ranking at all, $P_{UNIF}(F) \propto$

1.0. Note that our small document model is very similar to the ReDDE model if we use the constant entry centrality measure, $\phi_{CONST}$, and choose a prior that grows linearly with the size of the feed, $P_{LIN}(F) \propto N_F$. Initial testing with a linear prior for this task, however, yielded degraded performance.

Figure 5.1 shows a comparison of the effective entry weight when using the $P_{LOG}$ and $P_{UNIF}$ priors, assuming $\phi_{CONST}$ centrality measure. These effective entry weights, shown at the left, are given by

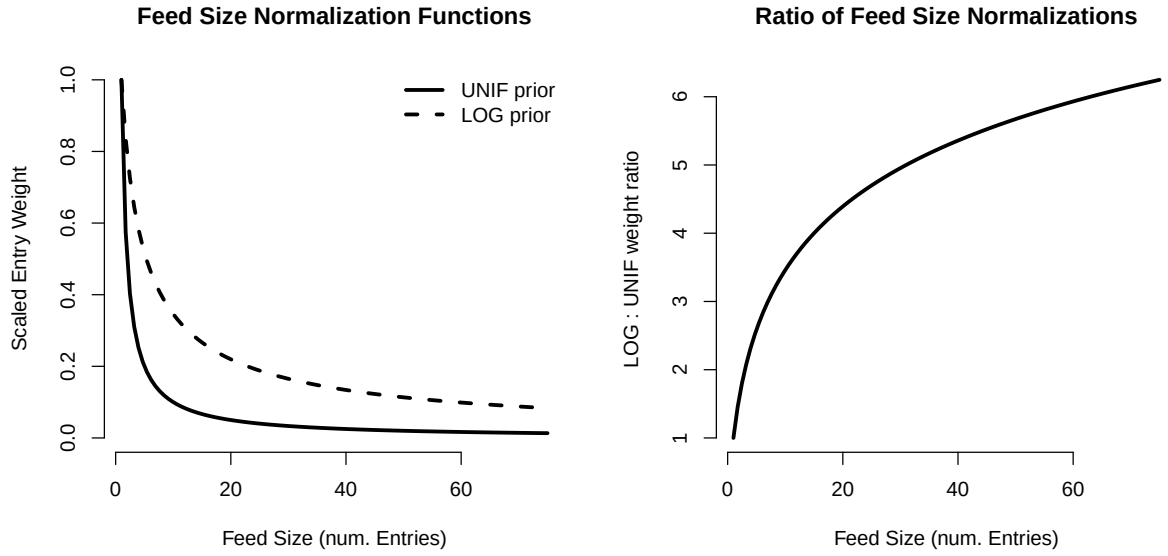



**Figure 5.1:** Left: Effective entry weighs for varying feed sizes, assuming $\phi_{CONST}$ centrality measure. Entry weights scaled so that a feed with a single blog post has a weight of 1.0. Right: Ratio of $P_{LOG}$ to $P_{UNIF}$ with $\phi_{CONST}$ centrality measure. This figure shows the $P_{LOG}$ prior places a higher weight on longer feeds than the $P_{UNIF}$ prior, for example roughly 5.6 times more weight per post on feeds with 50 posts.

the feed prior and entry centrality measures: $P(F)P(E|F) = P(F)/N_F$. To the right, we show the ratio of the uniform feed prior to log prior. This figure makes it clear that the $P_{UNIF}$ prior, which does no adjustment of the weighted averaging from the centrality component of our scoring function, has a tendency to place a much higher weight on entries from very short blogs. The $P_{LOG}$ prior, although still placing a higher weight on entries from short blogs, more gradually decreases the weight as blog size increases, relatively favoring entries from longer blogs more than the uniform prior. For blogs with 50 posts, for example, the $P_{LOG}$ prior relatively weights those entries 5.6 times more than uniform prior.

## 5.3 Retrieval Model Experiments

We evaluated these models using the 45 topics and relevance judgements from the 2007 TREC Feed Distillation task on the BLOG06 test collection [78, 82], using only the topic title text. This collection is described in more detail in Section 3.1. As stated above, this task is ranking blog *feeds* in response to a query, not blog posts. BLOG06 is a collection of blog home pages, blog entry pages (permalinks) and XML feed documents. For these tests, we chose to index only the feed XML documents. Although these documents potentially contain partial content of the blog posts rather than the full text, they tend to be less noisy. The feed documents typically do not contain advertisements, formatting markup or reader comments, all of which could lead to degraded retrieval performance. We index the feeds as structured

documents containing a series of <entry> elements for each feed entry, allowing index reuse across experiments.

All results reported are from 5-fold cross validation to choose the smoothing parameters used in the query likelihood calculations described above, and all experiments were performed with an extended version of the Indri search engine.

Our evaluations focused on the following questions:

1. Does a small document retrieval model that attempts to control for varying entry length outperform the large document retrieval model that treats the feed as a single bag-of-words?
2. Does a measure of entry centrality further improve performance? and
3. What is the effect of feed length?

| Model | Prior | Centrality | MAP | P@10 |
|---|---|---|---|---|
| **LD** | $P_{UNIF}$ | - | 0.290 | 0.400 |
| **SD** | $P_{UNIF}$ | $\phi_{CONST}$ | 0.277 | 0.391 |
| **SD** | $P_{UNIF}$ | $\phi_{GM}$ | $0.290^{\dagger}$ | $0.409^{\dagger}$ |
| **LD** | $P_{LOG}$ | - | 0.188 | 0.320 |
| **SD** | $P_{LOG}$ | $\phi_{CONST}$ | $0.298^{+}$ | $0.418^{+}$ |
| **SD** | $P_{LOG}$ | $\phi_{GM}$ | $\mathbf{0.315}^{\dagger+*}$ | **0.424** |

**Table 5.1:** Mean Average Precision and Precision at 10 for the large document (LD) and small document (SD) retrieval models with different centrality measures and different feed priors (Section 5.2.2). Statistical significance at the 0.05 level is indicated by † for improvement from $\phi_{GM}$, $+$ for improvement from $P_{LOG}$ and $*$ for improvements over the best LD model.

The full set of results is presented in Table 5.1 with significance testing performed with the Wilcoxon Matched-Pairs Signed-Ranks Test. First, looking at the top three rows using the uniform feed prior $P_{UNIF}$, we can see that the large document and small document retrieval models perform comparably when using the centrality measure ($\phi_{GM}$), but without the centrality measure ($\phi_{CONST}$) the large document model outperforms the small document model. Next, when using the logarithmic feed prior $P_{LOG}$, the small document model clearly outperforms the large document model. We see an improvement in the small document model performance of roughly 4.5-5.5% when applying the $\phi_{CONST}$ centrality measure and an improvement of 7.5-8.5% when applying the logarithmic prior. The best small document model performance ($\phi_{GM}$ and $P_{LOG}$) significantly outperforms the best large document model ($P_{UNIF}$), with an improvement in MAP of more than 8.5%. Using the centrality measure $\phi_{GM}$ clearly helps the small document model performance across tests. The feed prior has the opposite effect on the small and large document models, significantly hurting performance on the large document model and helping on the small document model.

Figure 5.2 shows the distribution of the sizes of retrieved feeds across a sample of the queries for the baseline large and small document models. In this figure, we see a clear tendency of the small document model with the uniform prior $P_{UNIF}$ to retrieve shorter feeds. The correction by the logarithmic prior $P_{LOG}$ aims to correct this tendency.

There are several conclusions we can draw from these results, answering the research questions above:

1. The small document model with uniform prior and constant centrality measure is an attempt to control for varying blog post length. The feed scoring is uniform across blog posts in this model, whereas long posts may have a tendency to dominate a feed scoring in the large document model that

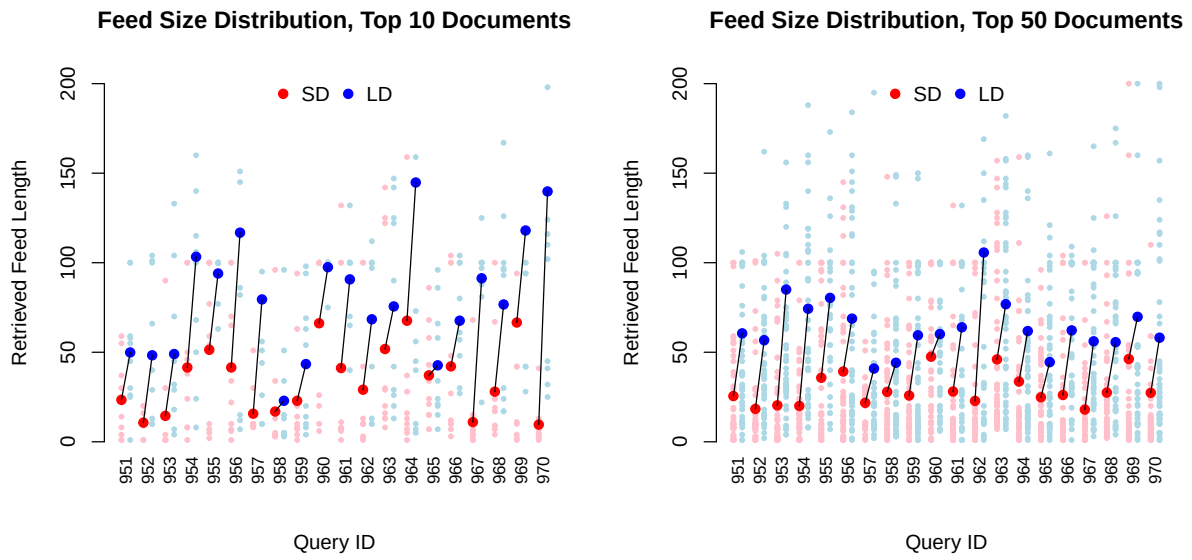



**Figure 5.2:** Size distribution of the top retrieved feeds at two different rank cutoffs by the baseline SD model ($\phi_{CONST}$ + $P_{UNIF}$, shown in red) and the LD model (shown in blue). Dark circles indicate the mean feed size retrieved in the top 10 or 50 documents for that query, and light circles represent sizes of each retrieved feed. SD model consistently across queries retrieves shorter feeds. Note: vertical axis is truncated to show the low-frequency distribution.

simply concatenates posts. Despite attempting to control for this effect, the naïve small document model (**SD**+$P_{UNIF}$ + $\phi_{CONST}$) performs poorly.

2. The centrality component ($\phi_{GM}$) is essential for the small document model to outperform the large document model. With either prior ($P_{UNIF}$ or $P_{LOG}$), the small-document model that includes the centrality component outperforms the one without, and performs as well or better than the large-document model.
3. The benefit of the length-biasing prior term is due to its counteracting of the small-document model's tendency to place too-high a weight on small feeds, rather than than an intrinsic feature of this task or dataset. The logarithmic prior substantially hurts the large-document model, but is an effective means to control for the tendency of the small-document model to favor short feeds.

Figure 5.3 shows per-query change in Average Precision between the **LD** model and the best performing **SD** model. In this figure, we see that the great majority of queries are helped by the application of the **SD** model (76%), and for those queries that are hurt, the AP difference is generally less than 0.1.

## 5.4 Previous Work on Blog Feed Search

In this section, we provide details of methods employed at the Text REtrieval Conference (TREC) and followup work. We focus on three aspects of retrieval models that have been studied with respect to this task:

- Comparing large document and small document models, particularly with regard to how to aggregate post-level scores into a blog-level score,

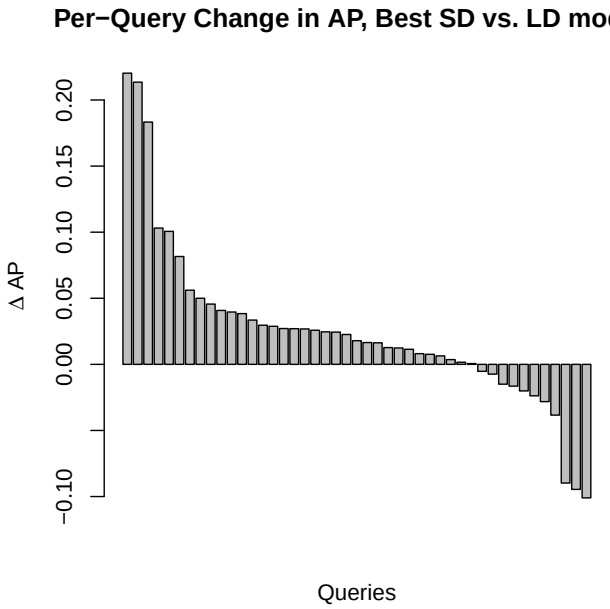



**Figure 5.3:** Per-query change in Average Precision between the **LD** model and the best performing **SD** model, $P_{LOG} + \phi_{GM}$.

- Methods to control for feed length, and
- Methods to take into account the topical dispersion of the posts within a blog.

### 5.4.1 Large and Small Document Models

The general theme of evaluating large-document and small-document retrieval models for blog feed search was explored by Macdonald and Ounis [80], Seo and Croft [110] and Balog et al. [15].

Several voting methods for aggregating blog post scores into feed scores were applied by Macdonald and Ounis [80]. The methods used were originally proposed to combine results from multiple search engines [52], but have also been shown to be effective in other tasks such as Expert Finding. Macdonald and Ounis found that a large-document model is most effective when the underlying index is the XML index, as used here, but a small-document model with an appropriate aggregation scheme is effective when using the HTML index. The aggregation scheme that worked best in their experiments was *expCombMNZ*, which favors blogs with many posts retrieved, as well as high-scoring posts.

Seo and Croft adapt the the Pseudo-Cluster Selection model (PCS, described in Chapter 2) to this task. This small-document model scores a feed based on the top-scoring $k$ documents with respect to the query. They found that the large-document model outperforms this small-document model, but some gains over the large-document model can be realized by combining the two.

Balog et al. apply models previously shown to work well in expert search to the task of blog feed search [15]. The expert models applied, referred to as the "Blogger Model" and "Posting Model" are quite similar to the Large and Small document models here. The authors found that, similarly to Seo and Croft, the large-document "Blogger Model" is more effective at the task of feed ranking.

These three studies, as well as ours, point to different conclusions. The probabilistic language modeling approaches of Seo and Balog both found the large-document models superior for this tasks. These

results agree with ours, that without differentially weighting the blog posts in a small document model, the large document model is more effective. But, Macdonald's voting method, which departs from the language modeling framework, shows the opposite when applied to the HTML collection.

### 5.4.2 Controlling for Feed Length

Previous studies have investigated whether there is benefit to controlling for the number of posts in a feed when ranking feeds. Macdonald and Ounis found that application of a length normalization proportional to $\log(1/N_F)$ yields moderate improvement. Compared to our logarithmic feed prior $P_{LOG}$ , this length normalization has the opposite effect, favoring *shorter* feeds rather than our approach that favors *longer* feeds. These seemingly opposing findings are likely due to the differences in our feed scoring functions — our centrality component tends to favor short feeds, whereas their voting method tends to favor longer feeds. The length normalizations, thus, work best when they cancel-out the natural tendencies of the feed scoring algorithms.

### 5.4.3 Accounting for Topical Dispersion

The guidelines of the TREC Blog Distillation Track state that relevant blogs will have a "central and recurring interest" in the topic of the query, and this has led to several research studies investigating methods to measure and integrate a centrality measure.

Macdonald and Ounis [80] and Hannah et al. [55] attempt to capture some sense of the overall cohesiveness of the blogger's interests. This cohesiveness is measured by the average cosine similarity between blog posts and the blog as a whole, and is used to adjust the final feed score. In their experiments applying the cohesiveness measure as an adjustment to the output of their voting model, this gross cohesiveness has a generally negative effect on the overall performance.

He et al. developed a similar measure, termed coherence, also an average cosine calculated over every pair of posts in a blog [56]. Naïve applications of their coherence measure fail to improve performance, but more sophisticated re-weighting of the coherence values, taking into account per-query normalizations, show a modest improvement.

All of the other measures proposed to take into account topical dispersion of the blog posts are query-independent measure of topical dispersion of the blog. Our method, the $\phi_{GM}$ centrality measure, is only calculated over the query terms, thereby biasing the topical dispersion to only the topic of the query. As opposed to the other measures proposed, we do see a significant and consistent improvement in performance with this approach.

## 5.5 Conclusion

In this chapter, we have presented our experiments on blog feed search. We developed several probabilistic feed retrieval models, showing that existing federated search algorithms can be effectively adapted to this task.

The large document model, which treats the feed as a single monolithic document, is used as a strong baseline. This model performed as the best non-expanded submission at the 2007 TREC Blog Distillation task. The large document model is difficult to outperform with a naïve small-document that scores posts individually and aggregates those scores into a feed score.

Our primary contribution in this chapter is the demonstration that a small document model *can* be adapted to outperform the large document model by considering two important factors. First, modeling the topical relationship between the individual posts and the blog as a whole can yield improvement. Second,

additional improvements are realized by controlling for the natural bias of the small-document model towards short feeds. Both of these techniques together yield a strong performing small-document model with significant and substantial improvement over the large document baseline — a 9% improvement in MAP and an 6% improvement in P@10.

Interestingly, this result is contrary to those previously published by us and others [7, 44, 109, 15] and demonstrates the need to effectively model the topical relationship between the feed and its entries. The major contribution of the small document model presented here is that it provides a novel and principled mechanism to measure the topical relatedness of the document to its collection and to integrate that into the retrieval algorithm.

The retrieval models presented here are not specific to blog feed retrieval and may have applications beyond this task. The small document model presented here can be sensibly applied to any retrieval problem where collections of topically related documents are ranked, including email or newsgroup thread retrieval, web results collapsing, cluster-based retrieval, and other federated search tasks. As part of the proposed work, we will extend these retrieval models and apply them to tasks beyond blog feed search.

# Chapter 6

# Online Forums Retrieval Experiments

In this chapter we present experiments with the two online forum datasets previously discussed in Chapter 3. First we explore a variety of different thread retrieval models and evaluate with both the MacRumors.com and Ancestry.com test collections. Second, we evaluate methods for integrating broader collection structure into the ranking algorithm, such as subforum or authorship information.

The questions we aim to answer with these experiments are as follows:

1. What are effective retrieval models for thread search, specifically methods for aggregating message scores into thread scores?
2. Can we improve retrieval performance by including content from broader structural elements in the collection, for example co-authored documents or the subforum?

### Chapter Roadmap

Our exploration of the thread search task in this chapter is organized as follows:

- In **Section 6.1** we describe retrieval models for ranking message threads. These models are characterized as *inclusive* models that use the content of every message in the thread and *selective* models that discard some of the messages in the thread. We perform experiments on two online forums collections and find that for this task, the selective models significantly and consistently outperform the inclusive models.
- In **Section 6.3** we discuss methods to include author information into the thread ranking algorithm. We present several models for integration of author information such as smoothing the post language model with the language model of the author and score combination. While in general these techniques are not found to provide any significant improvement over the baseline, the longest queries in our collection are helped significantly with an almost 10% improvement.
- In **Section 6.4** we present experiments integrating information from the thread's corresponding subforum into the thread scoring model. We fail to find any significant effect with any model to include subforum content.
- **Section 6.5** concludes the chapter.

### Research Contributions

The major research contributions presented in this chapter are

- An evaluation of document aggregate ranking models applied to message thread search. We find that representing a message thread as individual message documents is superior to representation

as one large thread document. We also find that selective retrieval models that discard some of the messages in the thread when scoring outperform inclusive models that use all messages.

- We demonstrate that for long and multifaceted queries, expanding the message language model with a language model built from co-authored messages can significantly improve performance.

## 6.1 Thread Retrieval Models

In this section we describe a variety of retrieval models that can be applied to thread search in online forums. The baseline models described here are taken from previous work in blog feed search [45], resource selection [109] and expert finding [80]. These models are described in more detail in Chapter 2. These initial models focus only on the structure of the message thread, and ignore any other structure in the collection, such as authorship or subforum information.

Thread search shares some similarities with blog feed search — these are both tasks of ranking aggregate documents, and both are ranking some form of user generated content. These tasks differ in three significant ways, however.

First, all the posts in a blog are usually written by a single author, whereas each message in an online forum thread could be written by different authors. On average, in the Ancestry.com forum, threads have 1.86 ($\pm$ 2.04) authors, and when looking only those with more than one message, this increases to 3.29 ($\pm$ 2.79) authors per thread. Figure 6.1 shows a distribution of the number of unique authors per thread. This figure shows that, while the average number of authors per thread may be relatively small, for a small

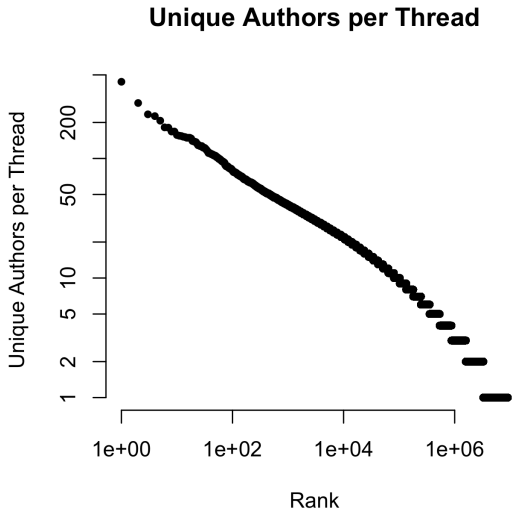



**Figure 6.1:** Distribution of the number of unique authors per thread in the Ancestry.com Forum collection.

number of threads, an extremely large number of authors participate in the conversation.

Second, the blog posts making up the blog feed tend to be longer and noisier than online forum messages. In our blog post index, documents have on average 1190 tokens each, whereas the Ancestry.com forum messages are on average 95 tokens. The blog posts are also raw HTML documents, containing boilerplate content, possibly advertisements and often a series of comments. The message board threads, although using equally informal language, do not generally contain off-topic comments, advertising, or

boilerplate content. We also index the message board documents as regularly structured documents, rather than relying an HTML parser to handle potentially malformed HTML structure.

Third, the blog feed task is defined as finding feeds with a "central and recurring interest" in the topic of the query. This differs from the thread search task, which is much more similar to traditional document retrieval tasks. It is unclear whether the techniques developed for blog feed search that specifically take into account centrality measures are equally applicable to the task of message thread search.

Below, we describe two classes of thread retrieval models. The *inclusive* models utilize all the content of the thread in scoring, making the assumption that every message makes a positive contribution to the final thread score. The *selective* models ignore some messages when calculating the final thread score. These models make the assumption that some of the messages are extraneous, possibly off-topic, or possibly too sparse to provide useful information for the ranking algorithm.

### 6.1.1 Inclusive Thread Ranking Models

The initial models we describe for thread search are based on those successfully applied to blog feed search. We first aim to understand whether models effective for ranking blog post aggregates (feeds) are equally effective at ranking message aggregates (threads). That is, to what extent do we have a similar goal in mind — to identify threads with a "central" focus on the topic described by the query. To this end, we apply blog search models directly to the task of thread ranking. The analogue of the large- and small-document models used for blog feed search (Equations 5.1 and 5.2) are given below:

$$P_{LD}(T|Q) \overset{\text{rank}}{=} \underbrace{P(T)}_{\text{Thread Prior}} \quad \underbrace{P(Q|T)}_{\text{Query Likelihood}} \tag{6.1}$$

$$P_{SD}(T|Q) \overset{\text{rank}}{=} \underbrace{P(T)}_{\text{Thread Prior}} \sum_{M \in T} \underbrace{P(Q|M)}_{\text{Query Likelihood}} \underbrace{P(M|T)}_{\text{Message Centrality}} \tag{6.2}$$

where in this case we are ranking thread ($T$) with respect to a user's query ($Q$) rather than blog feeds. As before, we can view the large document model as representing threads as a concatenation of their respective messages and estimating probabilities based on that "large" document. The small document model, on the other hand, assigns scores to messages ($M$) and then aggregates those scores into a thread score via a weighted average.

As with blog feed search, we may be interested in differentially weighting individual messages in a thread, for example favoring messages that more closely resemble the "central" topic of the thread. We use alternative estimations for the message centrality component $P(M|T)$ to do this, integrating a measure of the affinity of a message to the thread via the message centrality component of the small document model. In this work, as before, we investigate both a uniform message centrality measure ($\phi_{CONST}$), as well as a centrality measure based on the generation probability of the message given the thread ($\phi_{GM}$). See Section 5.2.2 for a detailed description of how these two centrality measures are estimated.

These retrieval models view all the messages in the thread as important for thread ranking, although possibly re-weighted with a centrality measure.

### 6.1.2 Selective Thread Ranking Models

In addition to the large- and small-document inclusive retrieval models, we also apply several thread retrieval models that rank threads by selective scoring of their constituent messages. In these models, we make the assumption that the every message in the thread is not necessarily useful for thread ranking,

and by discarding some messages we can improve the overall retrieval performance. The models studied here are variants of Liu's cluster representation models [72, 73] adapted to the thread search task. These models are described in detail in Section 2.4.2 and we give a brief overview here.

The first of the selective ranking model is based on the cluster selection model proposed by Liu and Croft [72] and applied to both blog feed search and forum thread search by Seo et al. [110, 108]. This model, referred to as Pseudo-Cluster Selection (PCS), is given by the following:

$$P_{PCS}(Q|T) = \left( \prod_{1 \leq i \leq k} P(Q|M_i) \right)^{\frac{1}{k}} \tag{6.3}$$

where the $i$ indexes the top-scoring $M_i$ messages in the thread with respect to the query $Q$. If less than $k$ messages belong to the thread, the product is padded by the minimum-scoring message retrieved for this query. For the experiments presented here, we let $k = 5$ which has shown good performance in other tasks such as blog feed search. This model has been shown empirically to perform well across a variety of tasks such as cluster-based retrieval [73] and blog feed search [110], and is theoretically grounded as a method to represent the centroid of the set of documents assuming the documents lie on a Riemannian manifold defined by the Fisher information metric [112].

Two additional selective retrieval models are also applied to this task. These two models score a thread based on a single message in that thread, either the start message or the single highest scoring message in the thread.

$$P_{ST}(Q|T) = P(Q|M_{START}) \tag{6.4}$$

$$P_{MAX}(Q|T) = \max_{M_i \in T} P(Q|M_i) \tag{6.5}$$

where $M_{START}$ is the thread's start message. This retrieval model is specific to to message threads, which have a temporal and logical ordering of the messages. Note that the $P_{MAX}$ ranking model is equivalent to the pseudo-cluster selection model with $k = 1$.

## 6.2 Experiments & Analysis

In this section we present initial experimental results using the baseline retrieval models for message thread search. We first look at the results on the MacRumors.com forum, and then turn our attention to the Ancestry.com forum.

### 6.2.1 MacRumors.com Forum

As stated above, the construction of the MacRumors.com forum may suffer from low coverage of the relevance judgements. For this reason, we must take care not to assume un-judged documents are not relevant. In this evaluation we take a recall-oriented approach, using as our primary evaluation measures are recall at various cutoff levels (R@10, 20, 30, 100).

These measures are particularly coarse and may not provide much distinction between ranking algorithms. For most queries, we have only a single relevant document. We also report Mean Reciprocal Rank (MRR), which is an evaluation measure typically used when there is a single known-relevant item in the collection. Although usually MRR is used to evaluate known-item finding tasks where there is one and only one relevant document, this is not the case here. We present MRR only to give a more sensitive evaluation metric, not to claim that this is a known-item finding task.

Performance results for the MacRumors.com forum are given in table 6.1. In this table we can see several clear trends. First, comparing at the large document ($LD$) and small document models ($SD$, $MAX$, $ST$ and $PCS$), the small document models consistently outperform the large document models. Second, the selective methods of thread ranking ($MAX$, $ST$ and $PCS$), which discard information in some of the messages when scoring the thread, outperform the methods that use all of the messages scores ($SD$). This indicates that when ranking threads in online forums, not all messages are useful in judging the relevance of a thread, and frequently the score of only a single message (as in $MAX$) achieves the best performance. In all evaluation metrics considered except R@100, the best performing selective model significantly outperforms the $LD$ and $SD + \phi_{GM}$ retrieval models using a 1-tailed paired t-test. In this dataset, there is not a significantly superior retrieval model among the selective methods.

| Retrieval Model | MRR | Recall at 10 | 20 | 30 | 100 |
|---|---|---|---|---|---|
| $LD$ | 0.0928 | 0.1553 | 0.2436 | 0.3113 | 0.5598 |
| $SD + \phi_{UNIF}$ | 0.0987$^{\dagger}$ | 0.2283$^{*}$ | 0.3481$^{*}$ | 0.4379 | 0.6448$^{*}$ |
| $SD + \phi_{GM}$ | 0.0922 | 0.1867 | 0.2993 | 0.4170 | 0.6239$^{*}$ |
| $ST$ | 0.1491 | 0.3095$^{*\dagger}$ | 0.3920$^{*}$ | 0.4587$^{*}$ | 0.5984 |
| $MAX$ | 0.1570 | 0.3253$^{*\dagger}$ | **0.4675**$^{*\dagger}$ | **0.5561**$^{*\dagger\bullet}$ | 0.6180 |
| $PCS$ | **0.1902**$^{*\dagger\bullet}$ | **0.3308**$^{*\dagger}$ | 0.4031$^{*}$ | 0.4158$^{*}$ | **0.6491** |

**Table 6.1:** MacRumors.com performance results, online forum thread ranking. Large Document Model ($LD$), Small Document Models with different centrality measures ($SD$), Max-Message ($MAX$), Start-Message ($ST$) and Pseudo-Cluster Selection ($PCS$) evaluated at Mean Reciprocal Rank (MRR) and Recall at various cutoffs. Symbols indicate significant improvement at the $p < 0.05$ level over the $LD$ model ($*$), $SD + \phi_{GM}$ model (†) and $SD + \phi_{UNIF}$ model ($\bullet$). We see that the selective methods consistently and significantly outperform the inclusive models, with the exception of the R@100 metric.

#### Hidden-Success Analysis & Limitations of the MacRumors.com Test Collection

As stated previously, due to the limited relevance information in the MacRumors.com test collection, we focus on recall-oriented evaluation measures. The absolute values of these performance metrics, however, indicates that even the best-performing retrieval algorithm may be performing unsatisfactorily. Most queries have only one relevant document, and a R@10 value of 0.37 indicates that 30 of the queries, more than half, failed to find a relevant thread in the top 10 results. A R@100 value of 0.68 indicates that approximately 15 queries failed to retrieve any relevant results in the top 100 documents.

We manually inspected the top 10 results of those failed queries. Through this, we identified more than 15 likely relevant results that were retrieved in the top 10. This is an indication that the relevance information in our test collection may be insufficient to adequately evaluate the thread ranking algorithms, particularly in a ranking task such as this when high-precision is likely to be an import factor. Although these experiments shed some insight into effective thread ranking algorithms, we turn our focus to the Ancestry.com collection, with a larger query set and richer relevance information, for the remainder of the online forum experiments.

### 6.2.2 Ancestry.com Forum

The above results on the MacRumors.com forum indicate that small-document models, and particularly selective models, are well suited for this task of thread search. We use these observations when applying

aggregation methods to build the document pool for the Ancestry.com collection. See Section 3.5.3 for details on the pooling procedure. We chose the same $SD + \phi_{UNIF}$, $MAX$ and $PCS$ models used above to each thread retrieval model when building the document pool. Table 6.2 gives the performance of each system and aggregation method used in the document pool. We show both $ppref@10$ and $mAPpref$.

Note that not all the message ranking algorithms used in the pool are interpretable as a query likelihood probability, $P(Q|M)$, as in Indri. For those other message retrieval models, we apply similar mathematical transformations to the message scores to produce a thread score. For example, the $SD + \phi_{UNIF}$ aggregation method simply becomes an unweighted arithmetic average of the message scores, referred to as $MEAN$ in the results table.

| System | Aggregation Method | $mAPpref$ | $ppref@10$ |
|---|---|---|---|
| **Indri** BOW | $SD + \phi_{UNIF}$ | 0.5428 | 0.4925 |
| | $MAX$ | 0.5998* | 0.5615* |
| | $PCS$ | 0.6569+* | 0.6402+* |
| **Indri** DM | $SD + \phi_{UNIF}$ | 0.5492 | 0.5067 |
| | $MAX$ | 0.6089 | 0.5716 |
| | $PCS$ | **0.6612**+* | **0.6466**+* |
| **Indri** fielded `#wsum` | $SD + \phi_{UNIF}$ | 0.4628 | 0.4291 |
| | $MAX$ | 0.5296 | 0.4985 |
| | $PCS$ | 0.5663 | 0.5420 |
| **Indri** fielded `#weight` | $SD + \phi_{UNIF}$ | 0.5422 | 0.4938 |
| | $MAX$ | 0.6105 | 0.5693 |
| | $PCS$ | 0.6572 | 0.6105 |
| **Terrier** $PL2$ | $MEAN$ | 0.5286 | 0.4934 |
| | $MAX$ | 0.5820 | 0.5381 |
| | $PCS$ | 0.6475 | 0.6225 |
| **Terrier** $InL2$ | $MEAN$ | 0.5406 | 0.4837 |
| | $MAX$ | 0.5828 | 0.5297 |
| | $PCS$ | 0.6554 | 0.6230 |
| **Zettair** $BM25$ | $MEAN$ | 0.5001 | 0.4586 |
| | $MAX$ | 0.5654 | 0.5271 |
| | $PCS$ | 0.6300 | 0.6083 |
| **Ancestry.com** | $MEAN$ | 0.4357 | 0.4036 |
| | $MAX$ | 0.4702 | 0.4481 |
| | $PCS$ | 0.5380 | 0.5808 |

**Table 6.2:** Pool system performance results for Ancestry.com. Significance tested for shaded rows only. Significant gain at the $p < 0.01$ level over BOW $SD$ and $MAX$ models indicated with * and + respectively.

There are two dimensions along which we can evaluate these results — the aggregation method, and the underlying retrieval system. First, looking at the aggregation methods, the results here agree with those on the MacRumors.com dataset. As before, we see a clear superiority of the selective models ($MAX$, $PCS$) over the inclusive model ($SD$/$MEAN$). For every input message ranking algorithm, and both $ppref@10$ and $mAPpref$, we observe that the $SD$ model is inferior to the $MAX$ model, which is

inferior to the $PCS$ model. We test significance of the three message aggregation methods using the Indri Bag of Words ranking model, and find that both $PCS > MAX$ and $MAX > SD$ at the $p < 0.01$ level.

Next, looking at the underlying message retrieval systems, these results are generally well aligned with our expectations. None of the systems used in the pool use any training data to set parameters, relying on default values or intuition to set field weights and smoothing levels. Indri and Terrier are modern, well maintained and widely used retrieval systems that consistently do well at the Text REtrieval Conference (TREC) (see, for example, in the Blog Track [82, 100, 102]). The Terrier models do not use term proximity or order, and the results with the $InL2$ model are almost indistinguishable from the analogous Indri bag-of-words (BOW) model. Indri's Dependence Model (DM) does leverage term proximity and order, and as expected performs slightly better than the models without this information. The performance difference between Indri's BOW and DM models is somewhat less than observed in other collections [86] and none of these differences are significant for this task and collection.

Interestingly, the fielded retrieval models do not perform as well for this task as expected. Typically, using within-document field structure in retrieval models yields significant performance improvement in tasks such as known page finding and xml element retrieval [95]. In the models used for these experiments the fields and field weights were chosen in a somewhat ad-hoc manner, using intuition and a handful of sample queries to choose the weights before any relevance assessment was performed. For this reason, it is unlikely those weights are optimal for this task, and further tuning of weights using training data will likely result in improved performance. Investigating the use of fielded retrieval models for forum search will require further experiments, but is out of the scope of the current work.

The Zettair search engine uses an implementation of the Okapi BM25 retrieval model [60] and is designed to be compact and fast search engine. This search engine is also used actively at TREC, performing generally well [53, 128]. It is unclear why Indri and Terrier consistently outperform Zettair in the thread search task, possibly having to do with Zettair's default parameters not being well suited to this tasks.

The Ancestry.com retrieval system uses a ranked boolean model, a less sophisticated ranking model than the other systems evaluated. This retrieval model is designed for use with richly structured query forms, including name, location and date fields as well as keywords. For the purposes of these experiments, all fields are treated as keywords, possibly contributing to its lower performance as compared to other systems.

We use Indri's Bag of Words model as our baseline going forward, as it is simpler and faster than the Dependence Model, with no significant difference in performance for this dataset.

### Query-Type Performance

As stated in Chapter 3, we selected a sample of Ancestry.com queries to be representative of the population of queries received at the site. These include roughly half *name* queries that consist of only a person's name, and half *name+* queries that include a location or other keywords in addition to the person name. Table 6.3 shows the performance characteristics of our three aggregation models across the different query types using the Indri BOW message retrieval baseline. Here, we see an identical trend in the effectiveness of the aggregation methods ($SD < MAX < PCS$) across both query sets. Again the difference in performance of the aggregation methods is significant within both query sets. Figure 6.2 shows the per-query change in performance between the $PCS$ model and the other two, for both the *name* and *name+* query sets. In this figure, we see that 55-70% of the queries are helped across the different query sets, with a more consistent improvement on the *name+* queries.

| Query Set | Aggregation Method | $mAPpref$ | $ppref@10$ |
|---|---|---|---|
| *name* | $SD + \phi_{UNIF}$ | 0.5718 | 0.5359 |
| | $MAX$ | 0.6398* | 0.6124* |
| | $PCS$ | **0.6813*** | **0.6640*** |
| *name+* | $SD + \phi_{UNIF}$ | 0.5128 | 0.4476 |
| | $MAX$ | 0.5585* | 0.5090* |
| | $PCS$ | **0.6318**$^{+*}$ | **0.6155**$^{+*}$ |

**Table 6.3:** Performance of Indri's Bag of Words model across the *name* and *name+* Ancestry.com query sets. Significant gain at the $p < 0.01$ level over the $SD$ and $MAX$ models indicated with $^*$ and $^+$ respectively. The selective aggregation methods, and in particular the $PCS$ method consistently perform well across both the *name* and *name+* query sets.

### 6.2.3 Conclusion

The experiments conducted in this section aim to shed light on which retrieval models are effective for aggregating message scores into thread scores. In both the MacRumors.com Forum and Ancestry.com Forum test collections, we find that *selective* models that ignore the scores of some of the messages in the thread are significantly more effective than *inclusive* models that use all the message scores. These results agree with those of Seo et al.'s work on forum thread search [108].

Additionally, we evaluate two different subsets of the queries for the Ancestry.com Forum dataset. The *name* queries, including only a person's name and no additional information, tend to be tend to be "easier" queries than the *name+* queries which include additional information. The the best $mAPpref$ and $mppref10$ values are roughly 8% higher on the *name* queries than the *name+* queries. In both query sets, again, the selective models tend to significantly outperform the inclusive models.

These findings have several implications for web search systems that may index online forum content. Online forum websites typically display a message thread in a single HTML document, with messages ordered sequentially (see Figure 3.2). Retrieval and indexing systems that treat message threads similar to other web documents, i.e. taking a large-document approach, are likely to be less effective than systems that are aware of the message-thread structure and can score messages individually. This approach is unlike the standard approach to web retrieval, where there is a one-to-one correspondence between documents in the index and crawled URLs. Indexing messages individually is also an appealing approach in a dynamic environment like online forums when new messages could be added to old threads at any time. Rather than re-indexing an entire message thread when a single new message is added and invalidating the existing thread document, indexing at the message-level allows incremental updating of the thread.

One explanation for the superior performance of the selective models over the inclusive models is that the selective models implicitly normalize for thread length. These models use only one message ($MAX$, $FIRST$) or the top-$k$ messages ($PCS$) to score the thread, and the same number of messages is used across all threads. The inclusive models, however, use a varying number of messages, potentially biasing those models towards threads with more or fewer messages. Several experiments were run in an attempt to make the $SD$ model sensitive to the number of messages in the thread, for example by including a thread length prior or a length-discounted calculation of $P(M|T)$ as in the Blog Feed Search experiments described in Chapter 5. These attempts to introduce length sensitivity in the inclusive models, however, consistently degraded performance. A more likely explanation of the success of the selective modes is their selectivity — some of the messages in the thread are not useful for scoring, and should be ignored.

Interestingly, these findings are considerably different than the findings in the Blog Feed search task,

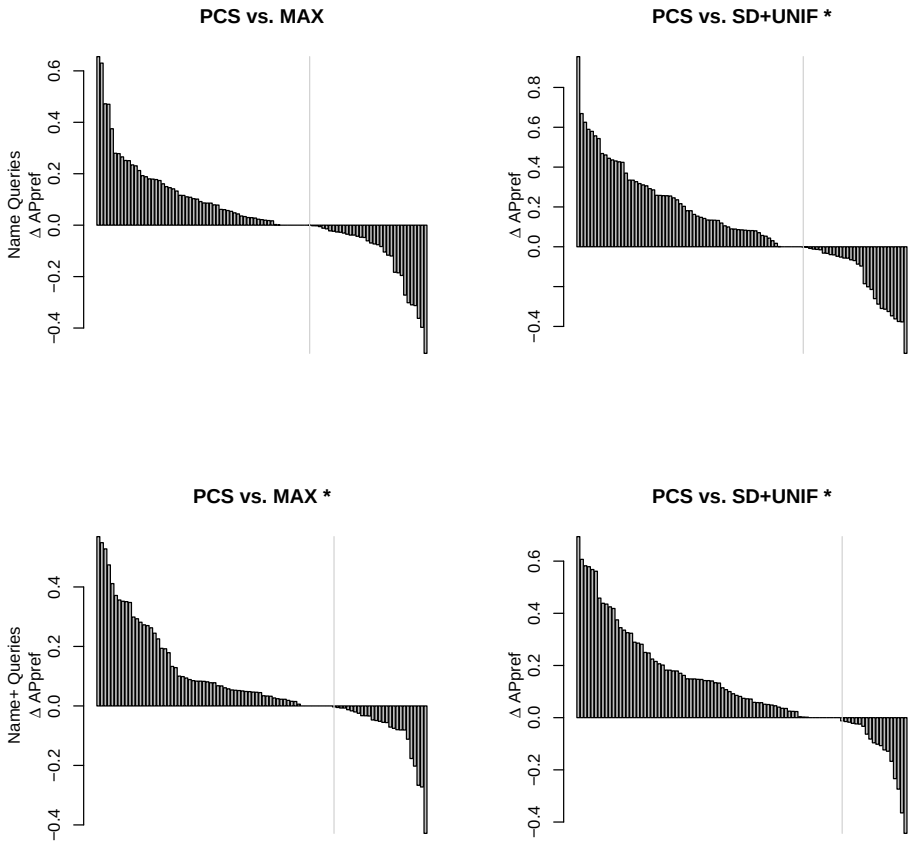



**Figure 6.2:** Per-query performance change across different thread aggregation methods. Change between $PCS$ and $MAX$ on the left, between $PCS$ and $SD + \phi_{UNIF}$ on the right, *name* query set on top and *name+* query set on the bottom. An asterisk (*) indicates significance at the $p < 0.01$ level.

presented in Chapter 5. In those experiments, we found that inclusive models worked well, and in particular the models that differentially weighted blog posts based on their similarity to the blog as a whole. Although these two tasks are tasks of ranking collections of documents — the blog is a collection of posts, and the thread is a collection of messages — different techniques are better suited for each task. There are several possible reasons for this:

1. A reader consumes the entire thread in the online forum search task. Although the relevance judgements in the blog collection are at the blog-level, the unit of consumption by a reader is the blog post.
2. The relevance guidelines differ in the two tasks. For this task, the assessors were instructed to marked more useful threads as preferred, whereas in the blog search task the assessors were instructed to favor blogs with an "ongoing interest" in the topic.
3. Although both collections are comprised of informally written documents, the forum collections are much cleaner, without advertising, comments, or boilerplate content.
4. The online forum messages are much shorter than blog posts.

In the following sections, we turn our focus to integrating content from throughout the collection into our ranking algorithms, specifically focusing on other content the message authors have written and the contents of the subforum containing the thread. Because of the larger query set and more complete relevance information, we run these experiments with only the Ancestry.com Forum dataset.

### 6.2.4 Related Work

Several previous lines of work have addressed search in online forums and related collections. The early years of the TREC Enterprise Track used a collection of public email lists from the W3C for several tasks [40, 39, 117]. These tasks include known item search and "discussion" search, both focusing on individual email messages as the item of retrieval. Participants at TREC, as well as subsequent studies, have found utility in combining content and structural features from messages and their containing thread [41, 98, 113]. Known-item search in newsgroups has also been studied by Xi et al. [129]. This study takes a learning-to-rank approach, combining content and structural features of messages, content from other messages in the discussion thread and previous behavior of the message authors such as the number of replies. The authors find several message and thread content features are most useful for ranking, but do not report the influence of other features such as author activity. Both of these tasks focus on message search, rather than thread search as studied here, and none use content features derived from other messages in the collection such as co-authored messages.

Seo et al.'s line of work investigating the extraction and use of hierarchical message threading structure [108, 113] is the most immediately relevant to that work presented above. In their work, the authors present a method for identifying hierarchical message-reply structure from a flat presentation of messages in a thread. Using this structural annotation of the thread, the authors then investigate methods of combining evidence from different "conversational contexts" within the thread to come up with a retrieval score for threads. These contexts include the entire thread, individual posts, and a "dialog context" — the reply chain of posts from the start message of the thread to messages at the leaf nodes.

Using the Pseudo-Cluster Selection (PCS) retrieval model, the authors score threads based on the top-$k$ of each context type within the thread. These context-sensitive thread scores are evaluated individually, as well as in combination. The authors find that there is some variability in performance across their test collections, but generally the combination of thread "thread context" (an inclusive model, analogous to the Large Document model above) and the "dialog context" performs the best. The authors' evaluation of a "post context" — corresponding to our $PCS$ model above — has somewhat inconsistent performance, and is usually outperformed by models that take into account the hierarchical conversation structure.

Although we do not investigate the use of hierarchical thread structure in our models, our results agree with most of Seo et al.'s findings. We found that the $PCS$ model outperformed the inclusive models, consistently across collections and evaluation measures. Their experiments showed the same for some online forum collections, but not all. The online forums collections evaluated in their work are considerably smaller than those evaluated here, with roughly 70% fewer queries, 80% fewer threads than the MacRumors.com collection and more than two orders of magnitude fewer threads than the Ancestry.com collection. See Table 3.6 for a comparison of the test collection sizes for those collections used here and in Seo et al. [108].

## 6.3 Integrating Co-Authored Content

One of the distinguishing characteristics of collections like online forums is the active participation of many different authors in the creation of the collection content. Previous behavior of an author, in partic-

ular the previous posts they have written, may give us clues as to the areas of an author's expertise. In this section, we investigate methods to integrate content from other message an author has written throughout the collection into the thread ranking algorithm.

The techniques we apply here are similar to those used in the blog post search tasks presented in Chapter 4. In both of these tasks, we score documents — blog posts, or forum messages — that share an author with other documents in the collection. In the blog collection, the documents with a shared author all belong to the same blog. There are several key differences between thread ranking and blog post ranking:

- Although blog posts may refer to each other, each post is most likely intended to serve as a standalone document. The authors in this collection, however, contribute messages that are most likely a part of a broader discussion, the message thread.
- The blog feed search task is one of ranking individual documents, but the thread ranking task is ranking document aggregates.
- By including content from authors' other messages throughout the collection, thread retrieval models may incorporate information from two orthogonal structural dimensions — the message-thread and the message-author organizations.

We explore several methods for including author content in the thread scoring. Given the association of an author with a message rather than the entire thread, the methods for integrating author content into the thread scoring focus on how to include the author content at the message scoring level. With these message scores including author information, the same aggregation models can be used to produce the thread score.

### 6.3.1 Models for Integrating Author Content

We take two primary approaches to integrating this author information. The first is to include the author content at the level of estimating the term generation probabilities, $P(q|M)$ for $q \in Q$, which generally take the form of smoothing models. This is an example of a *within-model* form of evidence combination, which has been shown to be effective in structured document search tasks [95]. The second approach is to generate query scores for the author language model and message language model separately, and then combine to generate the final message scores. The methods applied to achieve this include probabilistic mixture models or more general score combination models. These models are *model-agnostic*, able to take the output of any message-scoring and author-scoring technique to form a thread score.

We present the details of the within-model and model agnostic methods of including author content below.

#### *Within-Model* Use of Author Evidence

We first explore the efficacy of smoothing the message text with other text the author has written. This approach incorporates author information at the level of estimating term generation probabilities, rather than after author- or message-scores have been calculated. As before, we focus apply the two-level Dirichlet smoothing method proposed by Zhao and Callan [137]. Each message in the thread is smoothed individually with the corresponding author language model, before any aggregation method is applied. The two-level Dirichlet smoothing model is described in detail in Section 2.1.2.

To understand the motivation and benefit of integrating co-authored content at the term-level, consider

the standard language modeling formulation of the query likelihood, assuming term independence:

$$P(Q|M) \stackrel{\text{rank}}{=} \prod_{q \in Q} P(q|M)^{n(q,Q)}$$

where the message score is taken as the product over individual query term likelihoods. This product is essentially a probabilistic *AND* function, strongly penalizing a document if one or more of the individual query term probabilities is low [85]. By smoothing with the author language model when estimating these term probabilities $P(q|M)$, the goal is to ensure that those probabilities are not too low if an author has used the query term elsewhere in the collection, even if that query term does not occur in the message being scored. As we will show below, this technique has a stronger effect on long queries, which are less likely to have all query terms represented in a single message.

Note that we build the author's language model as a single distribution over terms including all the messages they have written throughout the collection. We do not consider building topic-sensitive author language models, such as through document clustering or biasing the model to messages likely to be related to the query. These techniques may be appropriate when an author has contributed a large number of messages in a variety of topical areas. We leave an exploration of more flexible an nuanced author language models for future work.

#### *Model Agnostic* Use of Author Evidence

In addition to the methods of integrating author-level information through estimation of the term generation probabilities, we can also integrate this evidence at the level of generating the message scores, $P(Q|M)$. For these models, we take an approach similar to the expert models found effective by Balog et al. [12], and the Large Document models previously applied to Blog Feed Search (Chapter 5).

Given an estimation of the query likelihood from the message $P(Q|M)$ and from the corresponding author's language model $P(Q|a(M))$, for example using the standard language modeling techniques presented in Chapter 2, we can apply standard probabilistic modeling techniques to combine the scores. Reestimating the message score as a mixture model of the two:

$$P_{mix}(Q|M) = \lambda P(Q|M) + (1-\lambda) P(Q|a(M)) \tag{6.6}$$

or a log-linear combination

$$P_{ll}(Q|M) = P(Q|M)^{\lambda} \times P(Q|a(M))^{(1-\lambda)} \tag{6.7}$$

in both cases, $0 \leq \lambda \leq 1$.

Additionally, we may interpret the probabilities as generic query scores, normalize and apply a simple score-combination method:

$$s(Q,M) = \text{norm}_q(P(Q|M)) + \alpha \, \text{norm}_q(P(Q|a(M))) \tag{6.8}$$

where $\alpha \in [-\infty, \infty]$ and $\text{norm}_q(\bullet)$ linearly scales the probabilities to the range $[0,1]$ with the shift-and-scale normalization used in Section 4.4.1.

#### Implementation Details

As with all the results presented in this section, we perform the experiments with Indri. Indri provides an implementation of the two-level Dirichlet smoothing model we present above, but the use of this

smoothing model requires that the collection be reformatted. In order to apply the two-level smoothing model, the message documents to be retrieved must be indexed as extents concatenated together to form an author pseudo-document. To retrieve messages after this re-formatting, we can construct our queries with Indri's extent restriction operator:

$$\#\texttt{combine}[\texttt{message}](< \texttt{originalquery} >)$$

Although this approach is functional, it has one major drawback. Indri scores every `message` extent within an author document if *any* message in that document contains a query term. For many queries, especially those that contain general terms, a large fraction of the messages in the collection are scored, even though they may not contain any of the query terms. If the queries are formed as above, this results in slow retrieval times.

We can avoid some of this overhead by changing our formulation of the Indri query in order to minimize which messages are scored by Indri:

$$\#\texttt{combine}[\texttt{message}](\#\texttt{filreq}(\#\texttt{syn}(< \texttt{name} >)\#\texttt{combine}(< \texttt{originalquery} >))$$

This formulation prohibits Indri from scoring any message without at least one of the tokens of the name portion of the query. For the *name* queries, the results are identical to the original query formulation, but for the *name+* queries we may not retrieve some documents that do not contain the name but do contain other tokens in the query. The assessment guidelines state that a document should be marked bad if it could not plausibly refer to the person in the query, but an assessor could still mark a document as preferred if it does not contain any of the name tokens in the query. Although we may miss some documents that could be preferred to other documents (i.e. not marked bad), we believe these are rare cases and this is an acceptable tradeoff for much faster query processing time.

When this `#filreq` query optimization is applied to the baseline bag-of-words model, we see less than 1% change in performance.

### 6.3.2 Experiments

For the experiments using the above models, we divide the query set into three folds for cross validation. For the full query set, we stratify each fold so that there is a balanced number of *name* and *name+* queries. The parameters $\alpha, \lambda, \mu_D$ and $\mu_C$ are fit via grid search. In all the following experiments, we use the $PCS$ aggregation method, which performed best in the experiments above.

Table 6.4 shows the performance results for the three query sets separately. Although we observe some performance improvements, none of these improvements are significantly better than the baseline. But, these results do indicate that some models that include author information may be more effective than others in certain situations.

The *name+* queries are naturally longer than the *name* queries, with 5.05 more words on average. The consistent improvement of the smoothing model ($P_{dir2}$) on the *name+* queries is an indication that this model may well suited for longer queries. Although we do not observe a significant improvement in these experiments on the *name+* queries, this query set contains queries with a considerable range in query length — from 3 to 14 terms. We assess at the performance of the models trained on only the longest queries from this query set, those *name+* queries with at least 8 words (46 queries). Again, we perform three-fold cross-validation, using grid search to tune the model parameters. On this subset of queries, we see a substantial and significant performance improvement of almost 10%. These results are shown in Table 6.5, with per-query $APpref$ differences shown in Figure 6.3. This figure shows significant improvements in *mAPpref*, but the improvements in *mppref10* are less impressive with only

| Query Set | Model | $mAPpref$ | $mppref10$ |
|---|---|---|---|
| Full | Baseline | 0.6569 | 0.6402 |
| | $P_{dir2}$ | 0.6644 | 0.6347 |
| | $P_{mix}$ | 0.6627 | **0.6481** |
| | $P_{ll}$ | **0.6667** | 0.6432 |
| | $s(Q, M)$ | 0.6625 | 0.6414 |
| *name* | Baseline | 0.6813 | 0.6640 |
| | $P_{dir2}$ | 0.6797 | 0.6545 |
| | $P_{mix}$ | **0.7023** | **0.6886** |
| | $P_{ll}$ | 0.6906 | 0.6880 |
| | $s(Q, M)$ | 0.6814 | 0.6038 |
| *name*+ | Baseline | 0.6318 | 0.6155 |
| | $P_{dir2}$ | **0.6490** | **0.6359** |
| | $P_{mix}$ | 0.6144 | 0.6178 |
| | $P_{ll}$ | 0.6452 | 0.6090 |
| | $s(Q, M)$ | 0.6383 | 0.6037 |

**Table 6.4:** Cross-Validation performance of integrating author evidence. Observed differences are not a significant improvement over the baseline.

| Query Set | Model | $mAPpref$ | $mppref10$ |
|---|---|---|---|
| *name*+ $\geq 8$ terms | Baseline | 0.6240 | 0.5965 |
| | $P_{dir2}$ | **0.6815**$^*$ | **0.6092** |
| | $P_{mix}$ | 0.5826 | 0.5570 |
| | $P_{ll}$ | 0.6342 | 0.5782 |
| | $s(Q, M)$ | 0.5979 | 0.5678 |

**Table 6.5:** Cross-Validation performance of integrating author evidence, longest *name*+ queries only. Significant improvements over the baseline ($p \leq 0.05$) indicated with $^*$.

a 2% improvement. This is an indication that the improvements in *mAPpref* are not due to improved ranking high in the ranked list, but rather further down the ranking. When looking at precision at deeper rank cutoffs, we see a much larger and significant performance gain, with a 17.7% gain in *ppref25* and a 19.1% gain in *ppref50*, both significant at the $p < 0.05$ level. It is not clear whether early precision or higher recall are more important features for this task.

### 6.3.3 Discussion

In this section, we presented and applied several methods for integrating content generated by authors from throughout an online forum within thread ranking models. While these models fail to produce a significant improvement in general, the smoothing model yields almost a 10% improvement in $mAPpref$ performance on the longest queries in our dataset. There are several conclusions we can draw from these results.

First, the models $P_{mix}, P_{ll}$ and $s$ seem to be more susceptible to over-fitting with the smaller query set used here — all three of these models achieve a lower test-set performance than the baseline on one

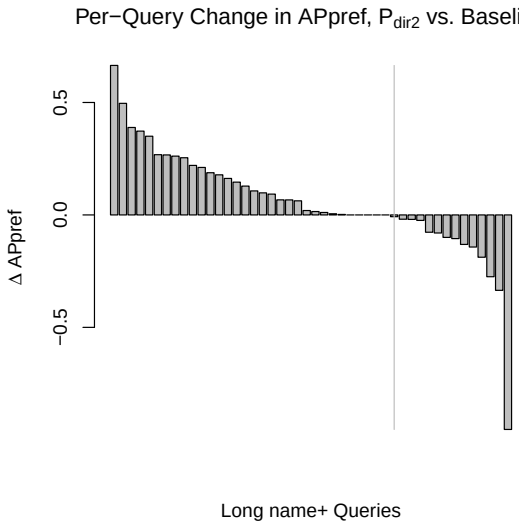



**Figure 6.3:** Per-Query $APpref$ test set performance difference for baseline vs. $P_{dir2}$, longest *name+* queries only. The mean improvement of 9.2% is significant at the $p < 0.05$ level, with one query dramatically hurt. See Section 6.3.3 for a discussion.

or both of the evaluation measures reported. This is particularly interesting because these models all have only a single tuning parameter, $\lambda$ in the probabilistic models or $\alpha$ in the score combination model, whereas the smoothing model has two parameters, $\mu_C$ and $\mu_D$. Even though the smoothing model has a higher degree of flexibility, we see better generalization performance.

Second, in Figure 6.3 showing per-query performance delta, we see that one query is dramatically hurt when applying the smoothing model — in fact, its $APpref$ performance dropped from 0.9540 to 0.0000 with the application of the smoothing model. Upon closer inspection, this query (*[wayne tennant kansas city wyandotte kansas canada english]*) has very few documents marked as preferred, and none of those documents contain the name portion *[wayne tennant]*. Due to our query optimization described above, avoiding scoring messages without mentions of the name, we do not retrieve any of those documents when applying the smoothing model, thus the $APpref$ performance of 0.000. Despite the drop in performance on this query, we still achieve a substantial and statistically significant performance improvement on average across these long queries, and we believe that further improvements could be realized if this query optimization were unnecessary.

Finally, we note that the queries helped most are in fact the most difficult queries in the test collection. They not only have a lower baseline performance than on any other query set considered, the longer queries tend to have more documents marked *BAD* by the assessors. Pearson's correlation between the query length and number of *BAD* documents is 0.38, and the longest queries with $\geq 8$ terms have on average 28.7 *BAD* documents whereas the query set on average has 15.2 *BAD* documents.

## 6.4 Integrating Subforum Content

We can integrate subforum content into the message scoring algorithm in a similar manner as author content. Both the *within-model* and *model agnostic* methods can be applied to smooth the post language model with the containing subforum language model, or to combine subforum- and post-scores before

aggregating into a message score.

Table 6.6 shows the application of these models for integrating subforum content into the message scoring. In this table, we see very slight differences in cross-validation performance in these results. None

| Query Set | Model | $mAPpref$ | $mppref10$ |
|---|---|---|---|
| Full | Baseline | 0.6569 | **0.6402** |
| | $P_{dir2}$ | **0.6600** | 0.6158 |
| | $P_{mix}$ | 0.6540 | 0.6275 |
| | $P_{ll}$ | 0.6568 | **0.6402** |
| | $s(Q, M)$ | 0.6555 | 0.6360 |
| *name* | Baseline | 0.6813 | 0.6640 |
| | $P_{dir2}$ | 0.6754 | 0.6038 |
| | $P_{mix}$ | 0.6780 | 0.6500 |
| | $P_{ll}$ | 0.6751 | 0.6642 |
| | $s(Q, M)$ | **0.6833** | **0.6673** |
| *name+* | Baseline | 0.6318 | **0.6155** |
| | $P_{dir2}$ | **0.6366** | 0.5841 |
| | $P_{mix}$ | 0.6316 | 0.5910 |
| | $P_{ll}$ | 0.6316 | 0.6154 |
| | $s(Q, M)$ | 0.6141 | 0.5817 |

**Table 6.6:** Cross-Validation performance of integrating subforum evidence. Observed differences are not significantly different than the baseline.

of the observed differences are significant.

Many of the subforums are quite small, and it is possible that the lack of performance improvement may be due to the sparsity of the content in some subforums. In order to test this hypothesis, we run another series of experiments aggregating scores across some forums up to a shared parent node. As described in Chapter 3, the Ancestry.com subforum organization has two primary components — surnames and locations. There is no obvious method to collapse the surname subforums to a conceptually shared parent node. The location subforums, on the other hand, are naturally hierarchical. In order to incorporate subforum information from shared parent subforums, we re-calculate the subforum query likelihoods as a mixture model:

$$P'(Q|S) = wP(Q|S) + (1-w) * P(Q|par(S)) \tag{6.9}$$

$$= wP(Q|S) + (1-w)\frac{1}{|child(par(S))|}\sum_{R \in child(par(S))} P(Q|R) \tag{6.10}$$

where $par(S)$ gives the parent node of $S$ and $child(X)$ is a set of all the children nodes of $X$. For the purposes of this experiment, we only recalculate the subforums scores for those subforums under the `localities` root node, and always resolve $par(S)$ to the country-level node. For example, specific location subforums such as `Localities > North America > Canada > British Columbia > Kootenay` are combined with the scores of all the subforums under the `Localities > North America > Canada` node.

This method of score aggregation across child subforums is an expedient approach, building a mixture model with the previously generated query scores. But, this approach still uses the query scores calculated

at the potentially sparse subforum level $P(Q|R)$. Alternatively, we may consider aggregating information across subforums via a large-document model or other aggregation method that combines evidence at the term- rather than subforum-level. We leave this exploration for future work.

Given the re-estimated subforum query likelihood, $P'(Q|S)$, we can apply the *model agnostic* methods above. Table 6.7 shows the results fo these experiments. In these results, we see a similar trend, with very

| Query Set | Model | $mAPpref$ | $mppref10$ |
|---|---|---|---|
| Full | Baseline | **0.6569** | **0.6402** |
| | $P_{mix}$ | 0.6543 | 0.6206 |
| | $P_{ll}$ | 0.6568 | **0.6402** |
| | $s(Q, M)$ | 0.6554 | 0.6360 |
| *name* | Baseline | 0.6813 | 0.6640 |
| | $P_{mix}$ | 0.6785 | 0.6521 |
| | $P_{ll}$ | 0.6750 | 0.6642 |
| | $s(Q, M)$ | **0.6833** | **0.6672** |
| *name+* | Baseline | **0.6318** | **0.6155** |
| | $P_{mix}$ | 0.6226 | 0.5902 |
| | $P_{ll}$ | 0.6317 | 0.6154 |
| | $s(Q, M)$ | 0.6123 | 0.5946 |

**Table 6.7:** Cross-Validation performance of integrating subforum evidence, collapsed `Localities` subforums. Observed differences are not a significantly different than the baseline.

little change in performance compared to the baseline.

## 6.5 Conclusion

In this chapter, we focus on evaluating methods of ranking message threads, as well as integrating content from throughout the online forum collection when ranking forum threads. When looking at methods to aggregate message scores into thread scores, we see a substantial and significant difference between inclusive models that use content from all messages from the thread and selective models that discard some thread content when ranking. The $PCS$ model that scores threads based on the top-$k$ scoring messages is the clearly superior model, benefitting both the short *name* queries and long *name+* queries, as well as being highly effective for the variety of input systems used in the original document pool. These observed improvements are substantial with roughly a 20% improvement over a large-document baseline, and significant at the $p < 0.01$ level.

When evaluating whether there is a benefit to incorporating content from throughout the collection into the thread ranking, the results are not so clean-cut. We evaluate two general techniques for integrating this content: *within-model* methods, incorporating extra-message content via two-stage smoothing methods, and *model agnostic* methods that combine message-scores with author- or subforum-scores. For the majority of cases, incorporation of either author or subforum content with either method do not improve retrieval performance. But, in the case of the longest queries in the collection, those with at least eight query terms, smoothing with the author language model does produce a substantial improvement of almost 10%, significant at the $p < 0.05$ level. These queries helped are the most difficult queries in the test collection, with more documents marked *BAD* on average, and a considerably lower baseline model

performance.

Although these are strong positive results, an open question remains whether the results of smoothing with the author language model generalize to other online forum collections. While the Ancestry.com Forum is an interesting collection, the general topic of discussion is unique. We observed a performance improvement on extremely long queries, which are not the norm in general web search environments. These queries made up a substantial portion of the query set used for this evaluation, and indicate the information needs of the users of Ancestry.com's current search system are particularly complex and multifaceted. Further research into thread retrieval using other online forum collections is needed to understand the types of information needs typical to those collections and validate the results observed here.

One finding in this line of work is that current retrieval systems, while quite flexible for experimenting with the use of within-document structure, do not afford the same level of support for evaluating collection-level structure. For the experiments conducted here, at least three different indexes of a forum collection must be created — defining documents at the message level, author level and subforum level. The documents defined for the author- and subforum-indexes contain extents representing each message belonging to that author or subforum. These indexes essentially cast the problem of non-hierarchical collection structure into that of hierarchical, nested document structure. Because of the strictly hierarchical nature of the resulting collections, it is not possible to investigate models that may, for example, smooth the message language model with the subforum and author language models simultaneously. Additionally, the extremely large documents resulting from this restructuring of the collection have detrimental effect on extent retrieval performance. When retrieving message extents from within these documents, often nearly all the messages in the collection are scored for each query. One major future research direction is to investigate enhancements to current retrieval systems that would allow faster and more flexible architectures for experimenting with collection structure.

# Chapter 7

# Conclusion

In this chapter, we summarize the work in this thesis, present the major research contributions, and provide directions for future work.

## 7.1 Summary of Experiments and Results

This thesis presents an exploration of search in conversational media, with a specific focus on the use of collection structure in search algorithms. We approach this exploration by investigating three search tasks in two different types of social media collections.

### 7.1.1 Blog Search

Chapters 4 and 5 present work on two search tasks in blog collections. Blog collections are challenging and interesting for several reasons. First, these collections tend to be noisy, containing spam blogs and potentially extraneous content such as boilerplate and comments. These aspects of the collection may necessitate special techniques for identifying high-quality blogs and extracting useful text from blog posts. Second, a blog post collection contains information about which blogger authored each post. This structural information enables deeper analysis of author expertise, integration of other blog-level features into a post ranking algorithm, and multiple levels of retrieval granularity. Many of these avenues are explored in the work presented in this thesis, and we give an overview of the work in blog search below.

#### Post Search

The first of these tasks, blog post search, is discussed in Chapter 4. This is the task of identifying blog posts relevant to a user's query. In the context of TREC, this task was studied in conjunction with opinion detection and polarity classification [82, 100, 102], however we evaluate our work only with respect to topical relevance and not opinion or polarity.

Our approach to blog post search focuses on leveraging blog-level evidence in the post ranking, and particularly methods to include language of the blog as a whole when scoring posts. We first demonstrate the utility of blog-level features by building an oracle model that is capable of re-weighting blog posts based on the blog they belong to. Given perfect knowledge of relevance judgements, this oracle model is able to achieve roughly an 85% improvement in performance over the baseline by combining the post retrieval score with a constant blog weight for each blog. Analysis of the blog weights learned in that model show a general trend towards down-weighting blogs with more retrieved posts, with a handful of blogs that generate significantly more relevant posts receiving moderate positive weights.

The oracle model results show that blog-level evidence is capable of improving post retrieval performance, and given that knowledge we turn our attention to methods of integrating the language of the blog as a whole into the post scoring. We evaluate two primary methods to accomplish this. Both of these methods are motivated by the hypothesis that a high match between the language of the blog as a whole and the query may indicate a the blogger is an *expert blogger*, one who has written a significant number of relevant posts. We hypothesize that by using the language of the blog as a whole in post scoring, we can favor those expert bloggers, thus improving retrieval performance. The two models we evaluated are:

1. A language modeling approach that smooths the post language model with the language model of the blog as a whole. This method can be interpreted as similar to document expansion, introducing content from other posts written by the blogger when scoring a post. This model is only capable of incorporating *positive* evidence in the scoring function, and combines evidence at the *term-level*.
2. A score combination approach that linearly combines the blog post retrieval score with a score derived from the blog as a whole. In contrast to the smoothing method, this method is capable of incorporating *negative* evidence, and combines evidence at the *document-* or *post-level*.

While we find that the retrieval score over bloggers is in fact an effective method to identify expert bloggers, the inclusion of a positive weight on the blog-level evidence in post scoring consistently degrades post retrieval performance. The score-combination model, however, does significantly improve performance by 6.5% when including the blog-level evidence as a negative feature. This finding is contradictory to our original motivation for the work, and further investigation led to several significant research contributions.

The first contribution is an exploration of the efficacy of a two-part model, which first classifies bloggers as expert or non-expert then differentially weights evidence from those two classes of bloggers when ranking posts. We demonstrate that accurate expert blogger classification can lead to further performance improvements, up to 12%, by allowing the retrieval model to dis-favor non-experts and favor experts. The classification models applied in this work fail to achieve sufficient accuracy to improve post retrieval performance, and the refinement of these models remains as an interesting and fruitful avenue for future work.

The second contribution is an examination of why a negative weight on the blog-level evidence leads to a positive improvement in post retrieval performance. We identify several reasons why high-scoring blogs are unlikely to contain many relevant posts, including query matches in the blog boilerplate content, and word sense mismatches between the query and dominant word sense used in the blog. Most of these reasons highlight the risk-reward tradeoff of favoring some blogs over others when ranking posts — by favoring more blog-level evidence, the diversity of the resulting post ranking is decreased. We find that, while some methods to eliminate boilerplate content or spam blogs will likely help retrieval performance, they cannot address all of the query-mismatch types identified. The technique applied here, using the blog-level evidence as a negative feature in a score combination method, is a lightweight and seemingly effective method to balance this risk-reward tradeoff without the need to perform expensive template analysis or spam classification.

### Feed Search

Chapter 5 explores the task of blog feed search, that of identifying blog feeds likely to contain an ongoing interest in the topic of the query. This is a task of ranking entire blogs, rather than blog posts as described above. In TREC, the feed search task is described as identifying blogs that have a "central and recurring interest" in the topic of the query [82, 102].

Our approach to blog feed search focuses on adapting models designed for federated search to this

task. We evaluate several probabilistic retrieval models for feed ranking, different methods to represent feeds. The retrieval models explored for this task are:

1. A "Large Document" model that views the feed as a single monolithic document. This model represents a blog feed as a concatenation of the corresponding blog posts, and applies straightforward language modeling retrieval techniques to the task of feed ranking. While this model is somewhat inflexible, unable to re-weight or select blog posts in the scoring, it is generally effective and previous research in blog feed search has found it to be a superior approach.
2. A series of "Small Document" models that view the feed as a collection of documents, the blog posts. These models score blog posts individually and aggregate those scores into a blog feed score. The small document models presented are inherently more flexible than the large document model, capable of selecting or re-weighting blog posts when scoring. These models are also appealing from a practical point of view, allowing the retrieval system to index blog posts individually, rather than continually re-index the entire blog text when new posts are discovered. We also consider several effective enhancements to the small document model, developing a "centrality" measure to favor blog posts more similar to the rest of the blog and adjusting for blog feed size.

When comparing the models for blog feed search, we present several interesting findings. First, we find that the large document model performs considerably better than a straightforward implementation of the small document model that does no feed length adjustment or post re-weighting. This finding is in keeping with most other work in blog feed search. But, when adjusting for feed length or re-weighting blog posts with the centrality measure, the small document model's performance becomes comparable to that of the large document model. The experiments show roughly a 4.5-5.5% improvement in the small document model due to the post re-weighting and a 7.5-8.5% improvement due to the feed length adjustment. Additionally, when applying these two enhancements together, the small document model's performance further improves and achieves significant improvements of roughly 8.5% over the baseline large document model.

### 7.1.2 Online Forum Search Tasks

In Chapter 6 we present a series of experiments looking at message thread search in online forums. Online forums provide a particularly rich level of collection structure, with messages organized into threads, which are then organized into topically focused subforums. Messages are written by authors who may have contributed other messages to threads throughout the collection. Our focus is on techniques to leverage this collection structure in ranking algorithms.

Our first set of experiments evaluate the efficacy of different thread retrieval models, again adapting approaches originally proposed for federated search and other document aggregate ranking tasks. We evaluate several models, including the large and small document models also studied with respect to blog feed search. We find that the small document models typically outperform the large document model for this task. In particular, *selective* models that discard some of the messages in the thread tend to work best. The best performing model across two collections and several evaluation metrics is the Pseudo-Cluster Selection method which scores message threads based on the top-$k$ scoring messages within the thread.

Our second set of experiments looks at integrating information from messages an author has written throughout the collection in the thread ranking. We evaluate several methods for introducing this information, similar to our approach in blog post search. These methods include within-model use of author evidence, such as document expansion or smoothing methods that include author evidence at the term-level, and model-agnostic methods that include author evidence at the message-score level. We find that for the longest queries in our dataset, those that represent multi-faceted information needs, we do achieve

a substantial and significant performance improvement of almost 10%.

Finally, we evaluate methods for integrating subforum information into the thread ranking. We evaluate similar techniques to those used to integrate author information, but fail to find any reliable or significant improvement.

## 7.2 Research Contributions

As social and structured document collections become more important online, the techniques for providing access to these collections must evolve. This thesis presents an exploration of search in several types of social collections, with several contributions towards that goal. These include novel and significant experimental results, explorations of new problem areas in information retrieval, and the release of a dataset. This section outlines those major contributions, with pointers to the specific chapters where we detail those contributions.

### 7.2.1 Datasets and Evaluation

In Chapter 3 we describe datasets built for the purpose of studying thread search in online forum collections. This chapter contains several research contributions. First, and most importantly, we present the Ancestry.com Forum test collection. This forum document collection is the largest IR test collection studied for the task of thread search in online forums, with many times more messages, threads, subforums and queries than previously studied collections. The collection will be released to the IR research community[1], allowing other researchers to build upon our results and explore other methods of improving thread search. In addition to investigating IR in online forums, this collection provides opportunities to study other tasks such as message-response predictions or investigation of graph-based authority measures such as "expertise networks" [137]. The availability of a large conversational social media document collection, queries and relevance judgements provide considerable utility to the IR research community going forward.

In addition to the Ancestry.com Forum test collection, we also present a novel method for building a retrieval test collection using interactions among the participants in the collection to indicate document relevance. Thus type of *in situ* relevance information, while sparse, does have several benefits over traditional test collections. For example, the relevance information is provided by participants in the discussion rather than third-party judges. We apply this technique to build the MacRumors.com Forum test collection.

Finally, we present an algorithm for collecting pairwise preferences to build a test collection, providing a software implementation and tool to collect document pair preferences via a web interface. We use this software to collect pairwise preference judgements for the Ancestry.com Forum test collection.

### 7.2.2 Experimental Results

This thesis presents novel and significant experimental results in three different retrieval tasks. In this section, we describe the research contributions to each of those tasks.

#### Online Forum Thread Search

In the area of thread search in online forums, our contributions are in the area of retrieval models for thread search, and methods to integrate co-authored content into thread ranking. These contributions are

[1] Details of dataset distribution are pending.

described in detail in Chapter 6. In our exploration of thread retrieval models, our major finding is that selective retrieval models that discard some of the messages in the thread outperform those that use all the messages in the thread. This finding is consistent across evaluation metrics, datasets, and differing query lengths.

We also show that, for the longest queries in the Ancestry.com Forum test collection, the expansion or smoothing of the message language model with the language model of the corresponding author improves retrieval performance. Although we have yet to validate these results on another collection, they are promising indications that the use of an author language model can improve retrieval in cases where the information need is complex and multifaceted.

#### Blog Post Search

Chapter 4 presents experiments in blog post search task, where we explore using the content of the blog as a whole in post scoring. We find that leveraging blog-level features in post scoring poses a risk-reward tradeoff. By incorporating more information from the language of the blog into the post scoring model, we bias the results towards a small number of prolific blogs which may have a differing interpretation of the query. On the other hand, by favoring posts from a diverse set of blogs, we can hedge against this situation. The work in this thesis presents the first demonstration that diversity can play a significant role in blog post search.

#### Blog Feed Search

Our work in Chapter 5 explores the task of blog feed search, evaluating the efficacy of large- and small-document retrieval models for this task. While previous work on blog feed search has shown a superiority of large-document models that represent a blog feed as a single document, our work demonstrates that an appropriately formulated small-document model is capable of achieving significantly better performance. The effectiveness of the small document model hinges on weighting blog posts by a novel measure of topical centrality between a blog post and the feed, as well as effectively controlling for feed size.

## 7.3 Generalizability of Results and Recommendations

This thesis took a broad view of search in social media, with several different tasks over different collections. Our focus was on evaluating the potential to improve search by using structural elements in social media collections, such as message threading or co-authorship information. Although we applied a variety of techniques several techniques have emerged as generally useful for collections such as these. On the other hand, several techniques take advantage of specific characteristics of these collections and may not broadly applicable across all conversational social media. In this section, we summarize the generalizability of our results, and give broad recommendations for search in conversational social media.

Throughout these experiments, we have found that small-document models work well for document aggregate ranking tasks in social media collections. Previous work has found that small-document representations are superior for some tasks such as resource ranking for federated search [114] and large-document representations are superior for other tasks such as blog feed search [15, 110]. The work here shows that aspects of the small document models must be tailored to the task, such as effective re-weighting of document aggregate elements and controlling for aggregate size. Additionally, small document models are practical in real-world settings, allowing incremental indexing of new documents rather than re-indexing of large-document representations as new content is added.

Along these lines, we also find that selection and re-weighting of elements within a document aggregate can be useful techniques. In the blog feed search task, we found that favoring documents "central" to the overall topic of the blog improved performance. In thread search, we found that scoring threads based on a subset of messages consistently outperformed models which used all the messages in a thread.

Integrating co-authored content in document ranking has potential to improve retrieval performance in several ways, although the findings presented here may not be broadly applicable across other collections or tasks. For the blog post search task, our results show that a high degree of query match at the blog-level is an indication of a match in the boilerplate content rather than in the post content. We present a technique to diversify the ranked list and hedge against query mismatches at the blog-level. However, the application of corpus preprocessing techniques to remove this boilerplate content will likely dramatically alter the blog post results reported here. Our findings may not apply to collections which have this content removed, or other retrieval techniques designed to exclude boilerplate content.

For thread search in the Ancestry.com Forum, we find that long and multifaceted queries can be helped by expanding messages with other content an author has written. These queries are the longest in the dataset, contain at least eight tokens, and are significantly longer than queries typically seen in web search engines. The rich annotation present in other datasets accessible on Ancestry.com (eg. census and military records, family trees) necessitates a highly structured query input form to access the data along multiple dimensions. Other online forum sites may not have a need to provide such a rich query interface, and visitors to other forum sites may not express such complex information needs. Further query log analysis across a variety of online forum websites is necessary to understand whether this type of information need is unique to Ancestry.com or whether it may be more widespread.

Finally, The techniques used to build the MacRumors test collection leveraged the interaction between participants and linking within the discussion to identify queries and relevant documents. As mentioned in Chapter 3, this contributors to this collection are particularly tech-savvy and the online forum software supports easily formatting links within a message. Popular forum software such as vBulletin[2] and phpBB[3] do support this functionality. The Ancestry.com collection, on the other hand, has virtually no hyperlinking within the collection, and the forum software has no support for formatting hyperlinks in a message. It is difficult to estimate the prevalence of within-forum linking in general, however it does seem to be present to some degree in largely technical discussion boards, for example on the topic of video games[4] or linux software[5]. Additionally, several popular collaborative question answering (CQA) sites[6], which share some similarities to online forums, allow for explicitly marking questions as duplicates to previously answered questions. This functionality essentially provides the same type of interaction we leveraged to build our test collection, although with structured metadata rather than in the message text. While the techniques used to build the MacRumors Forum collection cannot be applied in general to all online forum sites, the same within-forum linking and interactions among participants are present in at least some other collections. Further analysis of forums and CQA sites is necessary to understand how prevalent this type of interaction is.

[2] https://www.vbulletin.com/
[3] http://www.phpbb.com/
[4] http://boards.ign.com/
[5] http://ubuntuforums.org/
[6] Stack Overflow http://stackoverflow.com/, Quora http://www.quora.com/

## 7.4 Future Work and Extensions

As a relatively new area, search within social collections still holds many interesting challenges. Many of the avenues explored in this thesis just begin to scratch the surface in creating information systems to improve access to these types of data. In this section, we present several avenues that are particularly ripe for future research, building on the work here.

### 7.4.1 Further Exploration of Author Expertise

One motivation for much of the work in this thesis is to explore the role of expertise in document ranking. This motivation led to the experiments in blog post search (Chapter 4) where we present a model of post ranking that differentially weights the contribution of a blogger based on their expertise. In that case, expertise is defined by the fraction of relevant post written by that blogger in the retrieved documents. Given perfect knowledge of this expertise, substantial performance gains could be realized in post retrieval performance. In our work on thread search (Chapter 6), we make the assumption that an author's expertise would be reflected in a language model built from the message written by that author. We show that smoothing the message language model with the author's language model can significantly improve performance on long and multi-faceted queries.

These experiments show that retrieval performance can be improved in some cases by including content from other documents an author has written. However, we believe that these results only scratch the surface of using authorship information in search. There are several areas of study that could build on this work, including many additional sources of evidence to use in identifying author expertise and alternate definitions of expertise that may be appropriate.

#### Alternate Definitions of Expertise

In our work, we used the fraction of relevant posts written to identify query-specific expert bloggers. This may not the the most appropriate definition of expertise in all situations, and alternate definitions should be considered. Rather than looking at expertise as a binary classification, graded expertise levels may be useful. In this case, rather than predict a single "expert" class, one could predict a scale of expertise, and differently influence the resulting rank of posts written by more or less expert authors. Alternatively, one could also directly collect expertise labels from judges. This avoids the need to rely on the relevance judgements in the collection to identify experts, and is more along the lines of the approach to expert search at the TREC Enterprise Track [16].

#### Features for Identifying Expertise

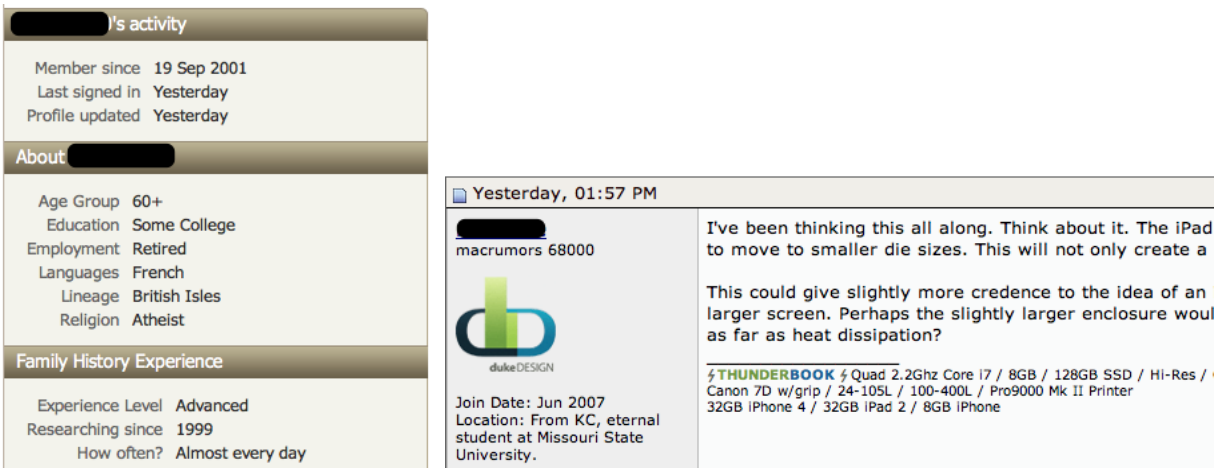
We presented one method to classify blog post authors as experts using a feature set derived primarily from the content of the blog posts and various counts in Section 4.4.2. This method did not achieve sufficient accuracy to improve retrieval performance, and a different feature set may produce more accurate results. Some alternate features that may have potential in identifying experts are described in this section.

Multifaceted topical modeling of author expertise could yield a more nuanced view of an author's expertise, and features derived from multiple topical facets could potentially help identify experts. An author may have written documents on several different topics, and teasing apart their level of expertise in each of those topics could lead to more nuanced and effective retrieval models. One method to do this is via Author-Topic Modeling [118], a variant of Latent Dirichlet Allocation (LDA) [22]. LDA is an unsupervised technique for learning a generative model of text, identifying a number of latent "topics" in a collection and associations between those topics and the documents in the collection. The Author-Topic

Model is an extension to LDA which takes into account the topical similarity of documents written by the same author, and additionally identifies associations between authors and topics. The association of an author to a mixture of topics may be a useful technique for identifying a more rich profile of author expertise.

Previous work in online forums and CQA datasets has represented authors in a conversational collection as a graph, with authors represented by nodes and responses creating edges [135, 62]. That work investigates random-walk algorithms such as PageRank [23] to identify authoritative or expert authors. These graph-based authority measures also have the potential to be effective for identifying expert bloggers. Preliminary work not presented in this thesis on the MacRumors.com and Ancestry.com forums shows these types of expertise measures may also be beneficial features in thread ranking. Those preliminary results are inconclusive, but do indicate there is potential for features derived from an author-graph to provide additional evidence in thread ranking models.

Many conversational social media collections allow authors to create a personal profile which could contain additional, possibly structured information. On online forums, for example, this information often appears in an authors "signature" in each message they post. Figure 7.1 shows two examples of user profile information from the two online forums presented in Chapter 3. This type of semi-structured data




(a) Ancestry.com Forum Profile (b) MacRumors.com Forum Profile and Signature

**Figure 7.1:** Online forum user profiles from two forums. The Ancestry.com Forum profile (7.1(a)) shows several structured fields that may indicate expertise, such as "Education" and "Experience Level". The MacRumors.com Forum profile (left) and signature (bottom) (7.1(b)) gives the author's tenure on the forum, location and types of hardware they may have expertise in.

often lists specific areas an author may be expert in as well as the amount of time they have been active in the forum. These features provide additional dimensions along which to identify experts, complementary to the content-based features used in this thesis.

Finally, it may be beneficial to identify other dimensions of author quality, especially with respect to other authors in the collection. For example, features such as the relative reading level of the posts on a blog or the post length compared to the averages in the collection could be an indication that a blogger writes higher quality content and is more likely to be considered an expert by others.

### 7.4.2 Cross-Site Author Normalization

Figure 3.8 shows a skewed, likely Zipfian distribution of messages per author, indicating most authors have written very few messages. For this reason, it may be necessary to identify additional sources of evidence to accurately estimate quality or expertise measures over authors. This additional evidence could be found in other online forum sites, but necessitates identifying which user accounts on different sites belong to the same author.

For many topics, there are multiple online forum sites with rich discussion and expert contributors. For example, both TastyBrew[7] and and HomebrewTalk[8] host forums dedicated to home-brewing beer, both MacRumors[9] and MacForums[10] host forums dedicated to Apple, Inc., and both HelpfulGardener[11] and GardenWeb[12] host gardening forums. The same author may have accounts at several forums dedicated to the same topic, or have multiple interests and accounts on message boards of different topics. By identifying those authors and accounts, we could increase the amount of data associated with authors and potentially improve the author quality and expertise estimation.

### 7.4.3 In-Situ Relevance

Building an IR test collection is a challenging, expensive and time-consuming process, requiring the development of information needs reflective of the content in the collection and assessment of documents with respect to those information needs. In Chapter 3 we present a technique for mining information needs and relevant documents from within a conversational collection, identifying instances of a question message and response linking to a potential answer thread. We refer to this type of relevance information as *in-situ relevance*. By using this technique, we mine sparse but usable relevance information and true information needs expressed by authors in the collection. Chapter 6 presents our results with a test collection built in this way, and these results agree with results using a more traditionally built collection.

We found this type of relevance information present in a technology-focused online forum which contains a user base likely to be familiar with hyperlinking. While that type of user base is not necessarily the norm, many other conversational social media sites often explicitly encourage this type of behavior. For example, in collaborative question answer (CQA) websites such as Quora[13] and Stack Overflow[14], facilities are provided for marking a new question as a duplicate of a previously answered question. This behavior is rewarded by some of these systems, as it eliminates duplicate content and isolates discussion of one topic to a single thread.

Mining in-situ relevance information presents several opportunities for future work. The test collection built in Chapter 3 uses only the text of the question message and linked-to thread in a response message. In a collection such as an online forum, much more information about the author of the question is available: messages the author has written, users the author has responded to, and subforums the author has contributed to. These factors may be useful for building a more rich user model for the author, possibly useful in augmenting the query representation. Investigation into query and retrieval models that can represent this type of rich contextual information, and application of these models to structured, conversational collections, is a ripe area of untapped research.

[7] http://www.tastybrew.com/forum/
[8] http://www.homebrewtalk.com/
[9] http://forums.macrumors.com/
[10] http://www.mac-forums.com/forums/
[11] http://www.helpfulgardener.com/phpBB2/
[12] http://forums.gardenweb.com/forums/
[13] http://quora.com
[14] http://stackoverflow.com

### 7.4.4 Tool Support

Throughout our experiments, we attempt to take advantage of relationships between documents and broader document organizations within the collection in retrieval algorithms. The types of structure we look at are primarily hierarchical message-thread-subforum structure and authorship structure. The retrieval models applied attempt to include content of several of these organizational units when scoring documents.

Currently, IR systems available to researchers can support some of these experiments, but only provide the ability to incorporate information from one organizational unit at a time. For example, the Indri search engine provides the facility to smooth a sub-document or passage language model with the containing document's language model. By reformatting our documents to contain all the messages written by a single author, we can smooth message language models with the author's language model during scoring. However, models that may smooth a document's language model across several organizational units are currently not supported. In the case of the online forums studied here, a hierarchical message-thread-subforum organization and an orthogonal author organization are present, as illustrated in Figure 7.2. Current tools are incapable of simultaneously incorporating content from both the message thread and

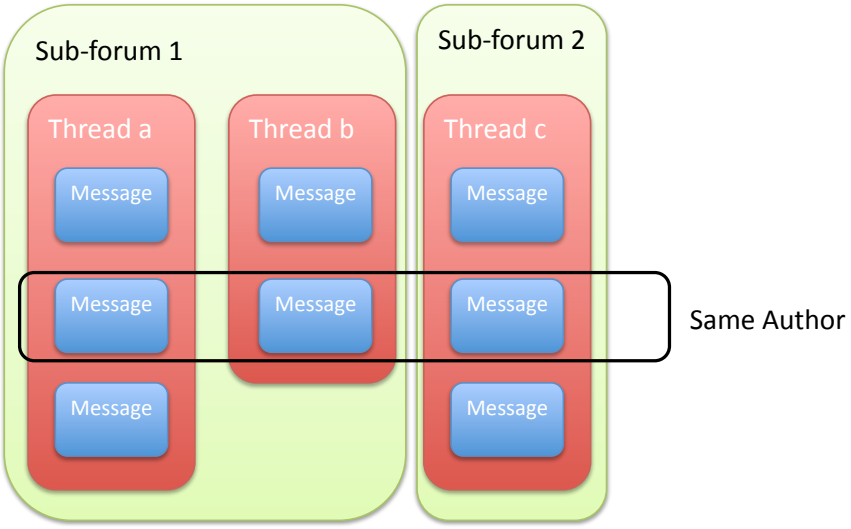



**Figure 7.2:** Online forum organization, showing hierarchical message-thread-subforum relationships and orthogonal authorship organization.

other documents written by the author when scoring a message.

In order to thoroughly investigate collection structure in retrieval models, more flexible scoring techniques must be built into existing search engines. Currently, tools do not support the ability to share content across documents within the same collection. The ability to define subsets of related documents, build language models from those subsets, and incorporate those language models into document scoring would greatly improve the ease of running experiments like those in this thesis, as well as enable more sophisticated retrieval models.

# Bibliography


[1] Internet Usage Over Time Spreadsheet (March 2000 - May 2010). Pew Research Center's Internet & American Life Project. http://www.pewinternet.org/Trend-Data/Usage-Over-Time.aspx, 26 July 2010. (document), 1, 1.1

[2] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 183–194, New York, NY, USA, 2008. ACM. 2.7.4, 3.4.2

[3] Alexa Internet, Inc. Alexa — blogger.com statistics. http://www.alexa.com/siteinfo/blogger.com, 26 July 2011. 1

[4] Alexa Internet, Inc. Alexa — top sites in united states. http://www.alexa.com/topsites/countries/US, 6 June 2011. (document), 1, 1.1

[5] Alexa Internet, Inc. Alexa — tumblr.com statistics. http://www.alexa.com/siteinfo/tumblr.com, 26 July 2011. 1

[6] Alexa Internet, Inc. Alexa — wordpress.com statistics. http://www.alexa.com/siteinfo/wordpress.com, 26 July 2011. 1

[7] Jaime Arguello, Jonathan L. Elsas, Jamie Callan, and Jaime G. Carbonell. Document representation and query expansion models for blog recommendation. In *Proceedings of the 2nd International Conference on Weblogs and Social Media*, ICWSM '08, 2008. 2.7.1, 5.1, 5.2.1, 5.2.2, 5.5

[8] Javed A. Aslam and Mark Montague. Models for metasearch. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 276–284, New York, NY, USA, 2001. ACM. 2.4.3, 3.5.3, 3.5.4

[9] Automattic Inc. Stats — wordpress.com. http://en.wordpress.com/stats/, 10 June 2011. (document), 1.1

[10] Automattic Inc. Traffic — wordpress.com. http://en.wordpress.com/stats/traffic/, 27 July 2011. (document), 1.1

[11] Peter Bailey, Nick Craswell, Arjen P. de Vries, and Ian Soboroff. Overview of the TREC-2007 enterprise track. In *Proceedings of the Fourteenth Text REtrieval Conference*, TREC '07, 2007. 1.1.2, 2.6.3

[12] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 43–50, New York, NY, USA, 2006. ACM. 1.1.1, 2.3, 2.4.2, 2.6.3, 5.2.2, 5.2.2, 6.3.1

[13] Krisztian Balog and Maarten de Rijke. Finding experts and their details in e-mail corpora. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 1035–

1036, New York, NY, USA, 2006. ACM. 2.3

[14] Krisztian Balog and Maarten De Rijke. Associating people and documents. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pages 296–308, Berlin, Heidelberg, 2008. Springer-Verlag. 2.4.2, 2.6.3

[15] Krisztian Balog, Maarten de Rijke, and Wouter Weerkamp. Bloggers as experts: feed distillation using expert retrieval models. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 753–754, New York, NY, USA, 2008. ACM. 2.3, 2.4.2, 2.7.1, 4.3.2, 5.2, 5.4.1, 5.5, 7.3

[16] Krisztian Balog, Ian Soboroff, Paul Thomas, Peter Bailey, Nick Craswell, and Arjen P. de Vries. Overview of the TREC-2008 enterprise track. In *Proceedings of the Fifteenth Text Retrieval Conference*, TREC '08, 2008. 7.4.1

[17] Ron Bekkerman, Andrew McCallum, and Gary Huang. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. Technical Report IR-418, Center of Intelligent Information Retrieval, UMass Amherst, 2004. 2.7.2

[18] Michael Bendersky and W. Bruce Croft. Discovering key concepts in verbose queries. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 491–498, New York, NY, USA, 2008. ACM. 3.4.4

[19] Paul N. Bennett. Personal communication, September 2010. 3.5.4

[20] Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. A few bad votes too many?: towards robust ranking in social media. In *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, AIRWeb '08, pages 53–60, New York, NY, USA, 2008. ACM. 2.7.4

[21] Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. Finding the right facts in the crowd: factoid question answering over social media. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, pages 467–476, New York, NY, USA, 2008. ACM. 2.7.4

[22] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003. 7.4.1

[23] Sergey Brin and Larry Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998. 1, 4.4, 4.5.1, 5.2.2, 7.4.1

[24] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 25–32, New York, NY, USA, 2004. ACM. 4.3.3

[25] Christopher J. C. Burges, Krysta M. Svore, Paul N. Bennett, Andrzej Pastusiak, and Qiang Wu. Learning to rank using an ensemble of lambda-gradient models. In *Journal of Machine Learning Research: Workshop and Conference Proceedings*, volume 14, pages 23–35, 2011. 2.2

[26] James P. Callan, Zhihong Lu, and W. Bruce Croft. Searching distributed collections with inference networks. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, pages 21–28, New York, NY, USA, 1995. ACM. 2.4.1, 2.6.2, 5.2.1

[27] Jamie Callan. Distributed information retrieval. *Advances in Information Retrieval*, pages 127–150, 2000. 2.3, 2.4.1, 2.6.2, 5.2.1

[28] Jamie Callan, W. Bruce Croft, and Stephen M. Harding. The inquery retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83. Springer-Verlag, 1992. 2.1

[29] Ben Carterette and Paul N. Bennett. Evaluation measures for preference judgments. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 685–686, New York, NY, USA, 2008. ACM. 3.5.4, 3.5.7, 3.5.7

[30] Ben Carterette, Paul N. Bennett, and Olivier Chapelle. A test collection of preference judgments. In *SIGIR 2008 Workshop: Beyond Binary Relevance: Preferences, Diversity and Set-Level Judgment*, SIGIR'08, 2008. 3.5.4, 3.5.4

[31] Ben Carterette, Paul N. Bennett, David Maxwell Chickering, and Susan T. Dumais. Here or there: preference judgments for relevance. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pages 16–27, Berlin, Heidelberg, 2008. Springer-Verlag. 3.5.4, 3.5.4, 3.5.5, 3.5.5, 3.5.5

[32] Vitor R. Carvalho and William W. Cohen. Preventing information leaks in email. In *Proceedings of SIAM International Conference on Data Mining (SDM-07)*, Minneapolis, MN, 2007. 2.7.2

[33] Vitor R. Carvalho and William W. Cohen. Ranking users for intelligent message addressing. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pages 321–333, Berlin, Heidelberg, 2008. Springer-Verlag. 2.7.2

[34] William W. Cohen. Learning rules that classify e-mail. In *Papers from the AAAI Spring Symposium on Machine Learning in Information Access*, 1996. 2.7.2

[35] Kevyn B. Collins-Thompson. *Robust Model Estimation Methods for Information Retrieval*. PhD thesis, Carnegie Mellon University, 2008. 3

[36] Gao Cong, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. Finding question-answer pairs from online forums. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 467–474, New York, NY, USA, 2008. ACM. 2.7.3

[37] Gordon V. Cormack. Email spam filtering: A systematic review. *Foundations and Trends in Information Retrieval*, 1:335–455, April 2008. 2.7.2

[38] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009. 4, 3.5.4

[39] Nick Craswell. W3C test collection, April 2005. (document), 2.7.2, 3.6, 6.2.4

[40] Nick Craswell, Arjen P. de Vries, and Ian Soboroff. Overview of the TREC-2005 enterprise track. In *Proceedings of the Fourteenth Text REtrieval Conference*, TREC '05, 2005. 1.1.2, 2.6.3, 2.7.2, 6.2.4

[41] Nick Craswell, Hugo Zaragoza, and Stephen Robertson. Microsoft cambridge at TREC–14: Enterprise track. In *Proceedings of the Fourteenth Text Retrieval Conf*, TREC '05, 2005. 6.2.4

[42] Fernando Diaz. *Autocorrelation and regularization of query-based information retrieval scores*. PhD thesis, University of Massachusetts Amherst, 2008. 2.5, 2.6.1, 4.6.2

[43] S. Ding, G. Cong, C.Y. Lin, and X. Zhu. Using conditional random fields to extract contexts and answers of questions from online forums. In *Proceedings of ACL-08: HLT*. ACL, 2008. 2.7.3

[44] Jonathan L. Elsas, Jaime Arguello, Jamie Callan, and Jaime G. Carbonell. Retrieval and feedback models for blog distillation. In *Proceedings of the Sixteenth Text REtrieval Conference*, TREC '07,

2007. 2.7.1, 5.1, 5.2.1, 5.2.2, 5.5

[45] Jonathan L. Elsas, Jaime Arguello, Jamie Callan, and Jaime G. Carbonell. Retrieval and feedback models for blog feed search. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 347–354, New York, NY, USA, 2008. ACM. 2.3, 2.7.1, 4.3.2, 1, 5.2, 6.1

[46] Jonathan L. Elsas and Jaime G. Carbonell. It pays to be picky: an evaluation of thread retrieval in online forums. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 714–715, New York, NY, USA, 2009. ACM. 1.1.1, 2.3

[47] Jonathan L. Elsas, Vitor R. Carvalho, and Jaime G. Carbonell. Fast learning of document ranking functions with the committee perceptron. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 55–64, New York, NY, USA, 2008. ACM. 4.2.3

[48] Jonathan L. Elsas and Natalie Glance. Shopping for top forums: discovering online discussion for product research. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, pages 23–30, New York, NY, USA, 2010. ACM. (document), 1.1, 1.1.1, 1.1.2

[49] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, June 2008. 4.4.2

[50] Yi Fang, Luo Si, and Aditya P. Mathur. Discriminative models of integrating document evidence and document-candidate associations for expert search. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 683–690, New York, NY, USA, 2010. ACM. 2.4.4, 2.6.3

[51] Donghui Feng, Erin Shaw, Jihie Kim, and Eduard Hovy. An intelligent discussion-bot for answering student queries in threaded discussions. In *Proceedings of the 11th international conference on Intelligent user interfaces*, IUI '06, pages 171–177, New York, NY, USA, 2006. ACM. 2.7.3

[52] Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *Proceedings of the 1993 Text Retrieval Conference*, 1993. 2.4.3, 5.4.1

[53] Steven Garcia. Rmit university at trec 2009: Web track. In *Proceedings of the Eighteenth Text Retrieval Conf*, TREC '09, 2009. 6.2.2

[54] Google. Social search goes global - inside search. `http://insidesearch.blogspot.com/2011/05/social-search-goes-global.html`, 7 June 2011. 1.1

[55] David Hannah, Craig Macdonald, Jei Peng, Ben He, and Iadh Ounis. University of glasgow at TREC 2007: Experiments with blog and enterprise tracks with terrier. In *Proceedings of the Sixteenth Text Retrieval Conference*, TREC '07, 2007. 2.7.1, 5.2.2, 5.4.3

[56] Jiyin He, Wouter Weerkamp, Martha Larson, and Maarten de Rijke. Blogger, stick to your story: modeling topical noise in blogs with coherence measures. In *Proceedings of the second workshop on Analytics for noisy unstructured text data*, AND '08, pages 39–46, New York, NY, USA, 2008. ACM. 5.4.3

[57] Marti A Hearst and Susan T. Dumais. Blogging together: An examination of group blogs. In *Proceedings of the 3rd International AAAI Conference on Weblogs and Social Media*, ICWSM '09, Menlo Park, California, USA, 2009. AAAI. 4

[58] Marti A. Hearst, Matthew Hurst, and Susan T. Dumais. What should blog search look like? In

*Proceeding of the 2008 ACM workshop on Search in social media*, SSM '08, pages 95–98, New York, NY, USA, 2008. ACM. 2.7.1

[59] Thorsten Joachims. Training linear svms in linear time. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2006. 4.2.3

[60] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. *Information Processing and Management*, 36:779–808, November 2000. 2.1, 3.5.3, 6.2.2

[61] Sydney Jones and Susannah Fox. Generations online in 2009. Memo, Pew Internet & American Life Project, http://www.pewinternet.org/PPF/r/275/source/rss/report_display.asp, January 2009. (document)

[62] Pawel Jurczyk and Eugene Agichtein. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 919–922, New York, NY, USA, 2007. ACM. 2.7.4, 7.4.1

[63] Pranam Kolari, Tim Finin, and Anupam Joshi. Svms for the blogosphere: Blog identification and splog detection. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006. 4.6.1

[64] Pranam Kolari, Akshay Java, and Tim Finin. Characterizing the splogosphere. In *Proceedings of the 3rd Annual Workshop on Weblogging Ecosystem: Aggregation, Analysis and Dynamics, 15th World Wid Web Conference*. University of Maryland, Baltimore County, 2006. 4.5.1

[65] Oren Kurland. *Inter-document similarities, language models, and ad hoc information retrieval*. PhD thesis, Cornell University, Ithaca, NY, USA, 2006. Adviser-Lee, Lillian. 4.3

[66] Oren Kurland and Lillian Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 194–201, New York, NY, USA, 2004. ACM. 2.5, 2.6.1, 4.6.2

[67] Amanda Lenhart, Kristen Purcell, Aaron Smith, and Kathryn Zickuhr. Social media and young adults. Memo, Pew Internet & American Life Project, http://www.pewinternet.org/Reports/2010/Social-Media-and-Young-Adults.aspx, June 2010. (document), 1.1

[68] Baoli Li, Yandong Liu, Ashwin Ram, Ernest V. Garcia, and Eugene Agichtein. Exploring question subjectivity prediction in community qa. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 735–736, New York, NY, USA, 2008. ACM. 2.7.4

[69] Yu-Ru Lin, Hari Sundaram, Yun Chi, Junichi Tatemura, and Belle L. Tseng. Detecting splogs via temporal dynamics using self-similarity analysis. *ACM Transactions on the Web*, 2:4:1–4:35, March 2008. 4.6.1

[70] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3:225–331, March 2009. 2.2

[71] Xiaoyong Liu and W. Bruce Croft. Cluster-based retrieval using language models. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 186–193, New York, NY, USA, 2004. ACM. 2.1.2, 2.3, 2.4.2, 2.5, 2.6.1, 4.6.2

[72] Xiaoyong Liu and W. Bruce Croft. Representing clusters for retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 671–672, New York, NY, USA, 2006. ACM. 2.3, 2.4.2, 2.6.1, 5.2.2, 6.1.2

[73] Xiaoyong Liu and W. Bruce Croft. Evaluating text representations for retrieval of the best group of documents. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pages 454–462, Berlin, Heidelberg, 2008. Springer-Verlag. 2.3, 2.4.2, 2.4.2, 5.2.1, 6.1.2, 6.1.2

[74] Yandong Liu and Eugene Agichtein. On the evolution of the yahoo! answers qa community. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 737–738, New York, NY, USA, 2008. ACM. 2.7.4

[75] Yandong Liu, Jiang Bian, and Eugene Agichtein. Predicting information seeker satisfaction in community question answering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 483–490, New York, NY, USA, 2008. ACM. 2.7.4

[76] David E. Losada and Leif Azzopardi. An analysis on document length retrieval trends in language modeling smoothing. *Information Retrieval*, 11:109–138, April 2008. 2.1

[77] Craig Macdonald. *The Voting Model for People Search*. PhD thesis, University of Glasgow, 2009. 2.6.3

[78] Craig Macdonald and Iadh Ounis. The TREC blogs06 collection: Creating and analysing a blog test collection. *Department of Computer Science, University of Glasgow Tech Report TR-2006-224*, 2006. 1.1, 3.1, 4.1, 5.3

[79] Craig Macdonald and Iadh Ounis. Voting for candidates: adapting data fusion techniques for an expert search task. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, CIKM '06, pages 387–396, New York, NY, USA, 2006. ACM. 2.4.3, 2.6.3

[80] Craig Macdonald and Iadh Ounis. Key blog distillation: ranking aggregates. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 1043–1052, New York, NY, USA, 2008. ACM. 2.4.3, 2.7.1, 5.2.2, 5.4.1, 5.4.3, 6.1

[81] Craig Macdonald and Iadh Ounis. Learning models for ranking aggregates. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 517–529, Berlin, Heidelberg, 2011. Springer-Verlag. 2.4.4, 2.6.3

[82] Craig Macdonald, Iadh Ounis, and Ian Soboroff. Overview of the TREC 2007 blog track. In *Proceedings of the Sixteenth Text Retrieval Conference*, TREC '07, 2007. (document), 2.7.1, 2.7.1, 3.1, 3.5.3, 4.1, 4.1, 4.6, 5.1, 5.3, 6.2.2, 7.1.1, 7.1.1

[83] Craig Macdonald, Iadh Ounis, and Ian Soboroff. Is spam an issue for opinionated blog post search? In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 710–711, New York, NY, USA, 2009. ACM. 4.6.1

[84] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. 3.5.7

[85] Donald Metzler and W. Bruce Croft. Combining the language model and inference network approaches to retrieval. *Information Processing and Management*, 40(5):735–750, 2004. 6.3.1

[86] Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In

*Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 472–479, New York, NY, USA, 2005. ACM. 2.1.1, 2.1.1, 3.5.3, 4.2.2, 4.4, 6.2.2

[87] Donald Metzler, Victor Lavrenko, and W. Bruce Croft. Formal multiple-bernoulli models for language modeling. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 540–541, New York, NY, USA, 2004. ACM. 2.1.1

[88] Donald A. Metzler. Automatic feature selection in the markov random field model for information retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 253–262, New York, NY, USA, 2007. ACM. 2.4.4

[89] Microsoft Bing. Facebook friends now fueling faster decisions on bing - search blog. `http://www.bing.com/community/site_blogs/b/search/archive/2011/05/16/news-announcement-may-17.aspx`, June 2011. 1.1

[90] Einat Minkov, William W. Cohen, and Andrew Y. Ng. Contextual search and name disambiguation in email using graphs. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 27–34, New York, NY, USA, 2006. ACM. 2.7.2

[91] Gilad Mishne. Using blog properties to improve retrieval. In *Proceedings of the International Converernce on Weblogs and Social Media*, ICWSM '07, 2007. 1.1.1, 2.7.1, 4.5.1, 4.6.1

[92] Gilad Mishne and Maarten de Rijke. A study of blog search. In *Proceedings of the 28th European conference on Advances in information retrieval*, volume 3936 of *ECIR '06*, page 289. Springer-Verlag, 2006. 2.7.1

[93] Marc A. Najork, Hugo Zaragoza, and Michael J. Taylor. Hits on the web: how does it compare? In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 471–478, New York, NY, USA, 2007. ACM. 1

[94] Sang-Hyob Nam, Seung-Hoon Na, Yeha Lee, and Jong-Hyeok Lee. Diffpost: Filtering non-relevant content based on content difference between two consecutive blog posts. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 791–795, Berlin, Heidelberg, 2009. Springer-Verlag. 2.7.1, 4.5.2, 4.6.1

[95] Paul Ogilvie. *Retrieval using Document Structure and Annotations*. PhD thesis, Carnegie Mellon University, 2010. 2.1, 2.1.2, 2.1.2, 5.2.2, 5.2.2, 6.2.2, 6.3.1

[96] Paul Ogilvie and Jamie Callan. Combining document representations for known-item search. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 143–150, New York, NY, USA, 2003. ACM. 2.7.2

[97] Paul Ogilvie and Jamie Callan. Hierarchical language models for retrieval of xml components. In *Proceedings of the Initiative for the Evaluation of XML Retrieval Workshop*, INEX '04, 2004. 2.1.2

[98] Paul Ogilvie and Jamie Callan. Experiments with language models for known-item finding of e-mail messages. In *Proceedings of the Fourteenth Text Retrieval Conference*, TREC '05, 2005. 6.2.4

[99] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Christina Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006. 3.5.3

[100] Iadh Ounis, Craig Macdonald, Maarten de Rijke, and Gilad Mishne. Overview of the TREC 2006 blog track. In *Proceedings of the Fifteenth Text Retrieval Conference*, TREC '06, 2006. (document), 1.1, 3.1, 3.5.3, 4.1, 4.1, 4.6, 6.2.2, 7.1.1

[101] Iadh Ounis, Craig Macdonald, and Ian Soboroff. On the TREC blog track. In *Proceedings of the International Conference on Weblogs and Social Media*, ICWSM '08, 2008. 2.7.1, 2.7.1, 2.7.1, 4.1

[102] Iadh Ounis, Craig Macdonald, and Ian Soboroff. Overview of the TREC 2008 blog track. In *Proceedings of the Seventeenth Text Retrieval Conference*, TREC '08, 2008. (document), 2.7.1, 2.7.1, 2.7.1, 3.1, 3.5.3, 4.1, 4.1, 4.6, 6.2.2, 7.1.1, 7.1.1

[103] Desislava Petkova and W Bruce Croft. Umass notebook TREC 2006: Enterprise track. In *Proceedings of the Fifteenth Text Retrieval Conference*, TREC '06, 2006. 2.1.2, 2.5, 2.6.3, 2.7.2

[104] 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*, SIGIR '98, pages 275–281, New York, NY, USA, 1998. ACM. 2.1

[105] Quantcast Corporation. tumblr.com — quantcast audience profile. `http://www.quantcast.com/tumblr.com/traffic`, July 2011. (document), 1.1

[106] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Proceedings of the AAAI Workshop on Learning for Text Categorization*, 1998. 2.7.2

[107] Gerald Salton. *Automatic Information Organization and Retrieval*. McGraw-Hill, 1968. 1, 2.6.1

[108] Jangwon Seo, W. Bruce Croft, and David A. Smith. Online community search using conversational structures. *Information Retrieval*, (to appear) 2011. (document), 2.3, 2.7.3, 3.5, 3.6, 6.1.2, 6.2.3, 6.2.4

[109] Jangwon Seo and W. Bruce Croft. Umass at TREC 2007 blog distillation task. In *Proceedings of the Sixteenth Text Retrieval Conf*, TREC '07, 2007. 2.7.1, 5.2, 5.5, 6.1

[110] Jangwon Seo and W. Bruce Croft. Blog site search using resource selection. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 1053–1062, New York, NY, USA, 2008. ACM. 2.3, 2.4.2, 2.7.1, 5.1, 5.2, 5.4.1, 6.1.2, 6.1.2, 7.3

[111] Jangwon Seo and W. Bruce Croft. Local text reuse detection. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 571–578, New York, NY, USA, 2008. ACM. 4.6.1

[112] Jangwon Seo and W. Bruce Croft. Geometric representations for multiple documents. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 251–258, New York, NY, USA, 2010. ACM. 2.4.2, 2.7.2, 6.1.2

[113] Jangwon Seo, W. Bruce Croft, and David A. Smith. Online community search using thread structure. In *Proceeding of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1907–1910, New York, NY, USA, 2009. ACM. 2.4.2, 2.7.3, 6.2.4

[114] Luo Si and Jamie Callan. Relevant document distribution estimation method for resource selection. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 298–305, New York, NY, USA, 2003. ACM. 2.3, 2.3, 2.4.1, 2.6.2, 5.1, 5.2.2, 5.2.2, 7.3

[115] Aaron Smith. 13% of online adults use twitter. Memo, Pew Internet & American Life Project, `http://pewinternet.org/Reports/2011/Twitter-Update-2011.`

aspx, June 2011. (document), 1.1

[116] Marc Smith, Vladimir Barash, Lise Getoor, and Hady W. Lauw. Leveraging social context for searching social media. In *Proceeding of the 2008 ACM workshop on Search in social media*, SSM '08, pages 91–94, New York, NY, USA, 2008. ACM. 1.1

[117] Ian Soboroff, Arjen P. de Vries, and Nick Craswell. Overview of the TREC-2006 enterprise track. In *Proceedings of the Fourteenth Text REtrieval Conference*, TREC '06, 2006. 1.1.2, 2.6.3, 2.7.2, 6.2.4

[118] Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 306–315, New York, NY, USA, 2004. ACM. 7.4.1

[119] Tumblr, Inc. About — tumblr. http://www.tumblr.com/about, June 2011. (document), 1.1

[120] Twitter. Twitter blog: #numbers. http://blog.twitter.com/2011/03/numbers.html, March 2011. 1.1

[121] C J van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann Newton, MA, USA, 1979. 2.6.1

[122] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 315–323, New York, NY, USA, 1998. ACM. 3.5.5

[123] Ellen M. Voorhees. Overview of the TREC-9 question answering track. In *Proceedings of the Ninth Text REtrieval Conference*, TREC-9, 2000. 2.7.4

[124] Ellen M. Voorhees. Overview of the TREC-10 question answering track. In *Proceedings of the Fourteenth Text REtrieval Conference*, TREC '01, 2001. 2.7.4

[125] Ellen M. Voorhees. The philosophy of information retrieval evaluation. In *Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems*, CLEF '01, pages 355–370, London, UK, 2002. Springer-Verlag. 3.4.3, 3.5.3

[126] Wouter Weerkamp, Krisztian Balog, and Maarten Rijke. Using contextual information to improve search in email archives. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 400–411, Berlin, Heidelberg, 2009. Springer-Verlag. 2.7.2

[127] Wouter Weerkamp and Maarten de Rijke. Credibility improves topical blog post retrieval. In *Proceedings of ACL-08: HLT*, pages 923–931, Columbus, Ohio, June 2008. Association for Computational Linguistics. 2.7.1, 4.6.1

[128] Mingfang Wu, Falk Scholer, and Steven Garcia. Rmit university at trec 2008: Enterprise track. In *Proceedings of the Seventeenth Text Retrieval Conf*, TREC '08, 2008. 6.2.2

[129] Wensi Xi, Jesper Lind, and Eric Brill. Learning effective ranking functions for newsgroup search. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 394–401, New York, NY, USA, 2004. ACM. (document), 2.7.2, 3.6, 6.2.4

[130] Xiaobing Xue, Jiwoon Jeon, and W. Bruce Croft. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 475–482, New York, NY, USA, 2008. ACM. 2.7.4

[131] Yahoo!, Inc. Yahoo! answers blog — 1 billion answers served! http://yanswersblog.com/index.php/archives/2010/05/03/1-billion-answers-served/, June 2011. 1.1

[132] Hui Yang, Luo Si, and Jamie Callan. Knowledge transfer and opinion detection in the TREC2006 blog track. In *Proceedings of the Fifteenth Text Retrieval Conference*, TREC '06, 2006. 2.7.1

[133] Yiming Yang, Shinjae Yoo, Frank Lin, and Il-Chul Moon. Personalized email prioritization based on content and social network analysis. *IEEE Intelligent Systems*, 25:12–18, July 2010. 2.7.2

[134] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22:179–214, April 2004. 2.1, 2.1

[135] Jun Zhang, Mark S. Ackerman, and Lada A. Adamic. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 221–230, 2007. 7.4.1

[136] Wei Zhang and Clement Yu. UIC at TREC 2007 blog track. In *Proceedings of the Sixteenth Text Retrieval Conference*, TREC '07, 2007. 2.7.1

[137] Le Zhao and Jamie Callan. A generative retrieval model for structured documents. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 1163–1172, New York, NY, USA, 2008. ACM. 2.1.2, 2.1.2, 4.3, 6.3.1, 7.2.1

[138] Katheryn Zickuhr and Lee Rainie. Wikipedia, past and present. Memo, Pew Internet & American Life Project, http://www.pewinternet.org/Reports/2011/Wikipedia.aspx, June 2011. 1.1

# Ancestry.com Assessment Interface

Figure 3 shows the assessment interface for collecting preference annotation.

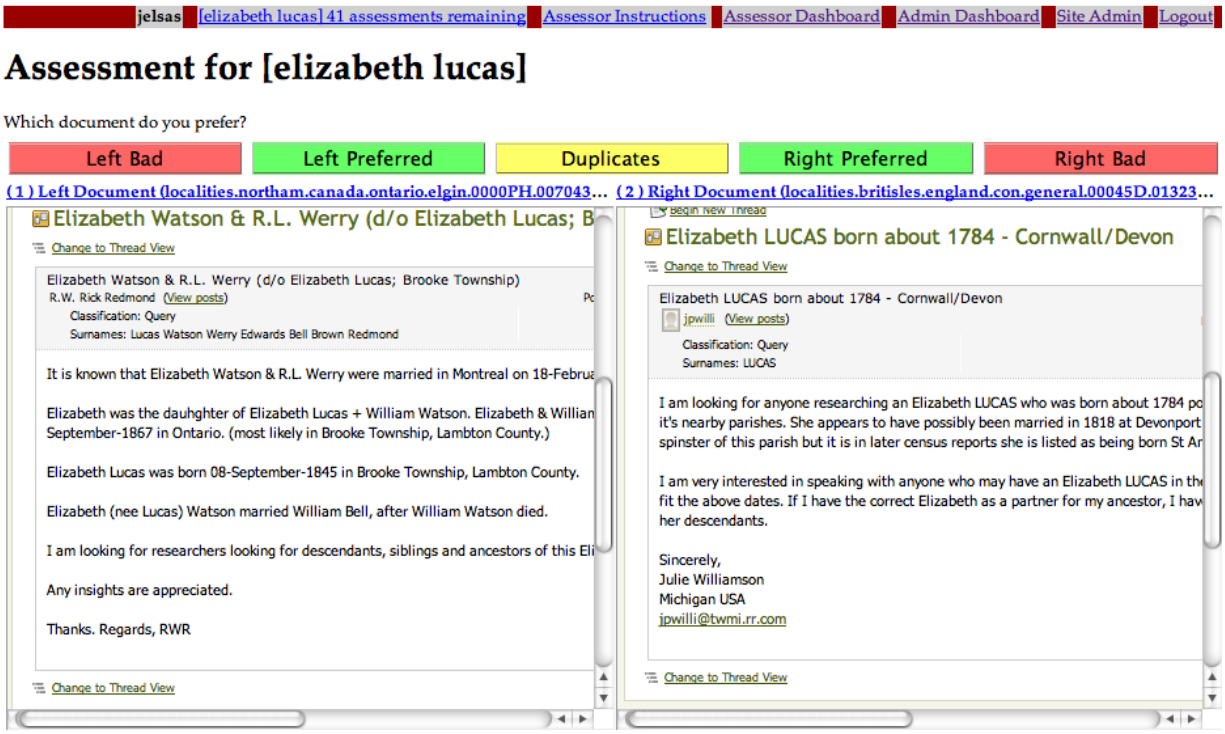



**Figure 3:** Ancestry.com assessment interface

# Ancestry.com Assessment Guidelines

The following assessment guidelines were used as instructions for the Ancestry.com data annotation:

> The user issuing the query is seeking information on the person, people or family named in the query. Some queries contain more detailed information, such as locations, dates, occupations or a race. Your task is to compare two search results for the query, and select the one that is most useful.
>
> Please follow the guidelines below when making decisions on the utility of a document. These guidelines are meant to give some reasons for favoring one document over another, but there may be other reasons one document could be preferred. This is inherently a subjective task, and in some cases there will not be a clear best answer.
>
> - Many queries are ambiguous and could refer to several different people with the same name. Only mark a document Bad if it cannot plausibly refer to the person in the query.
> - If you see an error loading a document, please click the link above the document containing the text "Left Document..." or "Right Document..." to re-load the document. If that does not work, then mark the document Bad.
> - If the two documents are duplicates, eg. have nearly the same text and the same author, then mark them as Duplicates.
> - Documents primarily discussing the person are the most useful, for example an obituary, or detailed description of the person's family.
> - Documents mentioning the person in passing, for example referencing the person as a sibling or spouse, are less useful.
> - Documents containing incidental mentions of the person, for example in a passenger listing, are less useful.
> - Documents with no mention of the person are not useful and should be marked Bad.
> - Documents providing information should be preferred over those that are only requesting information.
> - Documents providing more complete or thorough information should be preferred over those providing less information.
> - Documents by authors who may be an expert in genealogical research or an expert on this person or family should be preferred over those by authors who appear to be novices or are uninformed. For example, some authors may say they have previously researched a person or family, or may say they are new to genealogical research.