Methods for Ranking User-Generated Text Streams

A Case Study in Blog Feed Retrieval

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Mostafa Keikha

under the supervision of
Prof. Fabio Crestani

September 2012

Dissertation Committee

Antonio Carzaniga	Università della Svizzera Italiana, Switzerland
Kai Hormann	Università della Svizzera Italiana, Switzerland
ChengXiang Zhai	University of Illinois at Urbana-Champaign , USA
Ricardo Baeza-Yates	Yahoo Research, Barcelona, Spain
Fabrizio Silvestri	CNR, Pisa, Italy

Dissertation accepted on 14 September 2012

Research Advisor	PhD Program Director
Prof. Fabio Crestani	**Prof. Antonio Carzaniga**

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Mostafa Keikha
Lugano, 14 September 2012

To my beloved family ...

Specially to my brother, Mohammad,
who was my first and best mentor in life.

Abstract

User generated content are one of the main sources of information on the Web nowadays. With the huge amount of this type of data being generated everyday, having an efficient and effective retrieval system is essential. The goal of such a retrieval system is to enable users to search through this data and retrieve documents relevant to their information needs.

Among the different retrieval tasks of user generated content, retrieving and ranking streams is one of the important ones that has verious applications. The goal of this task is to rank streams, as collections of documents with chronological order, in response to a user query. This is different than traditional retrieval tasks where the goal is to rank single documents and temporal properties are less important in the ranking.

In this thesis we investigate the problem of ranking user-generated streams with a case study in blog feed retrieval. Blogs, like all other user generated streams, have specific properties and require new considerations in the retrieval methods. Blog feed retrieval can be defined as retrieving blogs with a recurrent interest in the topic of the given query. We define three different properties of blog feed retrieval each of which introduces new challenges in the ranking task. These properties include: 1) term mismatch in blog retrieval, 2) evolution of topics in blogs and 3) diversity of blog posts. For each of these properties, we investigate its corresponding challenges and propose solutions to overcome those challenges. We further analyze the effect of our solutions on the performance of a retrieval system. We show that taking the new properties into account for developing the retrieval system can help us to improve state of the art retrieval methods. In all the proposed methods, we specifically pay attention to temporal properties that we believe are important information in any type of streams. We show that when combined with content-based information, temporal information can be useful in different situations.

Although we apply our methods to blog feed retrieval, they are mostly general methods that are applicable to similar stream ranking problems like ranking experts or ranking twitter users.

Acknowledgements

This work was not possible to finish without the help and support of different people and organizations. First of all, I would like to thank my advisor, Prof. Fabio Crestani, who was not only a supportive advisor but also a good friend. His academic advices always helped me to find my way in the research and our friendly discussions were some of the best moments in my PhD. I like also to thank members of Center for Intelligent Information Retrieval (CIIR) for hosting me and supporting me during my visit from the center. Specially I thank Prof. Bruce Croft for his constructive advices and encouraging discussions.

I thank funding organizations who provided us with financial supports that made all of it possible. I specially thank Swiss National Science Foundation (SNSF) for being the main contributor in the costs of this work. The thesis was mainly supported by SNSF grant for the "Cross-Media Indexing (XMI) for Multimedia Information Retrieval" project (ProjectNr. 200021-117994/1). I would like also to thank SNSF for the visiting grant that they provided me to visit University of Massachusetts at Amherst for six months in 2011. The grant was supported as the "Temporal Analysis of User Generated Data" project (Grant Nr. PBTIP2-135848). Also I like to thank COST action for providing me a grant to visit The Sapienza University of Rome for two weeks in 2010.

I like to thank my parents who always encouraged me in my education and tried their best to not let any concern to interfere with my progress. I also like to thank my other family members, brothers, sisters, nephews, nieces, brothers and sister in laws for their pure love that was always there for me. I specially thank my older brother, Mohammad, who was best friend, best mentor and best brother that I could wish for. His supports still encourage me in my life and his memories will last with me forever.

Finally I like to thank my friends in University of Lugano and specially members of IR Group who created a friendly environment to work and discuss. Special thanks to Mark Carman, who significantly helped me in the first years of my PhD, and Amirhossein Malekpour, who was always there to discuss with me and challenge my ideas.

Contents

Figures

Tables

Chapter 1

Introduction

1.1 Motivation

Social media and user generated content are growing rapidly and have become one of the most important sources of information on the Web. Forums, mailing lists, on-line discussions and social networks like Facebook are examples of these resources, which have lately attracted the attention of researchers. Social media have provided a new environment in which users are not just consumers of the data but also they produce it. Every person, at the same time that he consumes available information, generates new data and shares them with other people through social media. This leads to huge amount of information being generated every day.

Given the amount of information being generated in social media, there is a pressing need for quality retrieval systems for accessing this content. Among different retrieval scenarios there are two main types of information needs that one can search for in the social media. One type of queries search for single documents related to a specific information need. However, there are other types of queries that are searching for whole stream of data related to one of users' information need.

Most of the current retrieval systems for user generated data focus on retrieving a single document and there has been little work on retrieving and ranking the whole stream of data. Ranking and retrieval of streams can be useful in different scenarios like twitter user recommendation, ranking the forum threads or ranking the blogs. As an example consider recommending similar Twitter users after receiving the interest of a Twitter user as a query. Those retrieved users should have similar interests as the query user and have regular posting on those topics. The same retrieval system can be beneficial in ranking of threads

in forums, ranking experts or ranking of blog feeds.

Although stream ranking has different applications and can be useful in different scenarios, it has not been well explored yet. Most of the available search engines perform the same document-level retrieval algorithms on user generated contents that are used for regular web data. In this thesis we try to go one step further and investigate the problem of ranking user generated streams. We point out the challenges that differentiate stream ranking from traditional information retrieval (IR) tasks and propose solutions for those challenges. As a case study we apply our solutions to the blog feed retrieval problem, but we believe that the solutions are general and applicable to other similar problems.

The goal of a blog feed retrieval system is to rank blog feeds in response to a given query. The item of retrieval is the blog as a whole that is a collection of documents. A relevant blog is assumed to report regularly about the topic and user wants to add it to his RSS reader. There are different properties of blogs that make blog retrieval a challenging task and different from other IR problems. Among all the properties, we focus on the following properties of blogs and propose solutions for solving/integrating them in a retrieval system:

1. Term-mismatch problem between posts and topics.

2. Evolution of blogs and topics.

3. Diversity of blog posts.

Each of these properties creates new challenges for the retrieval system and the goal of this thesis is to address these challenges.

1.2 Research Outline and Questions

In this section we describe the main research questions that will be answered in the rest of the thesis. The questions are divided into three groups and each group is about one of the main challenges that we found throughout our research.

1.2.1 Term Mismatch Problem

One of the challenges in any information retrieval task is the term-mismatch problem between queries, on one hand, and relevant documents on the other hand. By term-mismatch we mean that the query terms are not exactly same terms that occur in the relevant documents. Term-mismatch can be caused by

different reasons for example ambiguity of the words, spelling mistakes, different user intentions, etc. While this problem is common in every retrieval task, it is more severe in blog feed retrieval.

The term-mismatch problem in blog retrieval is due to two main reasons: 1) the noisy nature of user generated content, 2) generality of topics (multi-faceted topics with many possible subtopics) in blog feed retrieval. Blog posts are usually more spontaneous and less formal than other Information Retrieval (IR) test collections like news wires. Spelling mistakes are more common in blog posts which makes it harder to precisely estimate the term probability distributions in them. The high number of spam blogs and spam comments further intensify this problem. On the other hand, topics in blog distillation are very general, they are often multifaceted and can be discussed from different perspectives [EACC08]. The generality of topics makes the term mismatch problem between documents and topics even more important. Since the topics are looking for a collection of decrements as apposed to single documents, they usually have multiple aspects each which might need demand terms in their corresponding relevant documents

Since solving the term mismatch problem can have significant effects on the retrieval performance of a system, our first attempt is to find a way to eliminate or reduce the effects of term mismatch on the retrieval system. Our main idea for solving the term mismatch problem is to use the content based similarity between documents as an extra information in order to better represent a document. Although each document might not have enough information for precisely estimating its relevance to a query, other documents that are similar to it can provide us with a richer representation of the document. Based on this idea, the first question that we ask concerns the applicability of content-based similarity in distinguishing between relevant and non-relevant blogs:

RQ1: Is content-based similarity between posts useful for distinguishing between relevant and non-relevant blogs?

The previous question leads us to more detailed questions about using the similarity between posts in the retrieval system. We employ two different approaches for integrating similarities into the retrieval system. The first approach is a regularization method that changes the relevance score of posts according to new evidence. In this method we use the similarities to directly change the score of a post and then use the new scores in estimating blog relevance to the query. The second approach is a smoothing technique that changes the term

probabilities in a post based on its relation with other posts and might estimate non-zero probabilities for query terms that do not even occur in the post. In this approach, we indirectly change the score of a post by changing its term probabilities. The resulting term probabilities are then used for estimating post relevance and later estimating blog relevance scores. Our next questions are about these two approaches and the way to use them in blog retrieval:

RQ2: How can we use content-based similarity between blog posts to regularize their relevance scores?
RQ3: How can we use content-based similarity between blog posts to smooth their term probability distributions?

Both of the previous approaches utilize similarity between posts as a type of dependency. Content-based similarity is one type of dependency but it is not the only one. Other relations like hyperlink or temporal distance can also be considered as dependency between posts and be employed in the smoothing models.

Hyperlinks represent very meaningful relations between documents and have been employed in web retrieval in different ways. However, in current blog collections there are not many links between the posts. In the Blog06 collection there are only 3% of posts that have a link to another post in the collection. Lack of hyperlinks in the collection makes it difficult to extract useful information from them.

Temporal relations are another type of dependencies that are more available in the collections. We assume that temporal distance between two posts can give us an idea about their content. For example we can consider a topic that has different subtopics and those subtopics change over time. If two posts talk about the same general topic and are published around the same date, it is very likely that they discuss the same subtopic. Based on this assumption and as an extension of previous smoothing models, we try to add temporal relations into the models. In this direction, the main questions that we aim to answer are the following:

RQ4: How can we extend smoothing models to take into account temporal relations between posts?
RQ5: Are temporal relations between posts useful in a blog retrieval system?

1.2.2 Evolution

One of the properties of the blog retrieval task is its dependency on time. Time dependency in ranking blogs is a twofold phenomena. On one hand query topics can be dependent on time and on the other hand blogs can change over time.

There are many topics that have a time-dependent nature. These topics can have different meaning at different times or have different sub-topics at different times. For example consider a query about "Apple products". Any time that Apple releases a new product, the relevant terms to this query can change. If the user is looking for a blog that writes about "Apple products" and wants to read this blog frequently, he would expect the blog to be up-to-date and write about a new product anytime one is released. Thus we need to extract the most appropriate terms for the topic at each time point and consider them in estimating blog relevance.

Beside topics, blogs can change over time as well. Bloggers can change their interests and discuss totally different topics over time. Thus for a given topic, they might publish relevant posts for a while and stop talking about it after that.

Due to the importance of time, we need to incorporate it in our retrieval model. To this end, we want to model the evolution of topics and find blogs that have similar trends as the query topics. This modeling enables us to retrieve blogs that not only have relevant content to the query but also publish up-to-date information with respect to the query topics.

We propose an approach in which we model both blogs and queries as time series of term distributions. In order to generate a time-based representation for the queries, we use a variation of the pseudo relevance feedback technique. In this technique, we assume that the top retrieved posts for each query are relevant and are published around the proper dates. Based on this assumption, we can extract the most representative terms for the query at each point in time. The first question that we aim to answer is about the term extraction process:

> *RQ6: How can we find appropriate time-dependent expansion terms for a given query?*

We need to estimate the term distribution for blogs and queries at all time points to have their full time series model. After selecting expansion terms for the query, we have the term distributions for some of the days and we need to use them to estimate term probabilities for other days for which no observation is available. There is the same situation with blogs that do not publish a post

everyday. In other words, we have partially observed data for blogs and queries and want to use them in order to fully represent the corresponding time series. The next question to answer aims to find a way for representing items (blogs or queries) over time:

RQ7: How can we fully represent blogs and queries over time?

Finally we need a way to calculate similarity between the resulting representation of blogs and queries. Since we represent items as time series of term distributions, we can use time series similarity measures for calculating similarity between them. The resulting similarity between each blog and query will be used as the relevance score of the blog with respect to the query. The next question that we try to answer is about available similarity measures:

RQ8: How can we calculate similarity between the time-based representation of blogs and queries?

1.2.3 Diversity

For most of the topics, there is a lot of information available on the web and usually users would not like to read the same piece of information multiple times in their subscribed blogs. Thus users would not like to read repetitive and non-informative documents. If a user reads a blog frequently, he would like to see new information any time that the blog publishes a post. In this situation, any source of information that provides more informative and novel data would be a better choice for the users to follow.

Based on this idea, we try to capture diversity of blog posts in our weighting model and take it into account for ranking blogs. To this end, we need to consider the dependency between posts in the same blog and give higher score to blogs that have more novelty among their posts. Our main strategy in implementing this idea is that a blog will not gain any further relevance score by publishing those relevant pieces of information that it has previously published. This brings the idea that blog posts should not be similar to each other and each of them should discuss a different topic. However, just the dissimilarity of blog posts is not a good indicator of blog relevance and they should also be relevant to the query. We define the concept of on-topic diversity in which we require

blog posts to be related to the query topic and dissimilar to each other. We use this concept in our retrieval model and show that it can help us in distinguishing relevant blogs from non-relevant ones.

Since the existing blog collections and corresponding query judgments do not require diversity in the assessing process, it is not clear if diversity is a valid assumption in those collections. The first question that we answer is to check the relation between diversity and relevance in the existing data sets:

RQ9: How important is the diversity of blog posts to blog relevance?

It turns out that although diversity is not explicitly considered in the assessing process of TREC queries, it still exists in the data sets and relevant blogs are more likely to have higher on-topic diversity among their posts. Based on this finding, we further investigate the problem and try to integrate diversity in the existing blog retrieval methods. We define different types of diversity measures that can have meaningful interpretations in the blog retrieval problem and propose a general approach for using them in different retrieval methods. The main questions that we answer in this section are the following:

RQ10: What types of diversity measures can we define over blog posts?
RQ11: How can we capture the diversity of blog posts and integrate it into a blog feed retrieval method?

The main type of diversity measures examines content-based (topical) diversity of blog posts. In this type of diversity we assume that high content-based similarity between blog posts shows low diversity among them. However, besides content-based diversity, one can expect a relevant blog to have other type of diversities as well. As we discussed about importance of the temporal information in blog retrieval, our next diversity measure evaluates temporal diversity of posts. By temporal diversity we mean that blog posts should have high coverage over the temporal space and not be concentrated on specific time windows. This measure captures the fact that the blog talks about the topic regularly over time. Finally we propose a method in which we combine topical and temporal diversity measures and give higher score to blogs that publish novel posts at different points of time.

Finally we investigate the effect of diversity-based methods on different type of queries. There are different categories of queries and each category has different properties. While diversity can be useful for some types of query, it might

be less effective for other types. Our next attempt is to answer the following question regarding query types:

> *RQ12: For what type of queries can we expect diversity-based methods to be more effective?*

1.3 Contributions

The main contributions of this thesis can be summarized as follows:

- We investigate the importance of time in blog feed retrieval systems. We propose different approaches to take time into account for extracting useful information that can help us better estimate a blog relevance to a given query.
- We propose smoothing techniques that employ dependency between blog posts in order to have a better relevance estimation for posts. The new posts relevance score is then used in the existing blog retrieval methods. We consider topical and temporal dependency as sources of smoothing and propose a framework to combine the two in a unified representation.
- We propose a pseudo relevance feedback technique that models evolution of topics over time and correspondingly generates expanded queries for further retrieval. This model help us to retrieve blogs that publish relevant and up-to-date information with respect to the query.
- We investigate the importance of diversity in blog feed retrieval. We show that relevant blogs are more likely to publish diverse information related to the query topic. Further we propose different types of diversity and integrate them into existing blog retrieval techniques.

1.4 Publications

The contributions of our studies resulted in multiple publications in related journals and conferences. Some of these publications are directly presented in the thesis and some others are the basis for other works that shaped the thesis over time. A list of these publications is presented in the following:

- **Refereed Journal Papers**

  P1 Mostafa Keikha, Fabio Crestani. **Linguistic Aggregation Methods in Blog Retrieval.** *Journal of Information Processing and Management (IPM)*, volume 48(3), pp. 467-475, 2012.

  P1 Mostafa Keikha, Fabio Crestani, Mark James Carman. **Employing document dependency in blog search.** *Journal of the American Society for Information Science and Technology (JASIST)*, volume 63(2), pp. 354-365, 2012.

- **Refereed Conference Papers**

  P3 Mostafa Keikha, Shima Gerani, Fabio Crestani. **TEMPER: A Temporal Relevance Feedback Method.** *In Proceedings of ECIR 2011*, pp. 436-447, 2011.

  P4 Mostafa Keikha, Shima Gerani, Fabio Crestani. **Relevance stability in blog retrieval.** *In Proceedings of SAC 2011*, pp. 1119-1123, 2011.

  P5 Mostafa Keikha, Fabio Crestani. **Effectiveness of Aggregation Methods in Blog Distillation.** *In Proceedings of Flexible Query Answering Systems 2009*, pp. 157-167, 2009.

  P6 Mostafa Keikha, Fabio Crestani, W. Bruce Croft. **Diversity in Blog Feed Retrieval.** *Accepted in CIKM 2012.*.

- **Conference Posters and Workshop Papers**

  P7 Mostafa Keikha, Jangwon Seo, W. Bruce Croft, Fabio Crestani. **Predicting document effectiveness in pseudo relevance feedback.** *In Proceedings of CIKM 2011*, pp. 2061-2064, 2011.

  P8 Mostafa Keikha, Shima Gerani, Fabio Crestani. **Time-based relevance models.** *In Proceedings of SIGIR 2011*, pp. 1087-1088, 2011.

  P9 Mostafa Keikha, Mark James Carman, Fabio Crestani. **Blog distillation using random walks** *In Proceedings of SIGIR 2009*, pp. 638-639, 2009.

  P10 Mostafa Keikha, Fabio Crestani. **Experimental Results on the Aggregation Methods in Blog Distillation** *In Proceedings of Web Intelligence Workshops 2009*, pp. 151-154, 2009.

1.5 Thesis Outline

The rest of the thesis is organized as the following. Chapter 2 reviews the main concepts of information retrieval and the state of the art and related work in the area of blog feed search. Chapter 3 describes our experimental set up including collections, tools and evaluation measures that we used in our experiments. It also explains the language modeling approach with a simple smoothing technique that is used in most of our methods as a basic component. Chapter 4 discusses our approaches for solving term mismatch problem in blog feed search. We propose two methods for smoothing term probabilities and regularizing relevance scores and show their effect on the retrieval performance of the system. Chapter 5 investigates the temporal evolution of blogs and describes an approach to take time into account as a new parameter in pseudo relevance feedback. Chapter 6 presents diversity-based methods that take the novelty of blog posts into account when the relevance scores are calculated for blogs. It describes multiple diversity measures that are applicable to blog feed retrieval and analyses their effect on different systems and for different query types. Finally chapter 7 concludes the thesis by shortly explaining our findings and indicating possible directions for future work.

Chapter 2

Background

In this chapter we discuss the main concepts of information retrieval that will be used in the rest of the thesis. Then we describe temporal information retrieval (IR) methods and importance of time in different IR tasks. Finally, we review the main methods of blog retrieval and describe some of the methods that are used as our baselines in more detail. In the following chapters, whenever it is necessary, further background overview and citations are provided.

2.1 Information Retrieval

Storing and finding information has been an important problem for thousands of years [Sin01]. However, by increasing amount of available information through web, possibility of retrieving useful information became even more essential. The goal of an information retrieval system can be summarized as finding relevant documents in response to a given user information need. To this end, a system has access to a collection of documents that are preprocessed and indexed for future retrieval. A user submits his information need as a query and the system retrieves a ranked list of documents based on some relevance criteria. The main components of an IR process are as following:

- **Document Processing and Indexing**: The first step of any IR system is to process the available collection of documents and prepare them for the retrieval step. This collection can be a set of web pages or any other type of documents like patent applications, book chapters, etc. The output of this step, called *inverted index*, is usually a mapping from terms to their containing documents which provides a fast access to the documents [BYRN99]. Different transformations of texts can be done during

document processing. Stop-words removal, stemming [Hul96], phrase extraction [Fag89] are few examples of indexing-time transformations.

- **Query Processing**: When receiving a user query, in order to be able to match documents to the query, the system needs to perform the same transformations as done on the documents to the query. Beside the basic transformations, some more advanced operations can be done in this phase. Examples of such operations are query refinement [MRS08], query expansion using thesaurus [SJ71], relevance feedback [Roc71] or pseudo relevance feedback [BSAS95; LC01a].

- **Document Matching and Retrieval**: The main goal of the IR system is to match documents to the query. This component takes the output of the previous components and based on the existing evidence, assigns a relevance score to each document. Finally, it ranks documents based on their relevance scores. This component has two subcomponents:

  - Weighting Models: The goal of the weighting model is to assign a score to each document-query pair. There are different weighting models including Boolean models [MRS08], vector space models [SWY75], probabilistic models [RJ76], language models [PC98] and learning to rank methods [Liu09]. Each of these models has different assumptions and different components for weighting documents. However, most of them share some important assumptions and weighting components. Two widely used assumptions are term independence and documents independence which mean that the relevance of any term/document does not depend on the relevance of other terms/documents. The main weighting components that most of the methods use are term frequency (frequency of a query term in the document), inverse document frequency (related to the frequency of a query term in the collection) and document length.
  - Ranking Models: A ranking model takes the score of documents as input and generate a ranked list of documents as output. In the simplest model, where documents are assumed to be independent, the system ranks documents in a descending order of their scores [Rob77]. More advanced techniques take the dependency between documents into consideration for ranking documents [ZCL03]. Those techniques aim to model the diversity between documents [CG98] or the risk of a retrieved ranked list [WZ09].

- **Performance Evaluation**: Evaluation measures are what enable us to compare different models and decide on effectiveness of new models or features. Two important measures that are the base for most of the other measures are *precision* and *recall*. Precision measures what percentage of retrieved documents are relevant. Recall measure what percentage of relevant documents are retrieved. Detailed description of evaluation measures that we use in our experiments can be found in the next section.

2.1.1 Text Retrieval Conference (TREC)

The Text REtrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense, started in 1992 as part of the TIPSTER Text program. Its purpose is to support research within the information retrieval community by providing standard benchmarks for large-scale evaluation of text retrieval methodologies.

A TREC workshop consists of a set of tracks in which particular retrieval tasks are defined. Each track aims to a particular new challenge in information retrieval domain and tries to provide necessary material for researchers to investigate the problem in hand. These materials usually include data collection, queries and their judgments and evaluation methodologies and tools. Each track usually runs for multiple consecutive years until the organizers are convinced that they have provided enough materials for researchers to keep working on the problem. Following are examples of interesting tracks that have been proposed in TREC in recent years:

- **Novelty Track**: The goal of the system in the novelty track is to find relevant and new information for a given query. Systems are asked to return a ranked list of sentences that are relevant to the query and have new information compared to other provided sentences [Sob04].

- **Entity Track**: The goal is to answer information needs that are looking for specific entities like person, organization, location, etc [BSdV10]. The main task requires the system to return a list of entities and their homepages given a information need that indicates the target entity and its type.

- **Legal Track**: The goal is to facilitate the search over electronic documents for the legal community that can help them to find information about legal cases before making any judgment/decision [HTBO09].

- **Relevance Feedback Track**: The goal is to have a deeper understanding and analysis of the relevance feedback process for improving retrieval effectiveness. The systems are provided with a set of queries and relevant documents using which they are asked to retrieve the final list of relevant documents.

- **Session Track**: The goal of the track is to answer information needs that are expressed in a multiple-query session. If the user is not happy with the results of his query, he can change the query and submit a more general or more specific query or even a query with different information need. Systems are asked to use the knowledge of the previous queries and generate a better ranked lists of documents.

- **Blog Track**: In the blog track, the main goal is to investigate the information needs in the blogosphere and to develop methods that are specific for blog documents. Most of the work of this thesis is around the tasks that are defined in blog track and so we will describe it in more detail in the next section.

2.1.2 Blog Track in TREC

The blog track started in 2006 and continued until 2010 aiming to answer users' information needs in blogosphere. Since 2011 the track is continued as the micro-blog track that has a focus on short text generated by users on twitter. Different tasks were introduced in the course of the blog track that tried to analyse blog data from different perspectives.

In TREC 2006 there were two tasks: a main task, opinion retrieval, and an open task. The opinion retrieval task focuses on a specific aspect of blogs that is related to the opinionated nature of many blogs. The goal of opinion retrieval is to retrieve list of documents that are relevant to the topic and also contain an opinionated view around the topic. The second task was introduced to allow participants the opportunity to influence the decision about a suitable second task for 2007 on other aspects of blogs, such as the temporal/event-related nature of many blogs, or the severity of spam in the blogsphere.

In TREC 2007 a new main task and a new sub-task were added to the blog track. The new tasks were blog distillation (feed search) task and polarity sub-task, along with a second year of the opinion retrieval task. The polarity sub-task was added as a natural extension of the opinion task, and was intended to represent a text classification-related task which required participants to determine

the polarity (or orientation) of the opinions in the retrieved documents, namely whether the opinions were positive, negative or mixed. The newly introduced blog distillation task was an articulation of an ad hoc search task where users wish to identify blogs (i.e. feeds) about a given topic which they can subscribe to and read on a regular basis.

TREC 2008 continued the same tasks as the previous year investigating on opinion detection at post level and blog distillation at blog level. In TREC 2009 two new tasks were introduced considering other aspects of blog retrieval. These new tasks were faceted blog distillation and top stories identification.

Faceted blog distillation is a more complex and refined version of the blog distillation task where the goal is to retrieve the whole blogs as the retrieval units [MOS09]. The task, which was first introduced by Hearst *et al.* [HHD08], aims to consider not only the blog relevance to the topic but also the "quality aspects" (facets) of the blog. For each query, a specific facet is determined and only blogs that satisfy that facet are considered to be relevant. The introduced features in TREC 2009 include opinionated vs. factual, personal vs. company and in-depth vs. shallow facets [MOS09]. For example queries with personal facets are looking for those blogs that are relevant to the query and are published by a person as opposed to a company or an organization.

Top stories identification is another task where the goal it to rank new stories based on their importance that is estimated using blog data. The participants are given a collection of blog posts, a collection of news stories and a set of queries that are dates. Systems are asked to use the evidence in the blog collection in order to understand what are the most important new stories for the given date. In other words, they wanted to check if blog data had any relation with news data and if they could predict the most important news headlines.

2.2 Time in Information Retrieval

Time has been used in ad-hoc information retrieval in different ways. One of the ways to utilize temporal information is to use the temporal metadata of the retrieved documents in order to learn properties of the corresponding query. Diaz and Jones used time-based query features to build a query profile from which they predicted query precision [DJ04; JD07]. Predicting query performance is an interesting task that can enable the retrieval system to have different behaviours for different queries based on the predicted performance [CTZC02]. Diaz and Jones expanded the existing query prediction methods by adding temporal features to the predicting models. They extracted the temporal features

using the time-stamp meta-data of the top retrieved documents for the query.

Another way to employ temporal information is to use it in the weighting models where the score of a document depends on its time-based metadate. Li and Croft [LC03] and Erfon and Golovchinsky [EG11] used time-based methods in order to rank recent information for those queries for which recency is an important dimension. They introduced instances of language modelling framework which consider the publish date of the documents in scoring them. Traditional language model approach assigns a uniform prior probability to the documents [PC98; LC01a]. Li and Croft replaced the uniform distribution with an exponential decay function which gives higher weight to newer documents [LC03]. A similar approach was used by Dakka *et al.* where they proposed a more general framework for estimating temporal importance based on the properties of top retrieved documents for each query [DGI08]. Erfon and Golovchinsky went further and considered temporal information not only for the prior estimation of the documents, but also for the parameters of smoothing and pseudo relevance feedback methods [EG11]. Perkiö *et al.* proposed a relevance ranking algorithm that explicitly models the temporal behaviour of documents and queries on top of a statistical topic model [PBT05].

The time-stamp of links between pages is another temporal metadate that can be used in processing documents. Yu *et al.* used the date of hyperlinks in a variation of the PageRank algorithm [BP98] in order to accommodate the lack of links for new pages [YLL04]. The traditional PageRank algorithm works with a static version of web graph, while the web is dynamic and everyday new pages are added to it. Thus they hypothesized that the new pages have fewer in-links than old pages and consequently lower scores, even when the new pages are totally relevant and informative. Based on this hypothesis, they introduced a new version of PageRank algorithm that weights the links based on their dates and show that it is more effective than traditional PageRank algorithm. A similar approach was proposed by Dai and Davidson where they combined the freshness of page content with the freshness of its in-links for estimating the authority of the page on the web [DD10]. They proposed two measure for freshness of a page: Link freshness and Page freshness. In each of the measures, they take into account the activities of the page and those of its connected pages in order to estimate the page freshness. They proposed an authority estimation framework, called T-Fresh, that combines all the collected evidence in estimating the final freshness score and showed that freshness is an importance factor in the page authority.

Analysing the dynamics of documents and its effect on the document relevance is another way to employ temporal information. Jatowt *et al.* considered

the changes of a web page in their ranking method aiming to retrieve fresh and relevant documents [JKT05]. For each top retrieved web pages for a query, they analyse its history and extract the changes of the page. A page that has more recent query-related changes would have a higher score. Dynamics of the pages is also considered by Elsas and Dumais [ED10]. They show a strong relationship between the content change and relevance of a page. Furthermore, they employ a query independent document prior which favors dynamic documents and show that it can improve the performance of the retrieval system. They also model the document language model as a mixture of long-term, mid-term and short-term document language models. These three language models correspond to the three virtual documents which are created from terms appearing in all of the time slices, many time slices and a small number of time slices, respectively. In order to score a document, they combine the three language models and find the best combination with respect to the retrieval performance.

Beside temporal distribution and dynamics of documents, evolution of language has been studied for different purposes in information retrieval domain. Shaparenko *et al.* use the content of documents to extract temporal patterns of topics in a collection of documents [SCGJ95]. They extract the most important topics and model the change of their popularity over time. Based on the model of popularity, they find the most influential authors and documents in the collection. Jong *et al.* study the evolution of historical documents in order to disambiguate the words and be able to retrieve documents from old document collections based on queries in contemporary language [JD07].

Finally, Topic Detection and Tracking (TDT) is another closely related problem where temporal analysis of data plays an important role [ACD$^+$98]. The goal of a TDT system is to find and follow the events in a stream of news broadcast. To this end, the system should be able to perform three tasks: segment the stream of news into disjoint stories, detect the stories that discuss a new event for the first time and find the related stories that are published later until the event disappears from the news. Swan and Allan find a new event by estimating the significance of its terms [SA99; SJ00]. They propose a statistical test for determining the significance of a term in a specific time slice. To this end, they test the hypothesis that the distribution of a term in time t_0 is not different than its distribution in other time slices. The rejection of this hypothesis is an indicator of term significance in time t_0 and the corresponding stories is assumed to discuss a new event. Allan *et al.* use a clustering techniques for detecting stories that contains new events which is similar to information filtering models [APL98].

2.3 Blog Feed Retrieval Methods

Research in blog distillation started mostly after 2007, when the TREC organizers proposed the task as part of the blog track. Researchers have employed different approaches from related areas such as ad-hoc search, expert search, and resource selection in distributed information retrieval [SMM+12].

The simplest models use ad-hoc search methods for finding blog relevance to a specific topic. They treat each blog as one long document created by concatenating all of its posts together [ETO07a; EACC08; SC08]. These methods ignore any specific property of blogs and usually use standard IR techniques to rank blogs. Despite their simplicity, these methods perform fairly well in blog retrieval.

Some other approaches based on expert search methods have been applied to blog retrieval. Expert search is a task in the TREC Enterprise Track where systems are asked to rank candidate experts with respect to their predicted expertise about a query, using documentary evidence of the expertise found in the collection [SdVC06]. Based on the similarity between blog distillation and expert search, some researchers have adapted expert retrieval methods for blog retrieval [BdRW08; MO08]. In these models, each post in a blog is seen as evidence of blog interest in the query topic. Balog *et al.* adapt two language modeling approaches of expert finding and show their effectiveness in blog distillation [BdRW08]. MacDonald and Ounis use data fusion models to combine post-based evidence to compute a final relevance score of the blog [MO08].

Other researchers have employed resource selection methods from distributed information retrieval for blog retrieval. In distributed information retrieval, the cost of searching all servers for each query is considered prohibitively expensive, so server selection algorithms are used [HT05]. Queries are then routed only to servers that are likely to have many relevant documents for the query. Elsas *et al.* deal with blog distillation as a resource selection problem [EACC08; AECC08]. They model each blog as a collection of posts and use a Language Modeling approach to select the best collection. Similar work is described by Seo *et al.*, which they call Pseudo Cluster Selection [SC08].

In the following, we describe the main methods from each category in more detail. These methods are used as baseline methods through our experiments. We choose these methods because they are state of the art techniques and are among most effective and theoretically reasonable methods that are developed for blog retrieval [MSOS10a].

2.3.1 Blog Distillation as an Ad-hoc Search Problem

As the simplest blog retrieval model, one can skip all blog-specific features and use the ad-hoc information retrieval methods. Efron *et al.* generate one document for each blog by concatenating all the blog posts and use ad-hoc search methods for finding relevant blog to a specific topic. Given a query q they derive a score for each blog b in the corpus by the negative KL-divergence between the query language model and the language model for b.

$$\begin{aligned} s(q,b) &= -D(\theta_q \mid\mid \theta_b) \\ &= -\sum_w p(w|\theta_q)\ \log(p(w|\theta_b)) \end{aligned}$$

The KL-divergence measures how different is the language of query to that of the blog. Thus the negative KL-divergence is used as a similarity measure between the two languages. They use this method to compute similarity of topic to both blog and postings. Finally using linear combination, they combine the blog score with its postings score as the final score of the blog. Blogs are retrieved by decreasing order of their scores.

A similar approach is proposed by Elsas *et al.* that is called large document model (LDM). The Large Document Model (LDM) treats each blog as a single document created by concatenating all of its posts together. Blogs are ranked proportional to their posterior probability given the query:

$$\begin{aligned} P_{LDM}(b \mid q) &= P_{LDM}(q,b)/P(q) \\ &\overset{rank}{=} \underbrace{P(b)}_{Feed\ Prior}\ \underbrace{P_{LDM}(q \mid b)}_{Query\ Likelihood} \end{aligned}$$

If other types of information are available, like statistics regarding the number of subscribers for each blog, they can be used to set the blog prior $P(B)$ to an appropriate value. Otherwise the blog prior can be set uniformly or allowed to grow logarithmically with the number of posts in the blog, so as to favor longer blogs since they are more likely to contain useful information [EACC08].

Once a blog prior is chosen, we need to estimate the query likelihood. The query likelihood, $P(q|b)$, in LDM is estimated using a Dirichlet-smoothed term probability as the following:

$$P_{\mathtt{LDM}}(q|b) = \prod_{t \in q} P_{\mathtt{LDM}}(t|b) = \prod_{t \in q} \left(\frac{\mathrm{tf}(t,b) + \mu P_{\mathtt{ML}}(t|C)}{(\sum_t \mathrm{tf}(t,b)) + \mu} \right)$$

where $\text{tf}(t, b)$ is the total number of occurrences of term t in posts in the blog, μ is a smoothing parameter, and $P_{\text{ML}}(t|C)$ is the Maximum Likelihood (ML) estimate for the term distribution in the collection as a whole (i.e., the relative frequency across all the blogs).

In extension of their work, they use different document representations (for example, permalink text, feed title, entry titles and entry content) and estimate the query likelihood component as a weighted combination of query likelihood from the different document representations[AEY+08]:

$$P_{LDM}(Q \mid b) = \prod_j P_{LDM}(Q \mid b^{(j)})^{v_j}$$

where the j denotes different representations and v_j are learned weights for each of those representations. Query likelihood for each representation is estimated as before, using Dirichlet-smoothed maximum likelihood estimates.

2.3.2 Blog Distillation as a Resource Selection Problem

Resource selection in distributed information retrieval (DIR) is a similar problems to blog distillation task. The goal of a resource selection system is to rank collections servers in order to submit the query to the servers with highest probabilities for having relevant documents. Due to the cost of search, it is not possible to search all servers for each query. Thus some server selection algorithms should be employed. Some of the main server selection techniques are CORI [CLC95], KL Divergence [XC99] and ReDDE [SC03a].

The Small Document Model (SDM) is one of the methods that deals with blog distillation as a resource selection problem [EACC08; AECC08]. In these methods, each blog is modeled as a collection of posts and a Language Modeling approach is used to select the best collection. These methods are well justified from a probabilistic perspective and have been shown to perform well in practice. SDM is based on the ReDDE algorithm for resource selection in distributed information retrieval.

Similar to the large document model, the SDM ranks blogs according to their posterior probabilities given the query. However the way that SDM estimates the query likelihood of a blog is different than that of LDM. SDM treats each post within a blog as a separate document and attempts to quantify the importance (centrality) of each post to the rest of the content in the blog. The query likelihood for a blog is calculated by summing the query likelihoods for each post

in the blog scaled according to the probability (centrality) of the post within the blog:

$$P_{\texttt{SDM}}(q|b) = \sum_{p \in b} P(q|p)P(p|b) \tag{2.1}$$

Here p is a post in the blog, and $P(q|p)$ is the query likelihood for each post. This equation holds if we assume queries are conditionally independent of feeds given the entry. $P(q|p)$ is computed over query terms using Jelinek-Mercer smoothing:

$$P(q|p) = \prod_{t \in q} \lambda_p P_{\texttt{ML}}(t|p) + \lambda_b P_{\texttt{ML}}(t|b) + \lambda_C P_{\texttt{ML}}(t|C) \tag{2.2}$$

Where $\sum \lambda_* = 1$, $\lambda_* \geq 0$ and $P_{\texttt{ML}}(t|*) = \frac{tf(t,*)}{\sum_t tf(t,*)}$. The post centrality is computed using the likelihood that the blog would generate terms in the post:

$$P(p|b) \approx \prod_{t \in p} P(t|b)^{P_{\texttt{ML}}(t|p)}$$

Here $P(t|b)$ is estimated by average term probabilities in the blog posts using $\frac{1}{N_b} \sum_{p \in b} P_{\texttt{ML}}(t|p)$ where N_b is the number of posts in the blog [EACC08].

Seo and Croft also employed resource selection techniques in their blog retrieval method called Pseudo Cluster Selection (PCS). They assume that the highly ranked documents from each blog represent a pseudo-cluster of posts in that blog. In order to rank blogs, they use the cluster representation as the blog representation that is used for calculating the score of blogs. To this end, they use the geometric mean of document language models as cluster representation. Their ranking function is formed as follows:

$$score_{\texttt{PCS}}(b_i, q) = \left(\prod_{j=1}^{k} score(p_{ij}, q) \right)^{\frac{1}{k}}$$

here p_{ij} is the j-th top retrieved post from blog b_i. As can be seen, the final score of a blog is the geometric mean of query likelihood of the top k retrieved posts from that blog. In this method, if the blog has more than k posts retrieved for the query, the rest of the posts are ignored. However, if the blog has m posts where $m < k$, we need to find a way to give a non-zero score to the $(k - m)$ missing operands. Authors use the minimum score among all the retrieved posts as the score for the missing posts:

$$p_{min} = arg \min_{p_{ij}} P(q|p_{ij})$$

The final scoring function is obtained by replacing the missing posts with the minimum score:

$$\widetilde{score}_{\text{PCS}}(q, b_i) = \left(P(q|p_{min})^{K-\tilde{n}} \prod_{j=1}^{K-\tilde{n}} P(q|p_{ij}) \right)^{\frac{1}{k}}$$

where $\tilde{n}$ is the number of posts of the blog b_i that are retrieved among the top N posts for the query.

2.3.3 Blog Distillation as an Expert Search Problem

Expert Search is a task in the TREC Enterprise Track where systems are asked to rank candidate experts with respect to their predicted expertise about a query, using documentary evidence of their expertise found in the collection [SdVC06]. In this problem, documents are evidence for person's expertise on specific topic and the goal is to use this evidence to find the best experts [MO06b].

MacDonald and Ounis see the expert search problem as a voting problem, which they model by adapting fusion techniques [MO06b]. By retrieving top documents for a given topic, they create a profile for each person that contains associated documents to that person. They see each retrieved document as an implicit vote for the corresponding candidate to be an expert in the given topic. Then aggregate these votes using data fusion techniques into a ranking of candidates. Previously, data fusion techniques were used to combine separate rankings of documents into a single ranking with the aim of improving the performance of any constituent ranking. As apposed to the normal application of data fusion techniques, the proposed approach for expert search aggregates votes from a single ranking of documents into a single ranking of candidates. In other words, here the goal is not to combine different rankings into one ranking. But the goal is to change the granularity if the items in the ranking that is given one ranking of documents we want to generate another raking of candidates. To this end, they use the document-to-candidate associations of the already-generated candidate profile.

Balog *et al.* employ a language modelling approach for finding experts given a query as the observed variable [BAdR06]. They propose two approaches, called Model1 and Model2, for estimating the likelihood of an expert to a given query. The difference between two approach is in they way that they model an expert. Model1 directly estimates the probability of a term to an expert which is then used for estimating query likelihood. In other word, Model1 does not

estimate the term probabilities in each document separately and instead it directly estimates term probabilities for the expert as a whole. In contrast, Model2 estimates document models instead of experts. In Model2, the query likelihood of the expert is estimated based on the likelihood of its associated documents.

Blog distillation can be seen as a similar problem to expert search where each blog is an expert and each post is an associated document to that person. Based on this similarity, expert search methods are adapted for blog retrieval.

Hannah *et al.* use the voting models for finding relevant blogs [HMP+07]. A blogger with an interest in a particular topic will regularly publish about the topic, and his/her posts will likely be retrieved in response to the query. Blog distillation can then be seen as a voting process: each time a post is retrieved in response to a query, it is considered as a weighted vote for the expertise of the blogger in that particular topic. They then use fusion methods to aggregate the relevance scores of posts in each blog and rank the blogs based on the aggregated scores. The best performing aggregation methods in their experiments were *ExpCombSum* and *ExpCombMNZ*. The first method *ExpCombSum* ranks blogs according to the sum of the exponents of the relevance scores for the most relevant posts from the blog:

$$score_{\texttt{ExpCombSum}}(b,q) = \sum_{p \in b \cap R(q)} \exp(score(p,q))$$

Here $R(q)$ denotes the set of posts retrieved for query q, the intersection $b \cap R(b)$ denotes only those retrieved posts that come from blog b and $score(p,q)$ denotes the relevance score assigned to the post p for the query by the underlying search engine. The second method, *ExpCombMNZ,* takes into account also number of the retrieved posts from each blog in the weighting model:

$$score_{\texttt{ExpCombMNZ}}(b,q) = |b \cap R(q)| \sum_{p \in b \cap R(q)} \exp(score(p,q))$$

The author further use the two fusion methods with different parameters, e.g size of top retrieved posts, as different evidence for blog relevance and use them in a learning to rank framework. They show that combining those methods can provide significant improvement over each of them alone [MO11a].

Balog *et al.* adapt their language modelling approaches of expert search for blog retrieval [BdRW08; WBdR10a]. Based on their previous models, Model1 and Model2, they propose two blog search models called Blogger Model and Posting Model. The Posting Model is similar to the Small Document Model proposed by Elsas *et al.* [EACC08] where query likelihood is estimated for each post and then combined to estimate query likelihood of blogs. On the other

hand, Blogger Model estimates term probabilities directly for each blog and then based on these term probabilities, it estimates the query likelihood of the blog:

$$p(q|blog) = \prod_{t \in q} \widehat{p}(t|blog)^{n(t,q)}$$

where $\widehat{p}(t|blog)$ is estimated based on the term probabilities in the associated posts:

$$p(t|blog) = \sum_{post \in blog} p(t|post) \cdot p(post|blog)$$

where $p(post|blog)$ is assumed to be uniform.

The final term probabilities are estimated by smoothing using the general term probabilities in order to get rid of zero probabilities:

$$\widehat{p}(t|blog) = \lambda p(t|blog) + (1 - \lambda) p(t)$$

After obtaining the query likelihood, blogs are ranked based on their posterior probabilities that is:

$$p(blog|q) \propto p(q|blog) \cdot p(blog)$$

where the blog prior $p(blog)$ is estimated by a uniform distribution. Authors further show that combining posting model and blogger model in a two stage retrieval framework can improve system efficiency [WBdR11; WBdR10a]. Since posting model is based on the post index, it is easy and fast to implement on top of existing index. However, blogger model is more effective than posting model but we need a blog-level index for it that can be expensive to build for all blogs. So in the two stage method, authors first retrieve an initial set of blogs based on the posting model and then re-rank those blogs using the blogger model. In this way the reduce the cost of blogger model while still benefit from its strong performance.

Other features like cohesiveness (how similar are the posts in a blog to each other) and anchor text similarity to the query (from links pointing to the blog) are also investigated in blog search. However, experiments indicated that these features did not improve performance significantly [HMP+07].

2.3.4 Temporal Evidence in Blog Distillation

Temporal information is one of the most important available information in the blogosphere. A blogger can change the topic of his blog after a while or can write

about different topics in different seasons. Most of the temporal analysis of blogs have been focused on recency of blog relevance with respect to a given topic. This type of models give higher scores to more recent posts before aggregating them. Such models show some improvement over the baseline which uses only the content of the blog [EWdR07; WBdR08].

MacDonald and Ounis try to capture recurring interests of blogs over time [MO08]. Following the intuition that a relevant blog will continue to publish relevant posts throughout the timescale of the collection, they divide the collection into a series of equal time intervals. They then weight the relevance score of each blog by the total relative number of its relevant posts over different time intervals. Although they try to capture the recurring interest of bloggers to the topic by this measure, it fails in some cases. For instance, if all posts of a blog are relevant to the query, different post distributions over time intervals make no difference in the model. Weerkamp *et al.* change the score of a post based on its publish date [WBdR08]. The more recent the post has been published, the higher the relevance score it would have.

Nunes *et al.* use temporal evidence as an extra feature of blogs beside their content [NRD08]. They use the temporal span and the temporal dispersion as two measures of relevance over time, and show that these features can help in blog retrieval. The temporal span of a topic in a feed corresponds to the period between the newest relevant post and the oldest relevant post. Temporal dispersion is the dispersion of relevant posts over the time and they use neg-entropy (negative entropy) to represent it. For calculating this value, first they convert each post's publish date to a relative global scale between zero and one (the first date is zero, other dates are the number of days difference from the first date, divided by the total number of days). Neg-entropy value of a blog is then calculated as following:

$$NE(B) = -1 \times \frac{\sum_{i \in R(B)} p(i) \times \ln(p(i))}{\ln(N)} \tag{2.3}$$

where B is the blog, R(B) is the set of relevant posts in the blog and N is the number of relevant posts in the blog. They re-rank blogs based on NE value and combine this rank by BM25 rank of the blog using linear combination.

2.4 Conclusion

In this section we introduced concepts and works related to the blog distillation task. We saw that different approaches exist for blog retrieval and they have

fairly good retrieval performanceBLOG. However, there are some aspect of the blog retrieval task that are not well-explored yet. We believe one of the main aspects of blogs that need to be studied better is related to the temporal information available in blogs. Temporal information can give us better understanding of queries and blogs and consequently can help us to have better retrieval systems. Beside the temporal information, other blog features like dependency between posts of a blog or the noisy nature of blogs have the potential to be investigated and probably help the retrieval performance. These observations directed us toward specific topics that we pursued in our research and we will discuss them in the next chapters.

Chapter 3

Experimental Methodology

In this chapter we describe test collections and evaluation measures that are used throughout the thesis.

3.1 Test Collections

Most of the work we carried out during this study was done on the TREC blog data sets that are available for blog distillation. These data sets include two blog collections and four query sets and we use them for evaluating our methods.

Blog collections include Blog06 and Blog08 collections. The Blog06 collection is a crawl of about one hundred thousand blogs over an 11-weeks period [MOS07], and it includes blog posts (permalinks), feed, and homepage for each blog. In order to have a real snapshot of blogsphere, organizers salect different type of blogs to include in the collection. About 70% of the blogs were top blogs of the web that were provided by a specialized blog search company. Spam blogs are another important part of the collection that make about 18% of total blogs. The remainder of the collection, i.e about 12%, are manually selected blogs that cover general topics like news, politics, sports, health, etc [MO06a]. Organizers crawl both XML feed for each blog and HTML permalinks for each post. The permalinks are crawled after two weeks of fetching the posts to collect also comments of the posts. However in our experiments we do not use comments and focus on the main text of the posts.

Blog08 is a collection of about one million blogs crawled over a year with the same structure as Blog06 collection [MOS09]. Organizers expanded the list of Blog06 collection by sampling new blogs from online blog directories or blog providers, following outgoing links of existing blogs or using blog search engines. The crawling method resembles that of Blog06 but for a longer period

Quantity	Blog06	Blog08
Number of unique blogs	100,649	1,303,520
Number of permalinks (posts)	3,215,171	28,488,766
First feed crawl	06/12/2005	14/01/2008
Last feed crawl	21/02/2006	10/02/2009
Total compressed size	25GB	453GB
Total uncompressed size	148GB	2,309 GB
Number of unique terms	4,968,020	41,980,546
Total number of terms	2,775,315,208	55,427,425,416
Avg. document length	863.19	1,945.58
Avg. blog length	27,574.19	42,521.34

Table 3.1. Statistics of the TREC blog collections.

of time over larger set of blogs.

In our experiments we only use the permalinks component of the collection, which consist of approximately 3.2 million documents for Blog06 and about 28.4 million documents for Blog08. Table 3.1 shows general statistics of the collections [MSOS10b].

In most of our methods we use temporal information for which we need publish date of posts. However, small percent of posts provide a reasonable publish date. In most of the cases, publish dates are either missing or contain meaningless dates, e.g dates around 1970. In order to have an approximation of publish dates, we use the crawling date of posts as an estimation of their publish date.

Query sets that we use for the evaluation include TREC'07, TREC'08, TREC'09 and TREC'10 query setsz. The TREC'07 and TREC'08 query sets include 45 and 50 assessed queries respectively and use the Blog06 collection. The TREC'09 and TREC'10 query sets use the Blog08 collection and have 39 and 46 queries respectively.[1] Figure 3.1 shows an example of TREC topics. We use only title of the topics as the queries. Table 3.2 shows the general statistics of the query sets.

[1]Initially there were 50 queries in TREC 2009 data set but some of them did not have relevant blogs for the selected facets and are removed in the official query set [MOS09]. We do not use the facets in our experiments however we use the official query set to be able to compare with the TREC results.

Figure 3.1. Example of a TREC topic.

```
<top>
<num> Number: 1151 </num>

<query> information warfare </query>

<desc> Description:
I am looking for blogs on information warfare and cyberwarfare.
</desc>

<facet> indepth </facet>

<narr> Narrative:
I want to find blogs about information warfare, attacks by
    organizations or governments on computer networks or sites, and
    related information.  Blogs on cybersecurity are relevant.
</narr>

</top>
```

Quantity	TREC'07	TREC'08	TREC'09	TREC'10
No. of queries	45	50	39	46
No. of assessed blogs	17,411	18,002	21,112	11,707
No. of relevant blogs	2,221	1,943	1,021	1,177
Avg. No. of rel. blogs per query	49.3	38.8	26.17	25.58

Table 3.2. Statistics of the TREC query sets.

3.2 Implementation Detail

The primarily programming language that we use for implementing most of our methods is Java. However, some of our scripts, mainly for evaluating systems, are implemented in Python and R.

Our main indexing and retrieval systems are based on the Terrier Information Retrieval syste[2] and its java libraries. Beside Terrier, we use other java libraries like Colt library[3] for matrix calculation and Mallet library[4] for text similarly and clustering methods.

We need to carry on few pre-processing steps that are known to be effective in information retrieval tasks. One of these preprocessing steps is stemming in which we reduce different derived words to their root form, e.g. reducing "driving" and "driver" to "driv". The other important preprocessing step is to remove common english words that do not carry any significant meaning and thus would not effect the relevance of a document, eg. removing "of" and "the". In our experiments, we use default stemmer and stop words list that are provided with the Terrier toolkit.

3.3 Basic Retrieval Method

Unless otherwise specified, the language modeling approach using the Dirichlet-smoothing has been used to score posts and retrieve top posts for each query [PC98; ZL04].

In this model, posts are ranked based on their probability of being generated with the same language model that has generated the query. By assuming a uniform prior distribution over the documents, the score of a document becomes the likelihood of the query given the parameters that are estimated by observing the documents:

$$\begin{aligned} P(d|q) &= P(d) \cdot P(q|d) \\ &\overset{rank}{=} P(q|\vec{\theta_d}) \\ &= \prod_{t \in q} P(t|\vec{\theta_d}) \end{aligned} \tag{3.1}$$

where θ_d is the language model of the document d. In other words, θ_d is a parameter vector estimated by observing each document. It is also worth noting

[2]Available at: http://ir.dcs.gla.ac.uk/terrier/
[3]Available at:http://acs.lbl.gov/software/colt/
[4]Available at:http://mallet.cs.umass.edu/

that query terms are assumed to be independent and thus the final probability can be rewritten as the multiplication of query terms probability. The independence assumption is not a valid assumption in the real world data, but it extremely simplifies the calculation and experiments show that this simplified method is fairly effective.

Each document model is assumed to have a multinomial distribution over terms. So, each document model has a parameter for each term of the vocabulary that shows the probability of observing that term in the document. Using a Maximum Likelihood Estimation (MLE), one can estimate the parameters based on the observed terms in the document that will be proportional to the term frequency, number of occurrence of the term in the document. However, since only small portion of the vocabulary terms are observed in each document, the MLE method would estimate zero probability for most of the terms.

The resulting zero probability for some of the terms, would give a final zero score to a document when the document does not contain one of the query terms. Thus different smoothing methods are proposed to overcome this problem. On of the most popular smoothing methods that we use also in our experiments is Dirichlet smoothing. By assuming a Dirichlet prior over the parameters, the method updates parameter values by observing each document and estimates Dirichlet posterior distribution with new Dirichlet parameters:

$$P(\vec{\theta_d}|\vec{\alpha_C}, \vec{\alpha_d}) = Dir(\vec{\theta_d}|\vec{\alpha_C} + \vec{\alpha_d}) \tag{3.2}$$

where $\vec{\alpha_C}$ is the vector of pseudo-counts that we have for each term based on the collection as a whole (our prior knowledge about term probabilities). The pseudo-counts are proportional to the real counts of terms in the collection and set to $\mu P(t|C)$; μ is the parameter of the model and $P(t|C)$ is the probability of term in the collection; $\vec{\alpha_d}$ are the counts that we have observed in the document. If a term does not occur in the document, the count will be zero.

Using the posterior distribution of the parameters, one can use the expected value of the parameters as the point estimation for that parameter and obtain the probability of each term in the document as the following:

$$P(t|\vec{\theta_d}) = \frac{tf(t,d) + \mu \cdot P(t|C)}{|D| + \mu} \tag{3.3}$$

where t is a term, $tf(t,d)$ is the number of time that term t occurs in document d and $|d|$ is the length of document. Having the point estimation of parameters and assuming independence between terms in a query, we can estimate the

likelihood of the query given the estimated parameters:

$$P(q|\vec{\theta_d}) = \prod_{t \in q} P(t|\vec{\theta_d}) = \prod_{t \in q} \left(\frac{tf(t,d) + \mu.P(t|C)}{|D| + \mu} \right) \tag{3.4}$$

Beside query similarity ti a document, in some of our solutions throughout the thesis we need to calculate the similarity between two documents. Unless otherwise specified, we use the cosine similarity between documents as our similarity measure. Cosine similarity assumes that each term is one dimension in our document representation space and each document is a vector in this multi-dimensional space. Based on this analogy, cosine similarity measures the cosine of the angel between two vectors representing documents [SWY75].

$$\text{sim}(d_i, d_j) = \frac{\mathbf{d_i} \cdot \mathbf{d_j}}{\left\|\mathbf{d_i}\right\| \left\|\mathbf{d_j}\right\|} = \frac{\sum_{k=1}^{N} tf_{k,i} tf_{k,j}}{\sqrt{\sum_{k=1}^{N} tf_{k,i}^2} \sqrt{\sum_{i=1}^{N} tf_{k,j}^2}} \tag{3.5}$$

where $tf_{k,i}$ shows the frequency of term k in document i. More similar two documents are, closer their vector representations are and thus higher their cosine similarity is. If two documents are identical, the angel between their vector representations is zero and cosine value is one. And if two documents do not have any term in common, their similarity will be zero. Since the term frequency in a document is never negative, the similarity value will always be positive and between zero and one.

3.4 Evaluation Methods

We use the blog distillation relevance judgements provided by the TREC organizers for evaluating our methods. Different query sets and relevance judgments are available for each year and general statistics of these query sets can be seen in table 3.2. These judgments are generated by TREC participants or NIST employees and are the standard evaluation set for the blog retrieval task. For each topic, top 100 retrieved blogs are returned as the result list and evaluated based on the relevance judgments.

We use standard IR evaluation measures for evaluating our proposed blog retrieval methods [CMS10]. The reported measures include Precision at 10 documents (P@10), Mean Average Precision (MAP) as well as binary preference (bpref). In the following we describe each of these measures in more detail.

There are two main measures for evaluating any retrieval system on which the rest of the measures are based. These two measures include precision and

recall. Precision is the portion of retrieved documents that is relevant. In other words, precision is the conditional probability that a document is relevant given it is retrieved:

$$\begin{aligned} precision &= P(relevant|retrieved) \\ &= \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents }\}|} \end{aligned} \tag{3.6}$$

The other important basic measure is recall that is the portion of relevant documents that is retrieved. Recall is the conditional probability that a document is retrieved given it is relevant:

$$\begin{aligned} recall &= P(retrieved|relevant) \\ &= \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents }\}|} \end{aligned} \tag{3.7}$$

Usually in information retrieval tasks, it is assumed that the first ranked items are the most important ones for the user. Based on this assumption, different evaluation measures are proposed that emphasize the precision in the high ranks of the ranked list. On of the most popular measures in this category that we also use in the thesis, is the precision at 10 [CMS10]. Precision at rank k (P@K) is defined as follows:

$$precision(k) = \frac{|\{\text{relevant documents}\} \cap \{\text{top K retrieved documents}\}|}{k} \tag{3.8}$$

In precision at 10 (P@10), k is set to 10. Similarly we can define recall at rank k (R@K) that would calculate recall in the top k retrieved documents.

There is tradeoff between precision and recall. Generally, P@K decreases with increasing k and R@K increases with increasing k.Thus we need measures that consider both precision and recall in the evaluation.

One of those measures is Average precision. Average precision (AP) is defined based on the $precision(k)$:

$$AP = \frac{\sum_{k=1}^{n} precision(k) \cdot rel(k)}{\text{number of retrieved relevant documents}} \tag{3.9}$$

where n is the number of retrieved documents and $rel(k)$ is an indicator function that is one if the document at rank k is a relevant document or otherwise it is zero. In other words, AP is the average precision at all ranks where a relevant document is retrieved. In other words, AP is the average of precision value at those ranks wherever recall value changes.

Finally, Mean Average Precision (MAP) is the average of AP over all queries.

$Bpref$ is the other commonly used measure that we report in some of our experiments. $Bpref$ models binary preferences between documents. The idea is that any relevant document is preferred over all non-relevant documents and our ideal ranking should retrieve the relevant documents higher than any non-relevant ones. Thus, the number of times that a system retrieves a non-relevant document before a relevant one can show how bad the system works. Based on this fact, the $bpref$ measure is defined as the following [BV04]:

$$bpref = \frac{1}{R}\sum_{r}\left(1 - \frac{|n \text{ ranked higher than } r|}{min(R,N)}\right) \tag{3.10}$$

Usually systems are evaluated over a set of queries where the average value over all the topics need to be calculated as the performance measure of the system.

Since the systems are compared over a set of queries, one system may highly improve performance for some random queries while slightly hurts the performance for the rest of the queries. In such a scenario, the system would have higher mean value over all the queries but its improvement is not robust. In order to test if the improvement over mean values is random or not, we use statistical test for comparing the performance of two systems. One of the common test that we employ in our experiments is Student's Paired T-test. In the Student's Paired T-test the goal is to compare the population mean of two sample sets where the null hypothesis is that their means are equal. In this scenario, performance of each system for each query is assumed a sample from a distribution. Since queries are same between two systems, the samples are paired which means the system performance for each query is compared with its counterpart from other system. The test examines the difference between every pair values and if such differences are very unlikely to happen by chance, it rejects the null hypothesis and concludes that the mean values are not equal.

In most of the methods there are parameters to be tuned based on a training-testing paradigm. We use n-fold cross validation technique for estimating the best value of parameters. In this technique, we first divide the available data into n different sets. Then for estimating the parameter value, we use $n-1$ set as the training set based on which we find the value that maximizes our target evaluation measure. Then we use the resulting value for testing the system on the last n-th set of the data. We repeat this process n time and we will have the result for all the data. This process enables us to use the best value of the parameter and at the same time not over-fit the parameter.

3.5 Conclusion

In this chapter we discussed about the standard test collections that are provided by TREC organizers for the blog retrieval task. We saw that there are different test collections available for this task that enable us to try our proposed methods and compare them with the stat of the art techniques.

We also introduced evaluation measures that are used for evaluating blog retrieval results. Since blog retrieval is similar to other information retrieval problems where the goal is to retrieve and rank items for a given query, thus one can employ standard IR evaluation measures in order to evaluate the results.

Chapter 4

Term Mismatch Smoothing

One of the main problems in information retrieval in general and in user-generated content in particular is the term mismatch between queries and documents. In this chapter we investigate the effect of content-based similarity between posts on overcoming this problem. We test two different approaches for smoothing (regularizing) relevance scores of posts based on their dependencies. In the first approach, we smooth term distributions describing posts by performing a random walk over a document-term graph in which similar posts are highly connected. In the second, we directly smooth scores for posts using a regularization framework that aims to minimize the discrepancy between scores for similar documents. We then extend these approaches to consider the time interval between the posts in smoothing the scores. The idea is that if two posts are similar and temporally close, they are good sources for smoothing each other's relevance scores. We compare these methods with the state of the art approaches in blog search and show performance gains over the baseline techniques which do not take advantage of the relation between posts for smoothing relevance estimates.

4.1 Introduction

In this chapter we investigate the term mismatch problem in the blog retrieval and adapt some smoothing methods to decrease its effect on retrieval. The term mismatch problem in blog retrieval is mainly due to the amount of noise in the blog content and in the generality of topics provided by users.

Blog posts are usually more spontaneous and less formal than classical information retrieval test collections like news wires. Thus spelling mistakes are more common in blog posts, which makes it harder to precisely estimate term

probability distributions. The high number of spam blogs and spam comments further intensifies this problem.

On the other hand, users' information needs in the blogosphere are different from those of usual Web users and queries are usually more general. Mishne and de Rijke compared blog queries to general Web queries and divided blog queries into two broad categories called context and concept queries [MdR06]. In the context queries, users are looking for Named Entities with special interest in new events. While in the concept queries they are looking for information related to one of their topic of interests. These concept topics are usually general and the user is interested in finding and subscribing to blogs that are mostly related to the topic [MdR06; MO08]. The goal of a blog feed search system is to answer the second type of queries and so the system should be able to match a general query with posts that discuss a specific aspect of that query, which is another type of the term-mismatch problem.

We investigate the effect of content-based similarity between posts on the performance of the retrieval system. We use these similarities as the complementary information to the original query to overcome the term mismatch and topic generality problems. In addition to using a graph-based regularization method to regularize the relevance scores [Dia05], we propose a general graph-based framework for taking into account the content-based relations between posts. We use this model to smooth the term probability estimation.

Another noteworthy aspect of blog distillation, which differentiates it from other IR tasks, is related to the temporal properties of blogs and queries. Since queries are very general, they can evolve over time, and at each specific time period different aspects of a query can be discussed in the blogosphere. By integrating temporal similarities between posts into the smoothing models, we explore the effect of time in the smoothing of relevance scores or term probabilities of posts.

The goal of this chapter can be summarized in answering the following questions:

- Is content-based similarity between posts useful for distinguishing between relevant and non-relevant blogs?

- Can we use content-based similarity between blog posts to smooth their relevance scores?

- Can we use content-based similarity to better estimate term probability distributions?

- Can we extend smoothing models to take into account temporal relations between posts?

In the rest of the chapter, we describe the term mismatch problem in blog search and our motivations for using content similarity to solve it. Later we describe the investigated smoothing methods, namely a random walk-based method for smoothing term probabilities and a regularization framework for smoothing relevance scores. we also discuss extensions of those smoothing methods that take into account temporal information. Finally, experimental results over three different blog data sets are discussed.

4.2 Related Work

We discussed the existing blog retrieval methods in the chapter 2. Since we employ graph-based methods for capturing the relation between posts, we briefly present a review of graph-based smoothing techniques in information retrieval and discuss their similarity to our work.

Similarity between documents has been widely used for re-ranking an initial ranked list of documents. Kurland and Lee create a graph by connecting a document to its top k similar documents [KL05]. They then use the PageRank algorithm [BP98] to estimate the centrality of each document in the retrieved set of documents and re-rank them based on their centrality score. Diaz proposes an optimization framework for regularizing relevance scores of initially retrieved documents [Dia05]. The model is based on the cluster hypothesis and tries to give similar scores to similar documents. Mei *et al.* use a similar approach and propose a general optimization framework for smoothing language models [MZZ08]. Crestani exploits the term similarity in the term space to solve the term-mismatch problem [Cre00]. All of the mentioned methods consider only one type of node in the graph and need a separate similarity function to calculate the similarity between two objects to be used as the weight of the edges in the graph.

Having multiple types of objects in a unified graph (matrix) representation has been considered in different works in information retrieval and recommendation systems. Robertson *et al.* [RMC82] propose a unified probabilistic model that considers both user information needs and documents to calculate the probability of relevance for a given document. Poblete *et al.* [PCG08] propose a unified graph representation for document ranking that includes documents and queries as two types of objects and covers both structural and usage information of the web pages. Similar work has been done by Craswell

and Szummer over a click graph of queries and images to improve image retrieval perfomance [CS07]. Serdyukov *et al.* model the expert finding task as a random walk in a graph that consists of candidates and documents as its nodes [SRH08a; SRH08b]. Clements *et al.* [CdVR10] use a tripartite graph that includes users, tags and items where edges show the relation between these nodes. They employ this model for different recommendation and retrieval tasks. Closely related to our work, Lafferty and Zhai use a Markov chain representation of documents and terms for the query expansion where they find that using small number of top-ranked documents in smoothing is helpful [LZ01]. Similarly, Shakeri and Zhai used document and term graph and found that smoothing can improve the high precision measures while it can hurt other evaluation metrics [SZ08].

In our models, beside using a regularization framework for smoothing relevance scores of posts, we propose a unified graph that includes both terms and documents as nodes. In this model, smoothed similarities between objects are calculated by a random walk process and there is no need for a separate similarity function.

4.3 Document Dependency in the Blogosphere

None of the blog retrieval techniques described in chapter 2 have taken the dependency between posts from different blogs into account when calculating query relevance scores for blogs. In this section, we investigate whether the dependency between posts both within and across blogs can be used to improve the blog retrieval performance. First we consider the content-based relation to smooth document language models. Then we extend it by considering the temporal-closeness of posts as another relation and employing it in smoothing.

4.3.1 Term Mismatch and Content Similarity

As stated previously, the goal of blog distillation is to retrieve blogs with recurrent interest in a given topic. The large number of bloggers in the world and the variety of their writing style and vocabulary make this problem challenging, especially since blog posts tend to be less formal and their intention is usually less obvious than classical IR test collections.

Term mismatch and spelling mistakes are common in blog posts, which make it hard to precisely estimate the term probability distributions. For example, in the Blog08 test collection there are more than 40 million unique terms, more

than 60% of which occur only once in the whole collection and more than 99% of them are not in a standard dictionary.[1] In comparison the Wall Street Journal (WSJ) collection, a typical newswire corpus, has 34.3% singleton terms and 67.5% out of dictionary terms [ICC10]. It is obvious that these numbers are much higher in blogs than in other types of text, like news, and it is one of the main reasons for sparse term distribution and high term mismatch ratio. At the same time, topics in blog search are usually general and multifaceted. This makes the term mismatch more problematic and causes a wide vocabulary gap between the query and the relevant documents [EACC08].

Due to the generality of topics and the ambiguity in blogosphere, it is hard to extract relevance information from a post. However, for a specific post we can usually find a set of closely related posts among other blogs. These related posts can be seen as complementary information about the content of the post and can be utilized to create a better representation of the post, overcoming to a certain extent, the term mismatch problem.

In order to examine if the similarity between posts can provide more information than the original query for retrieving relevant blogs, we performed some preliminary analysis on relevant and non-relevant blogs extracted from the TREC'09 relevance judgments [MOS09].

First we compared the language model generating probability, as a simple relevance score, between posts from relevant and non-relevant blogs for each query. Figure 4.1 shows the average post relevance scores from relevant blogs compared to the value of non-relevant blogs for each query. It is interesting to that they are not distinguishable and for some queries non-relevant blogs have higher average post relevance scores than relevant ones. This means that the original query might not be enough to distinguish between relevant and non-relevant blogs, and thus we need to use some additional information to improve retrieval. It is worth noting that while the query likelihood of posts is not the only component of the blog retrieval models, and other features like length of the blog or blog cohesiveness also have an effect, it is the main part of all models.

Next we compare the similarity between posts from two relevant blogs against the similarity between posts from a relevant blog and a non-relevant blog. Figure 4.2 shows the average similarities for each query over the same set of posts analysed in Figure 4.1. It shows that for all the queries, the average similarity between posts from relevant blogs is higher than the average similarity between posts where one comes from a relevant blog and the other comes

[1] For this analysis we used *New Oxford American Dictionary* in order to check if a term exists in the dictionary or not.

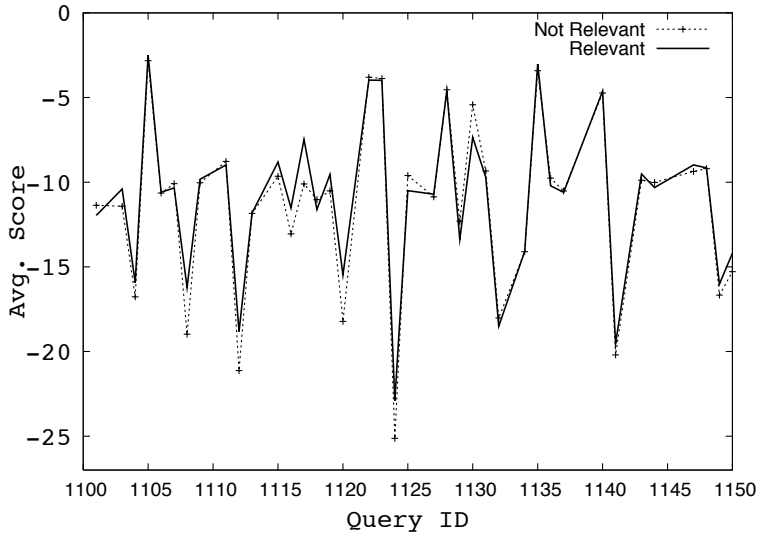

Figure 4.1. Average score of high ranked posts in relevant and non-relevant blogs for each query.

from a non-relevant one. Thus, while the average posts score for relevant blogs is not consistently higher across the topics compared to the non-relevant ones, the posts from relevant blogs do have consistently higher similarity to each other than to the posts from non-relevant blogs. This confirms our intuition that content relations between posts can help us to distinguish between a relevant and a non-relevant blog.

The idea of using the content relations between posts can be seen as similar to query expansion techniques such as pseudo-relevance feedback. However, it has been shown that simple query expansion techniques using the top retrieved blogs or top retrieved posts are not effective in the blog retrieval task and do not improve the performance of the system [EACC08]. Thus it remains an open challenge to use the content-based relation effectively in the blog retrieval context.

4.3.2 Smoothing Term Probabilities using Random Walks

One way to make use of the relationships between posts is to smooth the term probability distributions for each post by taking into account the number of

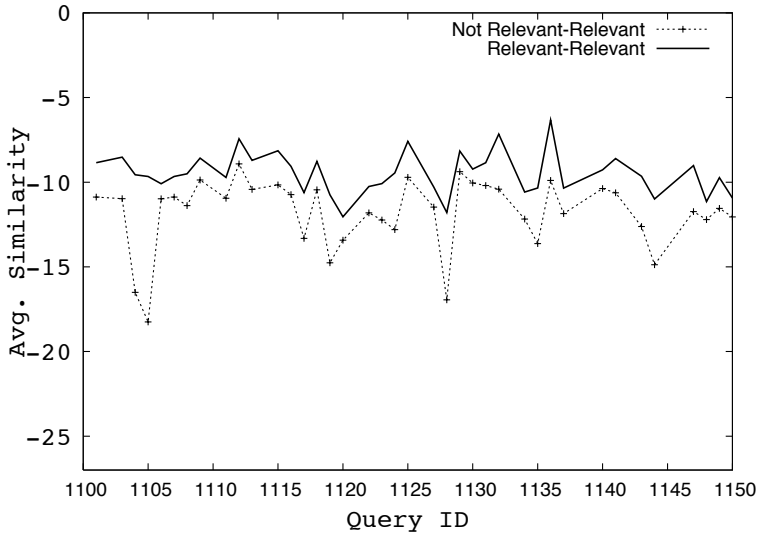

Figure 4.2. Average similarity between high ranked posts from relevant blogs and high ranked posts from relevant vs. non-relevant blogs.

terms it has in common with other posts. The more common terms there are between two posts, the more similar their term probability distributions should be.

We propose a graph-based representation of the most relevant posts for a particular query and their terms which enables us to model the content-based dependency between posts. Figure 4.3 shows part of such a graph where an edge between a post and a term indicates that the term occurred in that specific post. Weight of edges are calculated based on the frequency of a term in a document and is described in the following.

One can record this relation in a transition matrix A. In this matrix, A_{ij} (element of row i and column j in the matrix A) denotes the transition probability from node i to node j in one step of a random walk, i.e. $P(t_j|p_i) = A_{ij}$ shows the probability of a term for a given post (transition probability from a post to a term) and $P(p_i|t_j) = A_{ji}$ shows the probability of a post for a given term (transition probability from a term to a post). The outgoing probabilities from a node and hence the values in any row of A add up to one, i.e. $\sum_j A_{ij} = 1$. It is worth noting that the matrix A is a square matrix in which the number of rows (columns) is equal to the total number of posts and terms.

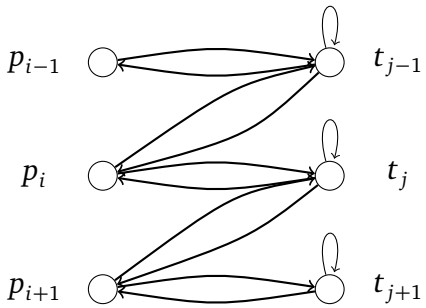

Figure 4.3. Example of a post-term graph.

We will use the notation $P_n(t_j|p_i) = (A^n)_{ij}$ to denote the transition probability from post p_i to query term t_j in n steps. Performing this random walk can be seen as a type of smoothing where we compute probability estimates even for terms not present in the document. This graph-based smoothing takes into account the frequency of each term in similar documents, where similar documents are determined by their common terms. For example, even if a particular blog post discussing some aspect of machine learning did not contain the word "regularization", the term would be assigned a non-zero value within a smoothed term probability distribution for the post, since other posts discussing machine learning would likely contain this term with high frequency.

Importantly, whenever a term occurs in a smoothed document model, the random walk can explain its presence and the probability value assigned. An obvious benefit of this approach is that we do not need to calculate similarity between all pairs of posts as done in the other graph-based frameworks, since the similarity is calculated implicitly during the random walk process.

Since we only need to compute $P_n(t_j|p_i)$ for terms in the query in order to calculate a score for the post, we can efficiently calculate this value, as indicated by Craswell *et al.* [CS07], by iterating forward from *each* query term node t_j to *all* post nodes p_i concurrently, as follows:

$$P_n(t_j|p_i) = (A^n)_{ij} = (A(...(A(A\mathbf{j})))_i \tag{4.1}$$

where $\mathbf{j}$ is a unit (column) vector with value 1 in the j-th row. Note that this calculation only needs to be performed once per query term. The transition probabilities between terms and documents are calculated using Maximum Likelihood estimate:

$$P(t_j|p_i) = \frac{\text{tf}(t_j, p_i)}{|p_i|} = \frac{\text{tf}(t_j, p_i)}{\sum_k \text{tf}(t_k, p_i)} \tag{4.2}$$

where $\text{tf}(t_j, p_i)$ is the frequency of the term t_j in the post p_i and $|p_i|$ shows the size of the post which is the total number of terms in it. Similarly, the probability of transitioning from a term node t_j to a post node p_i is defined using Bayes' Theorem by:

$$P(p_i|t_j) = \frac{\text{tf}(t_j, p_i)}{\sum_k \text{tf}(t_j, p_k)} \tag{4.3}$$

This formula can be derived directly from equation 4.2 by assigning proper prior probabilities for posts and terms. we set the prior probability of a post to be proportional to its size:

$$p(p_i) = \frac{|p_i|}{\sum_k |p_k|} \tag{4.4}$$

and probability of a term is set to be proportional to its frequency in the collection:

$$p(t_j) = \frac{\sum_k \text{tf}(t_j, p_k)}{\sum_k |p_k|} \tag{4.5}$$

Using these prior probabilities in Bayes' Theorem equations 4.2 and 4.3 can be derived from each other.

A random walk of length one is equivalent to a Maximum Likelihood estimate for $P_n(t_j|p_i)$, while an infinite random walk would generate a stationary probability distribution independent of the starting points. It is worth noting that we do not necessary need a stationary distribution in our calculation. The fact that after few steps of random walk we have smoothed estimation of term probabilities is enough for our purpose in this framework. In the following, we describe how with adding self loops to terms, we keep the dependency of the probabilities to the initial nodes.

By adding a self-loop transition to all term nodes, we turn a length n random walk into the weighted (exponentially decaying) average of walks of length 1 to n. One can do this by setting $\alpha = P(t_j|t_j)$ as the "self-loop" probability on the term nodes. This is a smoothing parameter which regulates the importance of shorter versus longer walks in the post-term graph. Thus the parameter α regulates how much smoothing is done on the initial Maximum Likelihood estimate. The smaller the self-loop probability α, the more the estimate will be smoothed with longer walks and vice versa. Similar behavior could be obtained using more advanced random walk techniques like personalized page rank in which closer nodes to the query terms would have higher importance [JW03]. However we leave that study and comparison for the future work.

To see the effect of α in the final probabilities in the random walk, assume we calculate a walk of length 20 in a graph without self-loop. At the end of this

calculation, we have $P_{20}(t_j|p_i)$ for all terms and posts. However the calculated probabilities do not have any information about probabilities in shorter walks and we miss that information. Thus, for example it is possible that the lengths of all paths between a post and a term are odd, so P_{20} for that post and term will be zero, while a shorter path of length 19 might have a non-zero value. By adding the self-loop on the terms, we can keep a history of all walks to that term.

Another important point is to note is about the final probability values that we assign to terms and documents. Our final conditional probabilities would be a distribution over all terms and documents and would sum up to one. In our scenario, we do not use the portion of probabilities that is between two terms or between two documents. In order to have a proper probability distribution, we would need to re-normalize the final values over the subset of nodes that are used in the calculation. But since our goal is to have a ranking of documents and we do not need the precise estimation of values, thus the final normalization step is not necessary in our application.

Finally, to make the matrix stochastic (where each row sums to one) given the term self-loops, we need to decrease the probabilities from terms to posts by a factor of $(1-\alpha)$. The generated matrix looks as follows:

$$A = \begin{bmatrix} 0 & M_{PT} \\ (1-\alpha)M_{TP} & \alpha I \end{bmatrix} \tag{4.6}$$

Here P indicates the posts, T the terms, and M_{XY} is a stochastic sub-matrix with transition probabilities from the object type X to the object type Y.

Once we have computed $P_n(t_j|p_i)$ for each term in the query, we can use these values (after further smoothing with a collection model $P(t_j|C)$) to calculate an estimate for the query likelihood given the post:

$$P_{RW}(Q|p_i) = \prod_{t_j \in Q} \lambda P_n(t_j|p_i) + (1-\lambda)P(t_j|C) \tag{4.7}$$

It is worth noting that the random walk based framework for smoothing term distributions is very general and can easily be extended with new forms of evidence such as hyperlinks between posts which can be encoded directly as additional edges into the post-term graph.

4.3.3 Blog Post Scores Regularization

Score regularization is a way to re-calibrate relevance scores of documents based on the relationship between them. The idea behind score regularization is that

in accordance with the Clustering Hypothesis, *related documents should have similar scores for the same query*. Multiple models for smoothing document scores were proposed based on this hypothesis [Dia05; MZZ08]. Diaz models the problem in terms of a regularized optimization problem on a document similarity graph [Dia05]. The goal is to calculate for each document a new (smoothed) score with two contending objectives: score consistency with related documents and score consistency with the initial retrieval score. He defines an overall cost function $\zeta(f)$ as follows:

$$\begin{aligned}\zeta(f) &= \sigma(f) + \mu\varepsilon(f) \\ &= \sum_{i \neq j} (w_{ij} f_i - w_{ji} f_j)^2 + \mu \sum_i (f_i - y_i)^2 \end{aligned} \tag{4.8}$$

where f is a vector of regularized scores over n documents, $\sigma(f)$ is a cost function associated with the inter-document consistency of the scores. High values of $\sigma(f)$ indicate inconsistent scores between related documents. A second cost function $\varepsilon(f)$ measures the consistency with the initial scores; if document scores are inconsistent with the initial scores, the value of this function will be high. A regularization parameter μ controls the trade off between inter-document smoothing and consistency with the initial score vector y. The coefficient w_{ij} in the expansion of $\sigma(f)$ weights the score of the ith document by its similarity to the jth document and is calculated by normalizing and taking the square root of values from a symmetric affinity matrix W as follows:

$$w_{ij} = \sqrt{\frac{W_{ij}}{\sum_j W_{ij}}}. \tag{4.9}$$

Here W_{ij} denotes the similarity between documents i and j. In order to keep the affinity matrix sparse, only the k most similar documents for each document i have a non-zero W_{ij} value[2]. The diagonal values in the matrix W are defined to be zero. An iterative solution for this optimization problem is the following:

$$f^{t+1} = (1 - \alpha) y + \alpha \bar{W} f^t \tag{4.10}$$

where $\alpha = 1/(1+\mu)$ is a parameter, $y = f^0$ is the initial score vector, f^t is the score vector after t iterations and $\bar{W}$ is a normalized affinity matrix such that $\bar{W}_{ij} = w_{ij} w_{ji}$. The closed form solution of this problem is given by:

$$f^* = (I - \alpha \bar{W})^{-1} y \tag{4.11}$$

[2]Although some documents may need to have more than k non-zero affinity values in order to keep the matrix symmetric.

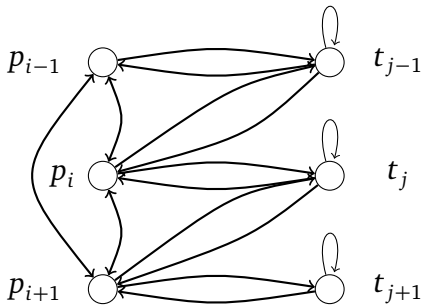

Figure 4.4. Example of a post-term graph with temporal relations.

We will use this equation in our experiments where we apply graph-based regularization frameworks [Dia05; MZZ08] to the problem of blog distillation.

4.3.4 Smoothing with Temporal Relations

In another approach to smoothing, we tried to use the temporal relation between posts for smoothing their scores. As mentioned before, several studies have considered time as an additional feature in blog retrieval demonstrating the importance of time in the blogosphere [ETO07a; EWdR07; WBdR08]. Also other works have shown a relationship between blogs and news [SHL08; MdR06]. They show that some news-related topics are bursty in the blogosphere and are discussed more in specific time intervals. Based on these observations, we consider query-related posts with less temporal distance as more dependent than posts that are temporally far away from each other. We can integrate these dependencies as another source of smoothing in our models.

We define the temporal similarity measure as a decay function:

$$Sim_{temporal}(p_i, p_j) = e^{-\frac{\Delta d}{\sigma}} \tag{4.12}$$

where Δd is the temporal distance in days and $\frac{1}{\sigma}$ is the decay constant.[3] The temporal transition probabilities are defined as follows:

$$P_{temporal}(p_j|p_i) = \frac{Sim_{temporal}(p_i, p_j)}{\sum_k Sim_{temporal}(p_i, p_k)} \tag{4.13}$$

To add the temporal relations to the post-term graph in the random walk model, we add new edges between each pair of posts based on their temporal distance.

[3]We tested other decay functions for temporal similarity such as the Gaussian kernel but experiments showed a simple exponential decay to be the best temporal similarity function.

Figure 4.4 shows the format of the new graph. We assign the weight of new edge as:

$$P(p_j|p_i) = \beta \; P_{temporal}(p_j|p_i) \tag{4.14}$$

where $P_{temporal}(p_j|p_i)$ is calculated by equation 4.13 and $\beta < 1$. we change the weight from posts to terms as the following, to keep the weights of out-links from each post summing to 1:

$$P(t_j|p_i) = (1-\beta)\,\frac{\mathrm{tf}(t_j,p_i)}{\sum_j \mathrm{tf}(t_j,p_i)} \tag{4.15}$$

The newly generated matrix will look like the following:

$$A = \begin{bmatrix} \beta \; M_{PP} & (1-\beta)\; M_{PT} \\ (1-\alpha)\; M_{TP} & \alpha \; I \end{bmatrix} \tag{4.16}$$

We note that there is a new sub-matrix M_{PP} with respect to equation 4.6 with transition probabilities from posts to posts as described above.

We also add the temporal relations to the regularization framework by changing the affinity function used in creating the affinity matrix. The new similarity measure is:

$$Sim(p_i,p_j) = Sim_{temporal}(p_i,p_j)\; Sim_{content}(p_i,p_j) \tag{4.17}$$

We use this new similarity as W_{ij} in the affinity matrix and perform regularization of post scores as before.

4.3.5 Computational Complexity

An important issue when comparing the two smoothing methods described above is their scaling behavior and whether their complexity can be reduced through approximation.

For regularization, the algorithm must first compute an affinity matrix, for which we need to compute similarity between every pair of documents, and then perform a matrix inversion operation, which results in a complexity of $O(n^2v + n^3)$, where n is the number of documents and v is the average vocabulary of the documents. If we approximate the affinity matrix using a hierarchical clustering approach [PLR07] and use the iterative method rather than the closed form to calculate a solution, then the complexity of the algorithm reduces to $O(n\log(n)v + skn)$, where s is the number of iterations, and k is the number of non-zero elements per document in the affinity matrix.

For the random walk computation, we need to multiply a sparse matrix by a vector l times, where l is the length of the walk and repeat that process for each term in the query (q times), resulting in a complexity of $O(qlvn)$, since there will be vn non-zero elements in the transition matrix.

Thus in terms of complexity, the random walk and the regularization algorithms are almost equally efficient compared to one another depending on the length of the query, the average vocabulary of posts, the length of the random walk, and number of the regularization steps required for convergence.

4.4 Experimental Results

In order to evaluate the methods, we use three years worth of TREC blog track data from the blog distillation task, including TREC'07, TREC'08 and TREC'09 data sets. Our goal is to examine if considering the similarity between blog posts can help us to improve the performance of retrieval systems. Further we want to test if the temporal similarity can add more information to content similarity and help us in retrieving blogs. To this end, we use the proposed smoothing techniques to estimate the new relevance score for each post based on its similarity to other retrieved posts. After smoothing the term probabilities using a random walk or regularizing post scores in a regularization framework, we use these new scores in different baseline methods for calculating blog relevance scores. We compare the performance of different baseline techniques before and after smoothing and show that smoothing can indeed have positive effect in different settings.

4.4.1 Parameter Settings

We use the Terrier Information Retrieval system[4] to index the collection and retrieve documents. Previous experiments show that working just with the most relevant posts (according to a retrieval score) improves the performance and requires less computational effort [LNK+08]. For this reason, we select the top 2000 posts for each query using the query likelihood probability and apply our smoothing methods on them.

Since we have multiple years worth of data, we considered each year as a separate test collection. For the parameters of the models, we train the values for each year using the other year data sets as training data and tuned the parameters to optimize each evaluation measure separately. This included tuning

[4] http://ir.dcs.gla.ac.uk/terrier/

language model smoothing parameter, α in equation 4.11 for regularization and α and β in equations 4.6 and 4.16 for term self-loop and temporal similarity importance. For all these parameters, we search for values between 0 and 1 at intervals of 0.1.

For the regularization and random walk based smoothing methods, we create the corresponding affinity and transition matrices using the top 2000 retrieved posts. After smoothing/regularization post scores, we calculate blog scores by the small document model (SDM) using the new post scores. The SDM method is discussed in section 2.3. Beside SDM as one of our baseline, our experiments show that uniform estimations of $P(p_{ji}|b_i)$ over posts of each blog performs better and is easier to calculate than the original centrality score of the posts proposed by Elsas *et al.* [EACC08]. In the new model that we use for calculating blog scores, called SDM-uniform, the final relevance score of a blog is calculated as follows:

$$score_{\texttt{SDM-uniform}}(q, b_i) = \frac{log(N_{b_i})}{N_{b_i}} \sum_{j=1}^{|b_i \cap R(q)|} P(q|p_{ji}) \tag{4.18}$$

where N_b is the number of posts in the blog b. We assume that the query likelihood of those posts that are not retrieved in $R(q)$ is equal to zero and thus we do not consider them in the summation. Similar estimations are used by Balog *et al.* [BdRW08].

In order to generate a kNN affinity matrix for the regularization method, we use the cosine similarity as a simple and effective similarity metric, that has been shown to be successful in graph based Language Model smoothing [MZZ08]:

$$sim(p_1, p_2) = \frac{\sum_t \text{tf}(t, p_1)\ \text{tf}(t, p_2)}{\sqrt{\sum_t \text{tf}(t, p_1)^2}\sqrt{\sum_t \text{tf}(t, p_2)^2}} \tag{4.19}$$

For each post, we select the average of its similarity with the other posts as the similarity threshold. We add to the matrix all posts with similarities higher than the threshold and for the rest we put zero.

Before generating the transition matrix for the random walk-based smoothing, we discarded terms with very high document frequency (more than 80% of the documents) or very low document frequency (less than 5 documents) in order to reduce the size of the graph. As with most of the parameters, we set the term self-loop probability α by learning from the training data. The length of the random walk is set to be long enough to approach the stationary distribution, our experiment showed that the length of 20 was sufficient for all the generated graphs.

Model	MAP	P@10	Bpref
ExpMNZ	0.2649	0.4444	0.3011
ExpSum	0.2570	0.4422	0.2880
SDM	0.2580	0.4356	0.2899
SDM-Uniform	0.2905	0.5040	0.3322
ExpMNZ-RW	0.2541	0.4133	0.2930
ExpSum-RW	0.2576	0.4222	0.2963
SDM-RW	0.2581	0.4289	0.2968 †
SDM-Uniform-RW	0.3029 †	0.5067 †	0.3484 †
ExpMNZ-Reg	0.2637	0.4311	0.2980
ExpSum-Reg	0.2639	0.4489	0.2963
SDM-Reg	0.2660 †	0.4489 †	0.2987 †
SDM-Uniform-Reg	0.3039 †	0.5156 †	0.3448 †

Table 4.1. Evaluation results of smoothing methods over TREC'07 query set.

4.4.2 Results

Tables 4.1, 4.2, and 4.3 show the results for the different methods on the TREC'07, TREC'08 and TREC'09 data sets respectively. In order to test for statistical significance, we use the Student's Paired T-test on scores for each query at the 0.05 level. Statistically significant improvements over corresponding baseline are shown by †.

We can see that smoothing methods outperform the baseline methods in MAP across different data sets. In most of the cases this improvement is statistically significant. For Precision at 10, there are improvements in some cases but the improvements are not consistent across all data sets. In the Bpref measure, the random walk method performs better than the regularization method in most cases. We interpret the better performance of the smoothing methods to mean that blog search retrieval functions that take the similarity between posts into account can indeed improve performance over those methods that do not.

It is interesting to see that SDM-uniform performs the best among all methods without smoothing. This supports the intuition for prior probabilities in this model, i.e with the same relevance evidence the shorter of two blogs is more likely to be relevant.

In the second part of our experiments we investigate temporal relations between posts for smoothing the scores. In the calculation of temporal similarities between posts, equation 4.12, we use the average temporal distance between the retrieved posts for each query as the σ value ($1/\sigma$ is the decay constant).

Model	MAP	P@10	Bpref
ExpMNZ	0.1754	0.2960	0.2205
ExpSum	0.1724	0.2980	0.2164
SDM	0.1740	0.3020	0.2171
SDM-Uniform	0.2197	0.3320	0.2705
ExpMNZ-RW	0.1700	0.2840	0.2176
ExpSum-RW	0.1824 †	0.2960	0.2238
SDM-RW	0.1809	0.3000	0.2235
SDM-Uniform-RW	0.2288 †	0.3400	0.2830 ‡
ExpMNZ-Reg	0.1684	0.2780 †	0.2163
ExpSum-Reg	0.1722	0.3060 †	0.2171
SDM-Reg	0.1719 †	0.3020	0.2155
SDM-Uniform-Reg	0.2264 †	0.3420	0.2706

Table 4.2. Evaluation results of smoothing methods over TREC'08 query set.

Tables 4.4, 4.5 and 4.6 show the performance of systems after adding the temporal relation between posts. As we can see, adding the temporal similarity to the smoothing methods improves the performance of the systems in most cases. This means that posts that occur within a shorter time interval from one another are better sources of information for smoothing language models than posts that occur far from each other in time.

Figures 4.5, 4.6, and 4.7 show interpolated Precision versus Recall graphs for the three data sets. The graphs compare the two similarity-based smoothing techniques (RW-Temporal and Reg-Temporal) with the best of the Language Modeling techniques (SDM-Uniform). Again we can see consistent improvements over the best baseline method (SDM-Uniform) across different levels of Recall for different data sets.

Comparing the random walk (RW) and regularization (Reg) techniques against each other, we see that the two techniques are very much comparable in terms of overall performance. Regularization appears to slightly outperform the random walk technique in terms of MAP, but the situation is reversed for Bpref. Since the difference in the performance of the two techniques is not large we recommend that the choice between the techniques be made based on the computational complexity issues previously mentioned, where factors like the average vocabulary of posts and queries need to be taken into account.

The smoothing method can have longer run time compared to simple retrieval method. Three main factor that influence the runtime of the smoothing

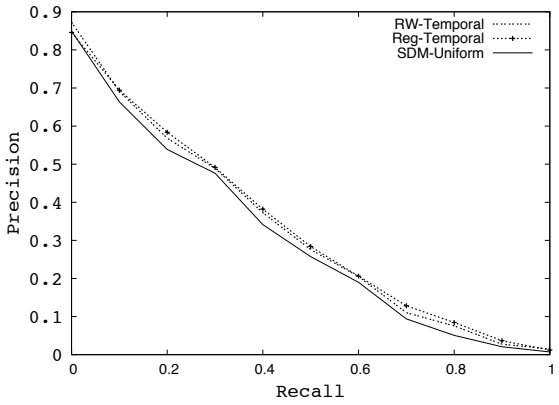

Figure 4.5. Precision-Recall for TREC'07 data set against best baseline method
.

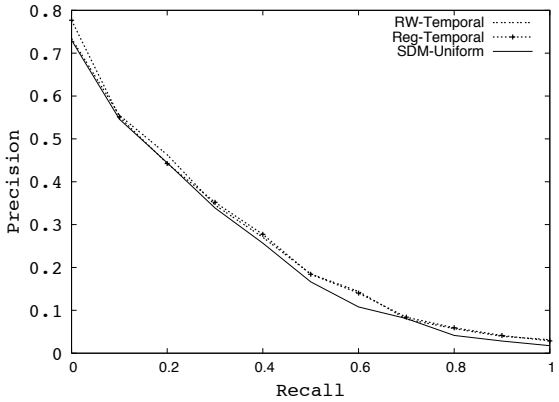

Figure 4.6. Precision-Recall for TREC'08 data set against best baseline.

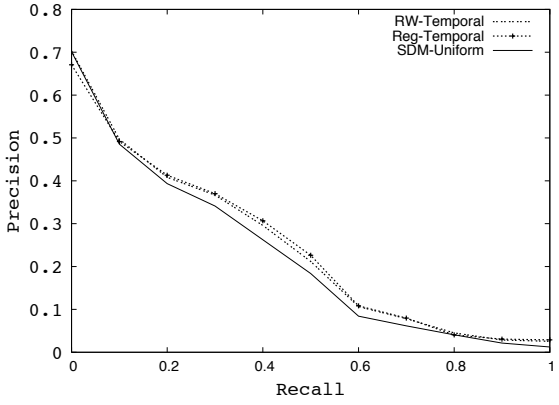

Figure 4.7. Precision-Recall for TREC'09 data set against best baseline.

Model	MAP	P@10	Bpref
ExpMNZ	0.1814	0.3077	0.2031
ExpSum	0.1643	0.2923	0.1942
SDM	0.1682	0.2974	0.1967
SDM-Uniform	0.2046	0.3410	0.2365
ExpMNZ-RW	0.1812	0.3000	0.2017
ExpSum-RW	0.1717	0.2974	0.1978
SDM-RW	0.1741 †	0.2974	0.1984
SDM-Uniform-RW	0.2131 †	0.3436 †	0.2421 †
ExpMNZ-Reg	0.1804	0.3051	0.1983
ExpSum-Reg	0.1814 †	0.3077 †	0.1997
SDM-Reg	0.1809 †	0.3077	0.1981
SDM-Uniform-Reg	0.2213 †	0.2974	0.2363

Table 4.3. Evaluation results of smoothing methods over TREC'09 query set.

Model	MAP	P@10	Bpref
SDM-Uniform	0.2905	0.5044	0.3322
SDM-Uniform-RW	0.3008	0.5000	**0.3454**
SDM-Uniform-RW-temporal	**0.3039**	0.5022	0.3448
SDM-Uniform-Reg	0.3039	0.5156	0.3434
SDM-Uniform-Reg-temporal	0.3028	**0.5200**	0.3429

Table 4.4. Evaluation results for temporally smoothing methods over TREC'07 query set.

algorithm is the number of selected documents for smoothing, number of the terms of selected documents and length of the random walk in the graph. Although the run time can be long, by limiting the influential factors we can substantially decrease it. Thus we limit the number of documents to 2000 and we remove the terms with very high (more than 85% of documents) and very low document frequency (less than 1% of documents). And also we limit the length of the random walk to 20 steps that would ensure us that it is long enough to reach related documents and on the other hand it is not very long for computational expenses.

Finally, one important issue for graph-based smoothing is the proportion of relevant and non-relevant documents in the graph. If the percentage of relevant documents in the graph is very low, the result of the smoothing might go toward the non-relevant documents and score them higher than the relevant ones. To

Model	MAP	P@10	Bpref
SDM-Uniform	0.2197	0.3320	0.2705
SDM-Uniform-RW	0.2288	0.3300	0.2768
SDM-Uniform-RW-temporal	**0.2343**	0.3440	**0.2833**
SDM-Uniform-Reg	0.2264	0.3420	0.2706
SDM-Uniform-Reg-temporal	0.2274	**0.3460**	0.2702

Table 4.5. Evaluation results for temporally smoothing methods over TREC'08 query set.

Model	MAP	P@10	Bpref
SDM-Uniform	0.2046	0.3410	0.2365
SDM-Uniform-RW	0.2131	**0.3436**	0.2409
SDM-Uniform-RW-temporal	0.2194	0.3308	**0.2431**
SDM-Uniform-Reg	0.2213	0.2974	0.2363
SDM-Uniform-Reg-temporal	**0.2268**	0.2949	0.2388

Table 4.6. Evaluation results for temporally smoothing methods over TREC'09 query set.

test if this was the case in our experiments, we calculated the Kendall τ correlation between the improvement for each query and the number of relevant blogs for that query amongst the relevance judgments. Table 4.7 shows the correlation coefficients for Reg-Temporal and RW-Temporal over SDM-uniform for the different data sets. We see that in all cases the values are close to zero and none of them is statistically significant. The low correlation coefficients indicate that the performance improvements are not correlated with the number of relevant blogs for each query. Thus, the graph-based smoothing can be effective even when the number of relevant blogs is low.

Model	TREC'07	TREC'08	TREC'09
SDM-RW-Temporal	0.03	0.03	-0.14
SDM-Reg-Temporal	0.11	0.04	-0.08

Table 4.7. Kendall τ correlation coefficient between improvement and the number of relevant blogs for queries

4.5 Conclusion

In this chapter we investigated whether the similarity between posts both in and across blogs could be used to smooth post relevance scores and thereby improve blog rankings. We tested two different approaches for smoothing relevance scores, one based on a regularization framework that aims to minimize the discrepancy between scores for similar documents and another based on a random walk algorithm that smooths the term distributions used to represent posts. We compared the methods with state of the art approaches in blog search that employ language modeling based resource selection algorithms and fusion-based methods for aggregating post relevance scores. We found that the smoothing methods performed better than the baseline techniques. There was, however, no significant differences in performance between the two smoothing techniques across the test sets.

We considered the temporal distance between posts as an another source of smoothing and showed that it can improve the performance of the system.

Chapter 5

Temporal Pseudo Relevance Feedback

5.1 Introduction

An important aspect of blog distillation, which differentiates it from other IR tasks, is related to the temporal properties of blogs and topics. The temporal evolution can be seen from two different point of views: Topics or Bloggers. Topics can evolve over time and have different sub-topics at each point of time. On the other hand, bloggers can change their interests and consequently change the topics of their blogs over time. To consider these properties in our ranking method, we propose different approaches based on pseudo relevance feedback techniques that try to extract useful time-related features and use them in ranking the blogs. In the first approach we use time as a new influential variable in a query expansion framework that can affect the term selection criteria. In the second approach, we extend the expansion framework and generate multiple expanded queries based on time.

Most of the temporal analysis of blogs have focused on the recency of blog relevance with respect to a given topic. These models give higher scores to more recent posts before aggregating them. Experiments of such models show some improvement over the baseline which uses only the content of the blog [EWdR07; WBdR08].

Nunes *et al.* use temporal evidence as an extra feature of blogs beside their content [NRD08]. They use the temporal span and the temporal dispersion as two measures of relevance over time, and show that these features can help in blog retrieval. In the most similar to our work, MacDonald and Ounis try to capture recurring interests of blogs over time [MO08]. Following the intuition that

TopicID	Topic	Assortativity	Description
1130	Fly Fishing	0.87	The latest fishing equipment and regulations ....
1136	Genome sequences	0.40	Newly sequenced genomes, ...
1119	No Child Left Behind	0.34	News of the "No Child Left Behind" program ...
1116	Homeopathic medicine	0.07	Homeopathic medicine and alternative ...
1109	Coin collecting	0.06	Business aspect of collecting and selling coins.
1105	Parenting	0.00009	Parenting, raising, or caring for children.

Table 5.1. The three most and three least assortative topics.

a relevant blog will continue to publish relevant posts throughout the timescale of the collection. Although they try to capture the recurring interest of bloggers, their method fails in some cases, e.g if all posts of a blog are relevant to the query.

Non of the previous method considered how up-to-date is a blog in any given point of time. Our intuition is that a relevant blog should publish up-to-date posts so that it covers popular subtopic at any given time. Based on this intuition, we propose our time-based query expansion technique that we describe in the rest of this chapter.

5.2 Relation Between Time and Content of Blog Posts

In this section we try to quantify the relation between blog content and its publish time. We are going to show that blogs content are time dependent, however the different topics have different behavior with respect to time. Previous work model the temporal behavior of queries based on the number of published documents at different times [DJ04]. They do not take into account the content of documents. We try to show that for time-dependent topics, time and content of posts are related. In other words, for those queries, closely published posts are likely to have similar contents. This provides us with a new way of categorizing queries and have different retrieval strategies for time-dependent or time-independent queries like having time-based retrieval models or time-based query expansion techniques.

By having time and content of posts as two features, we can cluster the posts based on each of these features. Then calculating the correlation between two types of clusters shows us how strongly they are related.

For clustering blog posts based on their content, we use the K-Means method with cosine similarity as the similarity measure [Mac67]. We use the Mallet implementation of K-Means for content-based clustering [McC02].

For clustering blog posts based on their publish date, we can divide the time-span of the collection into fixed number of windows and assume each window to be one cluster. However in this way of clustering, two posts that have been published close to each other might end up in different clusters. To avoid this problem, we generate a graph of blog posts in which two posts are connected if their temporal distance is less than our desired window size.

Now for each topic, we have content-based clusters of posts on one hand and temporally generated graph on the other hand. The goal is to see how likely it is for the posts in each content-based cluster to be connected to each other in the time-based graph. In other word, we investigate the tendency of posts in the graph to be connected to other posts with which they share a content-based cluster.

To evaluate this relation, we use the *assortativity coefficient* measure proposed by Newman [New03]. The measure is based on a matrix e in which e_{ij} shows fraction of the edges in the graph that connect a post from cluster i to a post from cluster j and satisfies following:

$$\sum_{ij} e_{ij} = 1, \quad \sum_{j} e_{ij} = a_i \tag{5.1}$$

where a_i is the fraction of edges in the graph that have a post from cluster i in one end. Since the temporal distance is a symmetric measure, in our graph e_{ij} and e_{ji} are equal. The *assortativity coefficient* is :

$$r = \frac{\sum_i e_{ii} - \sum_i a_i^2}{1 - \sum_i a_i^2} \tag{5.2}$$

r can have a value in [-1,1]. The higher the value of r is, the more likely it is that a post connects to other posts in its own cluster. The negative value of r shows that the posts are likely to connect to posts from other clusters.

We perform our experiments on the Blog08 collection and using 39 topics provided in TREC09 [MOS09]. For each topic, we select the top posts retrieved by a language modeling approach with Dirichlet smoothing. Then by generating the clusters and the graph, as described before, we calculate the assortativity coefficient for each topic separately.

There are three parameters effecting the assortativity coefficient: 1) number of the selected posts, 2) number of the clusters and 3) threshold for temporal

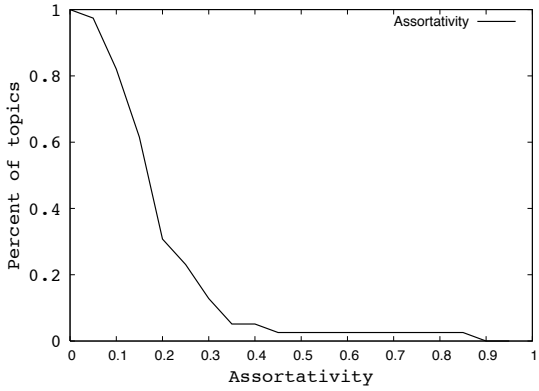

Figure 5.1. CCDF of percent of topics regarding to their assortativity

distance. We tune these parameters for each topic separately in order to get the maximum possible r.

Table 1 shows the three most and three least assortative topics. As we can see, in all the highly assortative topics there is a temporal requirement in the description. On the other hand, the three least assortative topics are fairly independent of time and can be discussed similarly in any point of time.

Figure 1 shows the Complementary Cumulative Distribution Function (CCDF) of the percent of topics with respect to their assortativity values. It can be seen that 61% of the posts have an assortativity value higher than 0.15 and the average value for all of the topics is 0.22. These values are higher than the values in the well-know assortative networks like co-authership where the coefficient is about 0.12 [New03]. These results show that on average blog queries have temporal dependency and it can be utilized in our retrieval models to improve the performance of retrieval system. Based on this observation, in the next section we develop our time-based query expansion method that employes the temporal information for estimating relevance model of query.

5.3 Query Expansion in Blog Retrieval

Query expansion is known to be effective in improving the performance of retrieval systems [CNGR08; LC01b; Sal71]. The idea is to add more terms to an initial query to disambiguate the query and solve the possible term mismatch problem between the query and the relevant documents. Automatic Query Expansion techniques usually assume that top retrieved documents are relevant to the topic and use their content to generate an expanded query. In some situa-

tions, it has been shown that it is better to have multiple expanded queries as apposed to the usual single query expansion, for example in server-based query expansion technique in distributed information retrieval [SAT09].

An expanded query, while being relevant to the original query, should have as much coverage as possible on all aspects of the query. If the expanded query is very specific to some aspects of the original query, we will miss part of the relevant documents in the re-ranking phase. In blog search context, where queries are more general than normal web search queries [EACC08], the coverage of the expanded query gets even more important.

Elsas *et al.* made the first investigation on the query expansion techniques for blog search [EACC08]. They show that normal feedback methods (selecting the new terms from top retrieved posts or top retrieved blogs) using the usual parameter settings is not effective in blog retrieval. However, they show that expanding query using an external resource like Wikipedia can improve the performance of the system. In a more recent work, Lee *et al.* [LNL09] propose new methods for selecting the appropriate posts as the source of expansion and show that these methods can be effective in retrieval. All these proposed methods can be summarized as follows:

- Top Feeds: Uses all the posts of the top retrieved feeds for the query expansion. This model has two parameters: the number of selected feeds and the number of terms in the expanded query [EACC08].

- Top Posts: Uses the top retrieved posts for the query expansion. The number of the selected posts and the number of terms to use for expansion are the two parameters of this model [EACC08].

- FFBS: Uses the top posts in the top retrieved feeds as the source for selecting the new terms. The number of selected posts from each feed is fixed among different feeds. This model has three parameters: the number of the selected feeds, the number of the selected posts in each feed and the number of the selected terms for the expansion [LNL09].

- WFBS: Works the same as FFBS. The only difference is that the number of the selected posts for each feed depends on the feed rank in the initial list, such that more relevant feeds contribute more in generating the new query. Like FFBS, WFBS has also three parameters that are: the number of the selected feeds, the total number of the posts to be used in the expansion, and the number of the selected terms [LNL09].

Among the mentioned methods, the Top Feeds method has the possibility to expand the query with non-relevant terms. The reason is that all the posts in a top retrieved feed are not necessarily relevant to the topic. On the other hand, the Top Posts method might not have enough coverage on all the sub-topics of the query, because the top retrieved posts might be mainly relevant to some dominant aspect of the query. FFBS and WFBS methods were originally proposed in order to have more coverage than the Top Posts method while selecting more relevant terms than the Top Feeds method [LNL09]. However, since it is difficult to summarize all the aspects of the topic in one single expanded query, these methods would not have the maximum possible coverage. It is worth mentioning that all the mentioned methods are based on the relevance model technique [LC01b] and their difference is in the selection of feedback documents.

In the following, we describe our proposed methods for query expansion in blog retrieval. In these methods, we mainly used temporal information for selecting appropriate terms in query expansion. In the first approach, we follow traditional settings in which one generates single expanded query using top retrieved posts. However we add time as a hidden variable that influence the term selection criteria. Finding that time is an important and effective factor in query expansion, in the second approach we use the temporal information for generating multiple expanded queries that can help us better represent queries over time.

5.4 Single Time-based Query Expansion

The goal of a relevance model is to select the most appropriate terms from initially retrieved documents to expand the original query [LC01a]. Terms are selected based on their relevance to the query with:

$$P(w|Q) \propto \sum_{\theta_D} P(w|\theta_D)P(\theta_D)\prod_{i}^{m} P(q_i|\theta_D) \tag{5.3}$$

where w is a candidate term and Q is the initial query. θ_D is the language model of an initially retrieved document D and $P(\theta_D)$ is the prior probability of the document that is usually set to the uniform probability distribution. $P(w|\theta_D)$ and $P(q_i|\theta_D)$ are the probabilities of a term w and a query term q_i in the document D respectively. At the end of the query expansion phase, top terms that have the highest probabilities will be added to the query and generate a new expanded query.

The final query relevance model is obtained as a mixture of the initial query and the expansion terms:

$$P(w|\theta'_Q) = (1-\alpha)P(w|\theta_Q) + \alpha P(w|Q) \tag{5.4}$$

where α is the parameter and in our experiments is set to be 0.5, which has been shown to be effective in different models [LZ09].

When we want to use relevance models in blog retrieval, we should decide about documents to be used in the expansion. Then we should define which terms from each document we want to select and add them to the original query. We propose a time-based relevance model where we assume that the date has effect on the terms written for a specific topic. In other word, we assume that the generative model of the query first selects a date for a given query and then select a term given the selected date and the query:

$$P(w|Q) = \sum_{day} P(w|day,Q) \cdot P(day|Q) \tag{5.5}$$

where $P(w|day,Q)$ shows the importance of a term in a given day for the query and $P(day|Q)$ shows the importance of a day for the query. These probabilities are estimated as follows:

$$P(w|day,Q) = \sum_{p \in day} P(w|p) \cdot P(p|day,Q) \tag{5.6}$$

$$P(day|Q) = \frac{\sum_{p \in day} score(p,Q)}{\sum_{day'} \sum_{p \in day'} score(p,Q)} \tag{5.7}$$

where p shows a post and $p \in day$ means the post is published in the specified day. $P(p|day,Q)$ is set to be uniform for all the posts in each day and $score(p,Q)$ defines the similarity score of the post to the given query which in our experiments is calculated using language modeling technique with Dirichlet smoothing.

The assumption is that selecting the expansion terms based on time would provide us with a better coverage over sub-topics than the Top Posts method. On the other hand, since we use the top retrieved posts in the expansion phase, we add less irrelevant terms than the Top Feeds method. This model has two parameters, including the number of posts considered in the expansion phase and the number of terms selected for expansion.

We employ the Blogger Model method for the initial ranking of the blogs [BdRW08]. Blogger Model is one of the most effective blog retrieval techniques with a solid probabilistic intuition based on the expert search methods

Model	MAP	P@10	Bpref
BloggerModel	0.2745	0.3974	0.2867
TopBlogs	0.2777	0.4128	0.2973
TopPosts	0.2848	0.4282	0.3064
FFBS	0.2857	0.4256	0.3031
WFBS	0.2882	0.4256	0.3072
TemporalRM	**0.2931** ⋆†	**0.4308**	**0.3116** ⋆†

Table 5.2. Evaluation results for single time-based query expansion over TREC09 data set.

[WBdR10b; BdRW08]. The detail of the model is described in section 2.3.3. All the parameters of the models are tuned by a 5-fold cross validation technique. These parameters, depending on the system, include the number of posts and blogs used in the expansion and the number of terms selected for expansion.

Table 1 summarizes the retrieval performance of Blogger Model and the baseline query expansion methods along with time-based relevance model on the TREC'09 data set. It shows that the temporal relevance model outperforms all the baseline techniques in all the evaluation measures. Statistical significant tests are performed using Paired T-test at 0.05 level of significance. It can be seen that time-based relevance model has statistically significant improvements over the Blogger model and the Top Blogs model which is shown by $\star$ and † respectively.

The experimental results confirm our intuition that using the temporal information as a source for selecting terms can lead us to a better expanded query. Based on these preliminary results, we extend the model by generating multiple expanded queries in which we model the evolution of the topics. In the next section, we will describe this model in detail.

5.5 The TEMPER Framework: Multiple Time-based Query Expansions

Distillation topics are often multifaceted and can be discussed from different perspectives [EACC08]. Vocabulary usage in the documents relevant to a topic can change over time in order to express different aspects (or sub-topics) of the query. These dynamics might create the term mismatch problem such that a

query term may not be a good indicator of the query topic in all different time intervals. In order to address this problem, we propose a time-based query expansion method which expands queries with different terms at different times. This contrasts other applied query expansion methods in blog search where they generate only one single query in the expansion phase [LNL09; EACC08]. Our experiments on different test collections and different baseline methods indicate that time-base query expansion is effective in improving the retrieval performance and can outperform existing techniques.

In this section we describe our novel framework for time-based relevance feedback in blog distillation called TEMPER. TEMPER assumes that posts at different times talk about different aspects (sub-topics) of a general topic. Therefore, vocabulary usage for the topic is time-dependant and this dependency can be considered in a pseudo relevance feedback method. Following this intuition, TEMPER selects time-dependent terms for query expansion and generates one query for each time point. The TEMPER framework can be summarized in the following steps:

1. Time-based representation of blogs and queries
2. Time-based similarity between a blog and a query
3. Ranking blogs according to the their overall similarity to the query.

In the remainder of this section, we describe our approach in fulfilling each of these steps.

5.5.1 Time-Based Representation of Blogs and Queries

In order to consider time in the TEMPER framework, we first need to represent blogs and queries in the time space. For a blog representation, we distribute its posts based on their publish date. In order to have a daily representation of the blog, we concatenate all the posts that have the same date.

For a query representation, we take advantage of the top retrieved posts for the query. We select the top K most relevant posts for the query and divide them based on their publish date while concatenating posts with the same date. In order to have a more informative representation of the query, we select the top N terms for each day using the KL-divergence between the term distribution of the day and the whole collection [ZL01a].

Note that in the initial representation there can be days that do not have any term distribution associated with them. However, in order to calculate the

relevance of a blog to a query, TEMPER needs to have the representation of the blog and query in all days. We employ the available information in the initial representation to estimate the term distributions for the rest of the days.

TEMPER generates a representation for each topic or blog for each day based on the idea that a term at each time position propagates its count to the other time positions through a proximity-based density function. By doing so, we can have a virtual document for a blog/topic at each specific time position. The term frequencies of such a document are calculated as follows:

$$tf'(t,d,i) = \sum_{j=1}^{T} tf(t,d,j) \cdot K(i,j) \tag{5.8}$$

where i and j indicate time position (day) in the time space; T denotes the time span of the collection; tf' shows the term frequency of term t in blog/topic d at day i and it is calculated based on the frequency of t in all days; $K(i,j)$ decreases as the distance between i and j increases and can be calculated using kernel functions that we describe later.

The proposed representation of a document in the time space is similar to the proximity-based method where they generate a virtual document at each position of the document in order to capture the proximity of the words [LZ09; GCC10]. However, here we aim to capture the temporal proximity of terms. As the kernel function, we employ the Laplace kernel function which has been shown to be effective in the previous work [GCC10] together with the Rectangular (square) kernel function. In the following formulas, we present normalized kernel functions with their corresponding variance formula.

1. Laplace Kernel

$$k(i,j) = \frac{1}{2b} \exp\left[\frac{-\left|i-j\right|}{b}\right] \qquad \text{where } \sigma^2 = 2b^2 \tag{5.9}$$

2. Rectangular Kernel

$$k(i,j) = \begin{cases} \frac{1}{2a} & \text{if } \left|i-j\right| \leq a \\ 0 & \text{otherwise} \end{cases} \qquad \text{where } \sigma^2 = \frac{a^2}{3} \tag{5.10}$$

5.5.2 Time-Based Similarity Measure

By having the daily representation of queries and blogs, we can calculate the daily similarity between these two representations and create a *daily similarity*

vector for the blog and the query. The final similarity between the blog and the query is then calculated by summing over the daily similarities:

$$sim_{temporal}(B,Q) = \sum_{i=1}^{T} sim(B,Q,i) \tag{5.11}$$

where $sim(B_i,Q_i)$ shows the similarity between a blog and a query representation at day i and T shows the time span of the collection in days.

Another popular method in time series similarity calculation is to see each time point as one dimension in the time space and use the Euclidian length of the *daily similarity vector* as the final similarity between the two representations [KP99]:

$$sim_{temporal}(B,Q) = \sqrt{\sum_{i=1}^{T} sim(B,Q,i)^2} \tag{5.12}$$

We use the cosine similarity as a simple and effective similarity measure for calculating similarity between the blog and the topic representations at the specific day i:

$$sim(B,Q,i) = \frac{\sum_w tf(w,B,i) \cdot tf(w,Q,i)}{\sqrt{\sum_w tf(w,B,i)^2 \cdot \sum_w tf(w,Q,i)^2}} \tag{5.13}$$

The normalized value of the temporal similarity over all blogs is then used as $P_{temporal}$.

$$P_{temporal}(B|Q) = \frac{sim_{temporal}(B,Q)}{\sum_{B'} sim_{temporal}(B',Q)} \tag{5.14}$$

Finally in order to take advantage of all the available evidence regarding the blog relevance, we interpolate the temporal score of the blog with its initial relevance score.

$$P(B|Q) = \alpha P_{\texttt{initial}}(B|Q) + (1-\alpha) P_{\texttt{temporal}}(B|Q) \tag{5.15}$$

where α is a parameter that controls the amount of temporal relevance that is considered in the model. We use the Blogger Model method for the initial ranking of the blogs [BdRW08]. The only difference with the original Blogger Model is that we set the prior of a blog to be proportional to the log of the number of its posts, as opposed to the uniform prior that was used in the original Blogger Model. This log-based prior has been used and shown to be effective by Elsas *et al.* [EACC08].

Model	MAP	P@10	Bpref
BloggerModel	0.2774	0.4154	0.2906
TopFeeds	0.2735	0.3897	0.2848
TopPosts	0.2892	0.4230	0.3057
FFBS	0.2848	0.4128	0.3009
WFBS	0.2895	0.4077	0.3032
TEMPER-Rectangular-Sum	0.2967 †	0.4128	0.3116 †
TEMPER-Rectangular-Euclidian	0.3014 † ‡ ∗	**0.4435** ∗	0.3203 † ‡ ∗
TEMPER-Laplace-Sum	0.3086 †	0.4256	**0.3295** †
TEMPER-Laplace-Euclidian	**0.3122** † ‡ ∗	0.4307	0.3281 † ∗

Table 5.3. Evaluation results of TEMPER over TREC'09 data set.

5.5.3 Experimental Results

In this section we explain the experiments that we conducted in order to evaluate the usefulness of the proposed method.

We compare the results of our method with different baseline methods. We perform our feedback methods on top of the Blogger Model method [BdRW08]. Therefore, Blogger Model is the first baseline against which we will compare the performance of our proposed methods. We also compare the results with the existing query expansion methods for blog feed retrieval [EACC08; LNL09]. In order to have a fair comparison, we implemented the mentioned query expansion methods on top of the Blogger Model. We tuned the parameters of these models using a 10-fold cross validation in order to maximize MAP. The last set of baselines is provided by TREC organizers as part of the blog facet distillation task in 2009. We use these baselines to see the effect of TEMPER in re-ranking the results of other retrieval systems.

We mainly focus on the results of TREC'09 data set, as it is one the most recent data set and has enough temporal information which is an important feature for our analysis. However, in order to see the effect of the method on the smaller collections, we briefly report the final results on the TREC'07 and TREC'08 data sets as well.

Table 5.3 summarizes the retrieval performance of the Blogger Model and the baseline query expansion methods along with different settings of TEMPER on the TREC 2009 data set. The best value in each column is in bold face. A dag(†), a ddag(‡) and a star(∗) indicate statistically significant improvement over Blogger Model, TopPosts and WFBS respectively. As can be seen from the table, none of the query expansion baselines improves the underlying Blogger

Model	MAP	P@10	Bpref
BloggerModel	0.2453	0.4040	0.2979
TopPosts	0.2567	0.4080	0.3090
WFBS	0.2546	0.3860	0.3087
TEMPER-Laplace-Euclidian	**0.2727** † ‡ ∗	**0.4380** † ‡ ∗	**0.3302** † ∗

Table 5.4. Evaluation results of TEMPER over TREC'08 data set.

Model	MAP	P@10	Bpref
BloggerModel	0.3354	0.4956	0.3818
TopPosts	0.3524 †	0.5044	0.3910
WFBS	0.3542 †	**0.5356** † ‡	0.3980
TEMPER-Laplace-Euclidian	**0.3562** †	0.5111	**0.4011**

Table 5.5. Evaluation results of TEMPER over TREC'07 data set.

Model significantly.

From table 5.3 we can see that TEMPER with different settings (using rectangular/Laplace kernel, sum/Euclidean similarity method) improves Blogger Model and the query expansion methods significantly. These results show the effectiveness of time-based representation of blogs and query and highlights the importance of time-based similarity calculation of blogs and topics.

In tables 5.4 and 5.5 we present similar results over TREC'08 and TREC'07 data sets. Over the TREC'08 dataset, it can be seen that TEMPER improves Blogger Model and different query expansion methods significantly. Over the TREC'07 dataset, TEMPER improves Blogger Model significantly. However, the performance of TEMPER is comparable with the other query expansion methods and the difference is not statistically significant.

Model	MAP	P@10	Bpref
stdBaseline1	0.4066	0.5436	0.4150
TEMPER-stdBaseline1	0.4114	0.5359	0.4182
stdBaseline2	0.2739	0.4103	0.2845
TEMPER-stdBaseline2	0.3009†	0.4308 †	0.3158†
stdBaseline3	0.2057	0.3308	0.2259
TEMPER-stdBaseline3	0.2493†	0.4026†	0.2821†

Table 5.6. Evaluation results for the standard baselines on TREC'09 data set.

Model	MAP	P@10	Bpref
TEMPER-Laplace-Euclidian	**0.3122**	**0.4307**	**0.3281**
TREC09-rank1 (buptpris 2009)	0.2756	0.3206	0.2767
TREC09-rank2 (ICTNET)	0.2399	0.3513	0.2384
TREC09-rank3 (USI)	0.2326	0.3308	0.2409

Table 5.7. Comparison with the best TREC'09 title-only submissions.

Model	MAP	P@10	Bpref
TEMPER-Laplace-Euclidian	0.2727	0.4380	0.3302
TREC08-rank2 (CMU-LTI-DIR)	**0.3056**	0.4340	0.3535
TREC08-rank1 (KLE)	0.3015	**0.4480**	**0.3580**
TREC08-rank3 (UAms)	0.2638	0.4200	0.3024

Table 5.8. Comparison with the best TREC'08 title-only submissions.

As mentioned before, we also consider the three standard baselines provided by TREC'10 organizers in order to see the effect of our proposed feedback method on retrieval baselines other than Blogger Model. Table 5.6 shows the results of TEMPER over the TREC baselines. It can be seen that TEMPER improves the baselines in most of the cases.

Tables 5.7, 5.8 and 5.9 show the performance of TEMPER compared to the best title-only TREC runs in 2009, 2008 and 2007 respectively. It can be seen from the tables that TEMPER is performing better than the best TREC runs over the TREC'09 dataset. The results over the TREC'08 and TREC'07 are comparable to the best TREC runs and can be considered as the third and second best reported results over TREC'08 and TREC'07 datasets respectively.

TEMPER has four parameters including: the number of the posts selected for expansion, the number of the terms that are selected for each day, the standard deviation (σ) of the kernel functions, and α as the weight of the initial ranking

Model	MAP	P@10	Bpref
TEMPER-Laplace-Euclidian	0.3562 †	0.5111	**0.4011**
TREC07-rank1 (CMU)	**0.3695**	**0.5356**	0.3861
TREC07-rank2 (UGlasgow)	0.2923	0.5311	0.3210
TREC07-rank3 (UMass)	0.2529	0.5111	0.2902

Table 5.9. Comparison with the best TREC'07 title-only submissions.

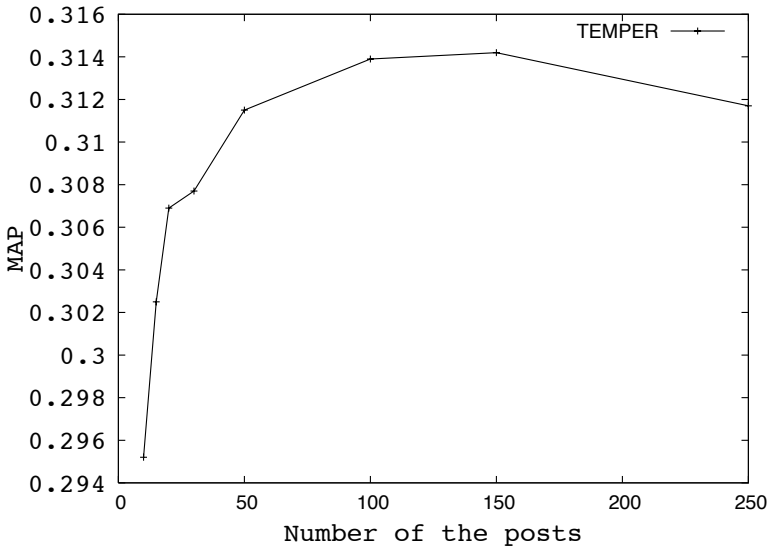

Figure 5.2. Effect of number of the posts used for expansion on the performance of TEMPER.

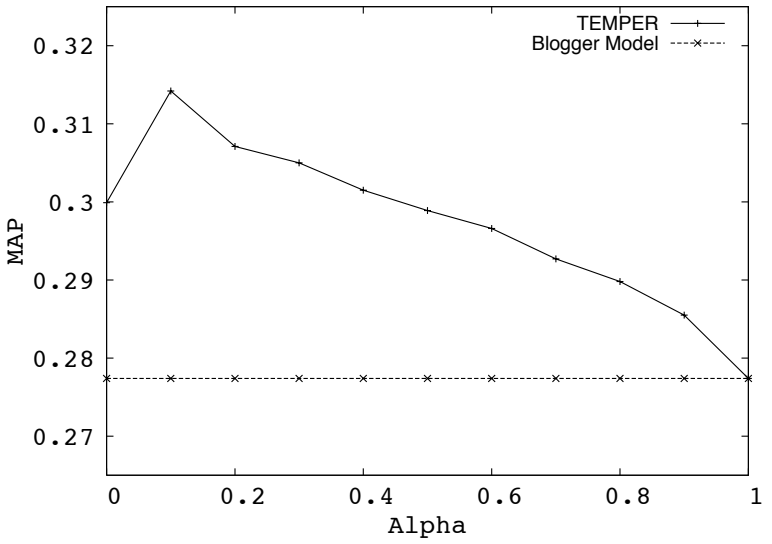

Figure 5.3. Effect of alpha on the performance of TEMPER.

score.

Among these parameters, we the fix number of the terms for each day to be 50, as used in a previous work [EACC08]. The standard deviation of the kernel function is estimated using top retrieved posts for each query. Since the goal of the kernel function is to model the distribution of distance between two consequent relevant posts, we assume the distances between selected posts (top retrieved posts) to be the samples of this distribution. We then use the standard deviation of the sample as an estimation for σ.

The other two parameters are tuned using a 10-fold cross validation method. Figure 5.2 and 5.3 show the sensitivity of the system to these parameters. It can be seen that the best performance is gained by selecting about 150 posts for expansion with any number more than 50 gives a reasonable result. The value of α depends on the underneath retrieval model. We can see that TEMPER outperforms the Blogger Model for all values of α and the best value of α is about 0.1.

The most time consuming part of the algorithm is extracting important expansion terms for a query and calculating time-based similarity between a query and a blog. The runtime for extracting expansion terms is correlated with the number of documents that we use in the expansion phase. We saw that number of documents does not need to be high and with less than 100 documents we can achieve reasonable results. By limited number of documents, our query expansion phase would not be much longer than the expansion phase in the traditional relevance model.

After expanding the query, we need to calculate the similarity between the time-based representation of query and the one from blogs. Each blog can have many posts that would make this phase time consuming. However, by using only the top-retrieved posts from each blog in its time-based representation, number of posts for each blog would be limited. And also, one can make changes into method to further optimize runtime of the algorithm. One of the possible optimizations would be to not generate fully time-based representation of blogs and queries. Instead we can have representation for each blog or query only in those dates that there is data. Then we can change the similarity measure that calculates similarity only between those days that we have representation for and decrease the resulting similarity promotional to the distance between dates. With such changes we can make the runtime of the algorithm significantly shorter.

5.6 Conclusion

In this section we focused on the temporal properties of blogs and its application in query expansion for blog retrieval. Following the intuition that the term distribution for a topic might change over time, we propose a time-based query expansion technique. We showed that it is effective to have multiple expanded queries for different time points and score the posts of each time using the corresponding expanded query. Our experiments on different blog collections and different baseline methods showed that this method can improve on state of the art query expansion techniques.

Chapter 6

Diversity

6.1 Introduction

Previous work have not investigated enough the dependency between posts of a blog in estimating the relevance. When it comes to blog relevance estimation, the usual emphasis is on the number of topic-related posts that a blog publishes. Nevertheless, the amount of new information that each of these posts adds to the blog is another influential factor that can affect the blog relevance. If a blog publishes repetitive information with low novelty, it would be less interesting for the user to follow.

In this section, we study the effect of on-topic diversity of blog posts in the blog relevance. By on-topic diversity of blog posts we mean that those posts that are about the query topic need to have high diversity and cover different sub-topics of the query. We investigate three types of on-topic diversity and their effect on retrieval performance: topical diversity, temporal diversity and hybrid diversity. Our experiments over different blog collections and different baseline methods show that on-topic diversity can improve the performance of the retrieval system.

In our diversity-based methods we utilize the novelty of each blog post and penalize those blogs that have low diversity in their posts related to the topic. There are different scenarios where considering the diversity might help the retrieval performance. Table 6.1 shows some real examples of blogs that were affected by one of our diversity-based methods. The examples are part of our experiments on two selected topics of TREC'09 and are compared to the best performing baseline method. We can see that by penalizing those blogs that have low diversity, we can filter out non-relevant blogs like spam blogs or blogs that publish similar posts multiple times. On the other hand, it can help us to

Topic ID: 1107; Topic title: Mountain Climbing Topic description: Looking for blogs about equipment, clubs and paths for climbing
Example 1 Feed no: BLOG08-feed-1218342 Blog URL: http://playwintersports.blogspot.com Blog title: "INFORMATION ABOUT DIFFERENT TYPES OF WINTER SPORTS AND WHERE TO FIND EQUIPMENT" Explanation: This is another retrieved non-relevant blog that has four posts about "*Mount McKinley*" with identical content. Those posts are probably published mistakenly and considering all of them for scoring the blog gives a high score to the blog. Its rank in the baseline method: 61 Its rank after applying diversity: above 100 (not retrieved)
Example 2 Feed no: BLOG08-feed-070681 Blog URL: http://himadventures.blogspot.com Blog title: "Trekking, Camping, Climbing And Traveling In Himalayas" Explanation: This is a relevant blog that belongs to a professional climber who describes their trips to Himalayas. He describes the climbs and conditions of the mountains in very detail. The detailed description and in-depth vocabulary usage makes its posts very long with low scores in response to the query. Beside, these properties make the posts less similar to each other while each one adds new information to the blog with respect to the query. Its rank in the baseline method: above 100 (not retrieved) Its rank after applying diversity: 81
Topic ID: 1122; Topic title: Skiing Topic description: looking for blogs with information and advice on skiing, ski resorts and ski organizations.
Example 3 Feed no: BLOG08-feed-616929 Blog URL: http://justjetskis.blogspot.com Blog title: "JUST SKI RESOURCES" Explanation: This is a spam blog advertising "jet ski". Most of the posts in the blog are advertisement for jet ski equipment with almost identical content. Its rank in the baseline method: 1 Its rank after applying diversity: above 100 (not retrieved)

Table 6.1. Examples where considering diversity improves blog retrieval

retrieve those relevant blogs that have high diversity among their posts while they might have low initial score for the query.

In this chapter we discuss how to add a measure of diversity to the existing blog retrieval methods and study its effect on retrieval performance. Our main aim is to answer the following questions::

- How important is the diversity of blog posts to the blog relevance?
- What types of diversity can we define over blog posts?
- How can we capture the diversity of blog posts and integrate it into a blog feed retrieval method?
- For what type of queries can we expect diversity-based methods to be more effective?

6.2 Diversity in Blog Retrieval

The relation between posts in a blog can give us useful information about the blog. Previous studies considered cohesiveness as a way to capture how posts in the blog are similar to the blog in general [EACC08; SC08; MO08]. In these studies, a blog with higher cohesiveness and thus less diversity is considered a better blog to retrieve. Elsas *et al.* define the centrality of a post as its similarity to the blog as a whole. They assume that if the retrieved posts from a blog have high centrality that blog is a better candidate for retrieval [EACC08]. Similarly, Seo and Croft penalize those blogs that have high diversity among their posts [SC08]. Defining the goal of a blog retrieval system to retrieve topic-centric blogs, they assume that blogs with high diversity are not topic-centric and thus should be penalized. MacDonald and Ounis define similar measures to retrieve blogs with focused interests [MO08]. Beside the content of the posts, they also consider temporal distribution of posts to retrieve blogs with recurring interests.

In contrast to previous work where all the posts of the blog are used for estimations, we focus only on those posts that are about the query topic. While diversity of a blog as a whole might be negative evidence for blog relevance, we assume that on-topic diversity is an asset. In other words, we assume that a relevant blog should have a high coverage over sub-topics and thus should have a high diversity among posts that it publishes about the topic.

We define three types of diversity over the posts and investigate their effectiveness on the performance of a blog retrieval system:

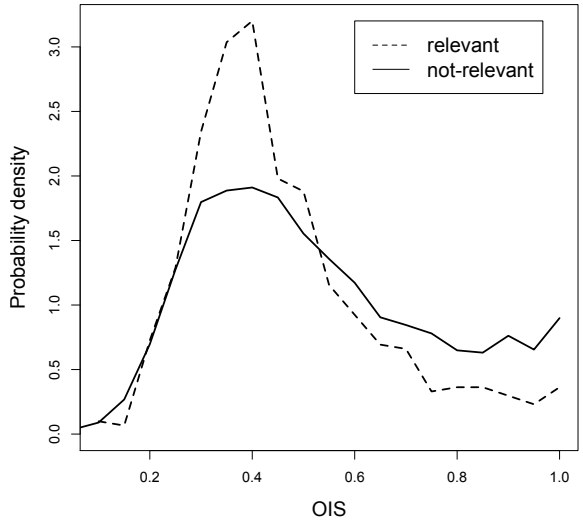

Figure 6.1. Distribution of On-topic Intra-feed Similarity using top 2000 retrieved posts.

- Topical Diversity
- Temporal Diversity
- Hybrid Diversity

In the following we discuss each of the methods in more detail.

6.2.1 Topical Diversity of Blog Posts

First we investigate topical diversity among the posts of each blog. Blog distillation queries tend to be more general than normal web queries and thus have a wider range of sub-topics [EACC08]. We assume that posts that are retrieved from each blog for the query should have high diversity in order to cover different sub-topics. We are interested to determine how the on-topic diversity of posts can affect the relevance of a blog to the query.

In order to test if the diversity of blog posts is an important factor in the relevance of blogs, we carry out preliminary experiments on the Blog08 collection. In these experiments, we assume that higher similarity of blog posts indicates less topical diversity among them. This assumption is consistent with previous work on diversification [CG98].

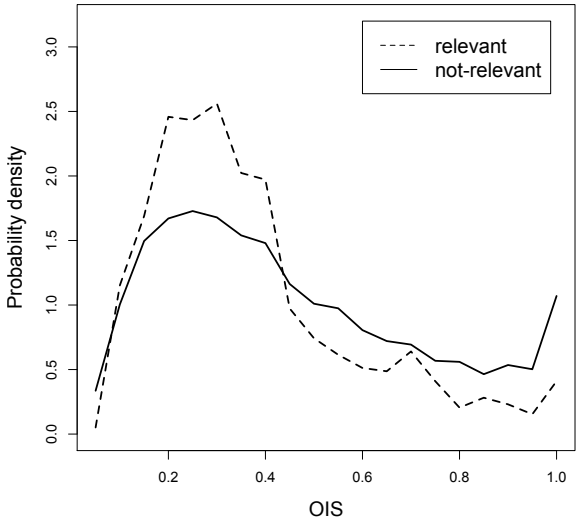

Figure 6.2. Distribution of On-topic Intra-feed Similarity using top 15000 retrieved posts.

For each query, we first retrieve the top n posts using a traditional retrieval method. Then for each blog that has more than one post in the retrieved set, we calculate the average similarity between its retrieved posts, which we call *On-topic Intra-feed Similarity* (OIS):

$$OIS(b_i, q) = \frac{\sum_{j=1}^{|b_i \cap R(q)|} \sum_{l=j+1}^{|b_i \cap R(q)|} sim(p_{ji}, p_{li})}{\binom{|b_i \cap R(q)|}{2}} \tag{6.1}$$

We use cosine similarity as the similarity measure between two posts. The higher the *OIS* value, the less diverse is the feed with respect to the query.

Figure 6.1 shows the distribution of *OIS* for relevant and non-relevant blogs. It can be seen that non-relevant blogs are more likely to have high *OIS* values. The mean of *OIS* for relevant blogs is 0.43 compared to 0.50 for non-relevant blogs. Based on the Student's t-test, the difference between the mean values is statistically significant with a p-values equal to 2.6e-16. This shows the possibility of using the diversity of posts for discriminating between relevant and non-relevant blogs and consequently having a better retrieval system.

In figure 6.1, we use the top 2000 posts for each topic. Similar behavior has

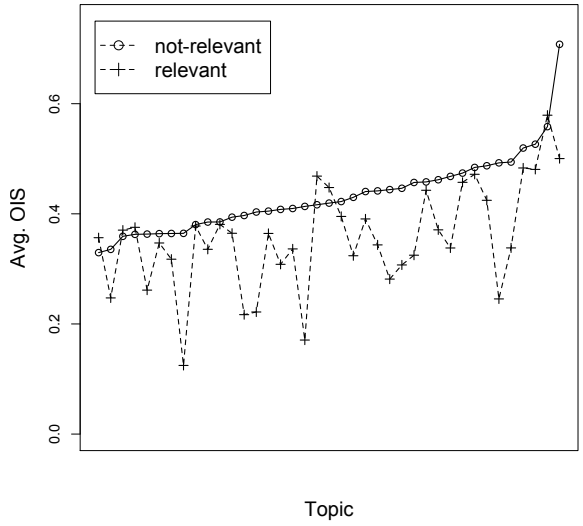

Figure 6.3. Average *OIS* values for each topic using top 15000 retrieved posts.

been observed for a smaller number of posts. However, it is possible that with an increasing number of retrieved posts, we retrieve more posts for each blog and we end up with less diverse relevant blogs or more diverse non-relevant blogs. In order to test this possibility, we run the same analysis with the top 15000 posts for each query. Figure 6.2 shows the outcome of this experiment. The mean of *OIS* values for relevant blogs is 0.34 compared to 0.42 for non-relevant blogs. The difference between the mean values is statistically significant with the p-value less than 2.2e-16. As we can see, the *OIS* values decrease with increasing numbers of posts. However, the difference between relevant and non-relevant blogs still exists. This is quite surprising, since we expect that by retrieving more posts we increase the chance that non-relevant blogs will have a more diverse set of retrieved posts.

The per-topic analysis in figure 6.3 shows the difference more clearly. This figure shows the average of *OIS* values for relevant and non-relevant blogs for each query topic. For clarity, topics are sorted by the average *OIS* of their non-relevant feeds. As it can be seen for most topics, non-relevant feeds have higher similarity among their posts compared to the relevant ones. Based on these observations, we propose methods to take into account the diversity of blogs in the ranking.

To capture the diversity of blog posts, we adapt a variation of the Maximal

Marginal Relevance (MMR) [CG98]. The goal of MMR is to maximize diversity of a retrieved set of documents. It selects documents that are more similar to the query and less similar to already retrieved documents:

$$MMR \stackrel{def}{=}$$
$$Arg\ max_{d_i \in R \setminus S} \left[\lambda sim(d_i, q) - (1-\lambda) \max_{d_j \in S} sim(d_i, d_j) \right]$$

where R is an initial set of documents and S is the set of already retrieved documents.

Similar to the MMR method, our diversity detection method exploits the fact that similar documents are less diverse, thus it penalizes posts that are similar to other posts in the blog. In other words, only information of a post that does not appear in other blog posts can contribute to blog relevance:

$$\begin{aligned} &score_{div-topical}(p_i, q) = \\ &score_{ini}(p_i, q)(1 - \lambda \max_{p_j \in S} sim_{topical}(p_i, p_j)) \end{aligned} \tag{6.2}$$

where $score_{ini}(p_i, q)$ is the initial score of post p_i for the query q. S is the set of posts that belong to the same blog and have higher scores than p_i. This method assumes that $sim(p_i, p_j)$ is in $[0, 1]$ and thus it decreases the post score based on its similarity to other posts. The parameter λ controls the importance of the post novelty in its score, which can vary for different similarity methods or different queries. $sim_{topical}$ captures the content similarity between the two posts and can be replaced by any similarity measure. We use $cosine$ similarity, since it always has a value in $[0, 1]$ and does not need an extra normalization step:

$$sim_{topical}(p_i, p_j) = \frac{\sum_w tf(w, p_i) \cdot tf(w, p_j)}{\sqrt{\sum_w tf(w, p_i)^2 \cdot \sum_w tf(w, p_j)^2}} \tag{6.3}$$

where $tf(w, p)$ is the term frequency of the term w in the post p. When there is no similarity between a post and other blog posts that have higher score, the maximum of similarities will be zero. In this case, the diversity score of the post will be the same as its initial score and therefore all the relevance evidence of the post can contribute to the blog relevance score. In contrast, if there is another post very similar to the current post, then the cosine similarity will be close to one. As a result, depending on the value of λ, the diversity score of the post can be close to zero. Therefore, the blog will not gain any relevance from that post even if the post has high query likelihood.

After obtaining new scores for posts, any existing aggregation method can be used to aggregate the new scores and calculate the score of blogs. Applying the diversity-based scores introduces a new parameter λ that needs to be estimated.

6.2.2 Temporal Diversity of Blog Posts

In addition to the content of blog posts, temporal information is another important source of information that can be used by the retrieval system [KGC11; MO08]. Analogous to topical diversity, we can assume that a relevant blog should have a high temporal diversity among its published posts for the query. In other words, we expect that blog posts have high coverage over the temporal space and not be concentrated on specific time windows.

We employ a similar approach to the topical diversity where we penalize the posts that have high temporal similarity to other posts. To this end, we define a temporal similarity function between two posts as an un-normalized Gaussian function:

$$sim_{temporal}(p_i, p_j) = e^{\frac{(t_i - t_j)^2}{2\sigma^2}} \tag{6.4}$$

where t_i is the time-stamp of p_i given in days. We can see that the temporal similarity has a value between zero and one. If the two posts are published in the same day then their similarity will be one. The similarity value decreases with increasing the temporal distance between the two posts. Finally, we penalize post scores based on their temporal similarity:

$$\begin{aligned} &score_{div-temporal}(p_i, q) = \\ &score_{ini}(p_i, q)(1 - \lambda \; max_{p_j \in S} \; sim_{temporal}(p_i, p_j)) \end{aligned} \tag{6.5}$$

Using this method, if the post is published around the same date as some other posts, it will contribute less to the blog relevance than a post which is published at a distant time. In other words, this method captures the property of whether the blog posts are published all over the temporal space or if they are mostly published in some specific small time windows.

The temporal similarity function adds a new parameter σ to the model which is the standard deviation of the Gaussian function.

6.2.3 Hybrid Diversity

Finally, we define hybrid measure of diversity that takes both topical and temporal diversities into account. Two possible cases in which temporal and topical diversities can complement each other can be described as follows:

- Retrieve a relevant blog that writes about similar sub-topics but in different time windows. This might be a subtopic that is discussed periodically over time. For example any seasonal query that repeats over time can have such a property. In this case the topical diversity would be low and the temporal diversity would be high. Thus a combination of the two diversities can better handle the situation than considering only topical diversity which would over-penalize the blog.

- Retrieve a relevant blog that writes about different subtopics in the same time window. A blog can heavily discuss the topic in some time periods and if the published posts are about different sub topics, they might be valuable for the blog relevance. In this case, we would have low temporal diversity and high topical diversity and may need a combination of the two measures for better representation.

To consider these cases in our method, we define an hybrid similarity function as follows:

$$sim_{hybrid}(p_i, p_j) = sim_{topical}(p_i, p_j) \,.\, sim_{temporal}(p_i, p_j)$$

where $sim_{topical}$ and $sim_{temporal}$ are calculated using formula 6.3 and formula 6.4 respectively. Similar to topical and temporal diversity scores in formulas 6.2 and 6.5, we penalize those posts that have high hybrid similarity with other posts.

We can see that in this method, we mainly penalize those posts that are published around the same date as other posts and also have very similar content to them. Thus, in any of the two mentioned scenarios where one of the similarities is high and the other one is low, the post will not be highly penalized and can still contribute to the blog relevance.

6.3 Experimental Results

We conduct our experiments over four years worth of TREC blog track data from the blog distillation task, including TREC'07, TREC'08, TREC'09 and TREC'10 data sets.

We use the techniques discussed in section 2.3 as our baseline methods including $CombSum$, SDM and PCS methods. We apply each of these methods on the language model scores and also on the three diversified scores calculated by our proposed methods.

Since TREC'07 and TREC'08 query sets share the same collection, we use one to tune the parameters for the other. We do the same for the TREC'09 and

Model	MAP	P@10	Bpref
CombSum	0.2259	0.3844	0.2721
PCS	0.2695	0.4378	0.3115
SDM	0.2867	0.4444	0.3439
CombSum-topical	0.2506 ↑	0.4244 ↑	0.2859 ↑
PCS-topical	0.2901 ↑	0.4422	0.3184 ↑
SDM-topical	0.3168 ↑	0.4756 ↑	**0.3667** ↑
CombSum-temporal	0.2612 ↑	0.4267	0.2840 ↑
PCS-temporal	0.2844 ↑	0.4911 ↑	0.3134
SDM-temporal	0.3023 ↑	0.4756 ↑	0.3539
CombSum-hybrid	0.2466 ↑	0.4333 ↑	0.2866 ↑△
PCS-hybrid	0.2961 ↑△	0.4622	0.3197 ↑△
SDM-hybrid	**0.3191** ↑△	**0.5156** ↑△ ▲	0.3622 ↑△

Table 6.2. Evaluation results for the implemented models over TREC'07 data set.

TREC'10 query sets that share the Blog08 collection. By fixing the parameters k and σ in the PCS method and the Dirichlet smoothing parameter across all the methods, the remaining parameters to be tuned are the following:

- The number of top retrieved posts n, that are initially retrieved for each query. The examined values for this parameter are 500, 1000, 2000, 4000, 8000 and 15000.

- The diversity parameter λ, in each of the diversity methods. We tested different values for this parameter as follows: 0.01, 0.05, 0.1, 0.2, ..., 0.9, 0.95 and 0.99.

- The standard deviation of the Gaussian function σ, in the temporal and hybrid diversity methods. For the Blog06 collection we tried the values 3, 5, 7, 15, 30, 45, 60, 90. Since the Blog08 collection has a wider time span, we also tried the values 120, 250 and 370 for the TREC'09 and TREC'10 experiments.

Tables 6.2, 6.3, 6.4 and 6.5 show the performance evaluation of the methods over TREC'07, TREC'08, TREC'09 and TREC'10 respectively. The first three rows in each table represent the performance of baseline methods. The second three rows show the performance of the corresponding methods based on the

Model	MAP	P@10	Bpref
CombSum	0.1719	0.3000	0.2380
PCS	0.2016	0.3400	0.2709
SDM	0.2096	0.3740	0.2699
CombSum-topical	0.1847	0.3340	0.2446
PCS-topical	0.2086	0.3460	0.2640
SDM-topical	0.2284 ↑	0.3820	0.2729
CombSum-temporal	0.1852 ↑	0.3380 ↑	0.2430
PCS-temporal	0.2073	0.3660 ↑	0.2540
SDM-temporal	0.2289 ↑	0.3800	0.2769
CombSum-hybrid	0.1918 ↑▲	0.3460 ↑	0.2422
PCS-hybrid	0.2179 ↑△	0.3500	0.2661
SDM-hybrid	**0.2329** ↑	**0.3840**	**0.2777**

Table 6.3. Evaluation results for the implemented models over TREC'08 data set.

Model	MAP	P@10	Bpref
CombSum	0.1889	0.3154	0.2148
PCS	0.2093	0.3103	0.2279
SDM	0.2636	0.3821	0.2858
CombSum-topical	0.2180 ↑	0.3282	0.2297 ↑
PCS-topical	0.2196 ↑	0.3282	0.2374 ↑
SDM-topical	0.2888 ↑	**0.4333** ↑	0.3091 ↑
CombSum-temporal	0.2087 ↑	0.3051	0.2187
PCS-temporal	0.2082	0.3103	0.2250
SDM-temporal	0.2759	0.3846	0.2909
CombSum-hybrid	0.2176 ↑△	0.3282 △	0.2334 ↑△
PCS-hybrid	0.2224 ↑△	0.3462 ↑△	0.2387 ↑△
SDM-hybrid	**0.2961** ↑△▲	0.4308 ↑△	**0.3125** ↑△

Table 6.4. Evaluation results for the implemented models over TREC'09 data set.

Model	MAP	P@10	Bpref
CombSum	0.1631	0.2696	0.1631
PCS	0.1758	0.2870	0.1771
SDM	0.2273	0.3022	0.2273
CombSum-topical	0.1856 ↑	0.2435	0.1856 ↑
PCS-topical	0.1843	0.2913	0.1843
SDM-topical	0.2420 ↑	0.3435 ↑	0.2559 ↑
CombSum-temporal	0.1623	0.2196	0.1623
PCS-temporal	0.1759	0.2500	0.1759
SDM-temporal	0.2151	0.2891	0.2231
CombSum-hybrid	0.1738 △	0.2239	0.1831 ↑△
PCS-hybrid	0.1842	0.3000 ↑	0.1842
SDM-hybrid	**0.2571** ↑△ ▲	**0.3500** ↑△	**0.2649** ↑△

Table 6.5. Evaluation results for the implemented models over TREC'10 data set.

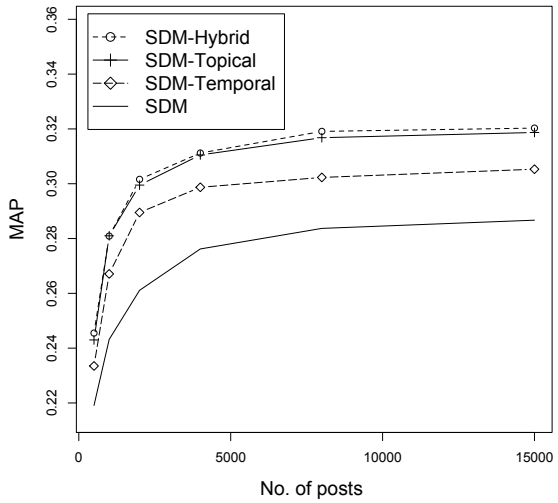

Figure 6.4. Effect of the number of the top retrieved posts in the performance over TREC'07 data set.

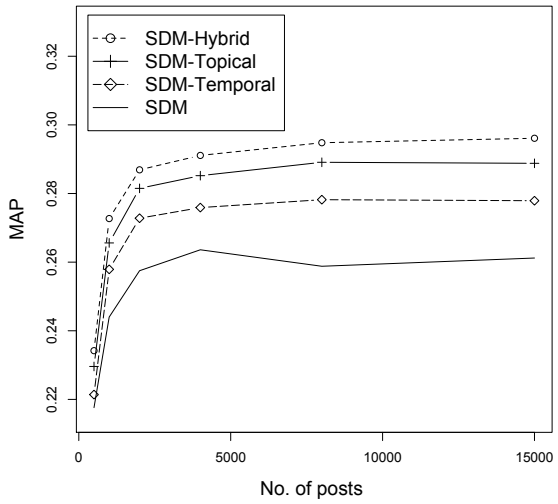

Figure 6.5. Effect of the number of the top retrieved posts in the performance over TREC'09 data set.

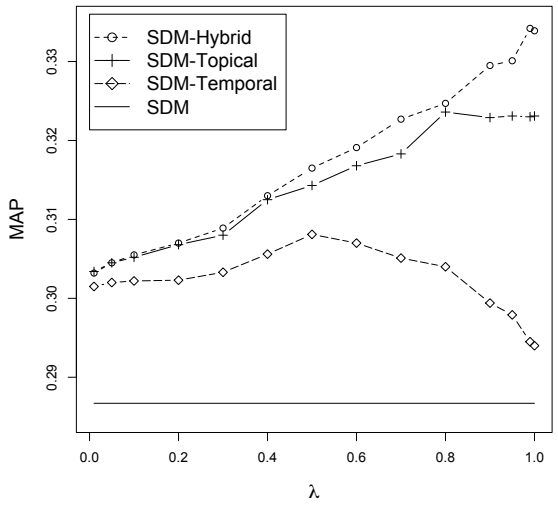

Figure 6.6. Effect of the diversity parameter λ in the performance over TREC'07 data set.

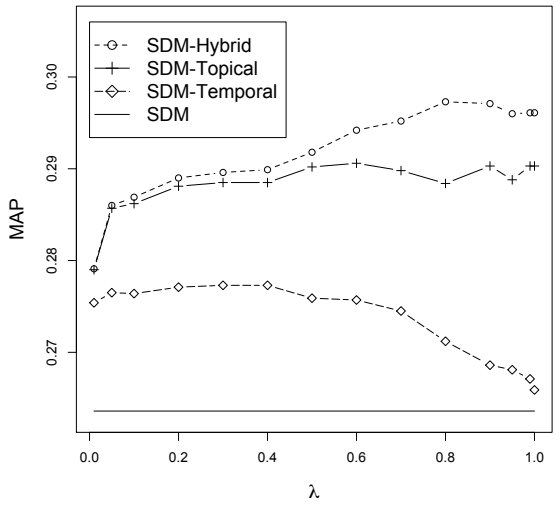

Figure 6.7. Effect of the diversity parameter λ in the performance over TREC'09 data set.

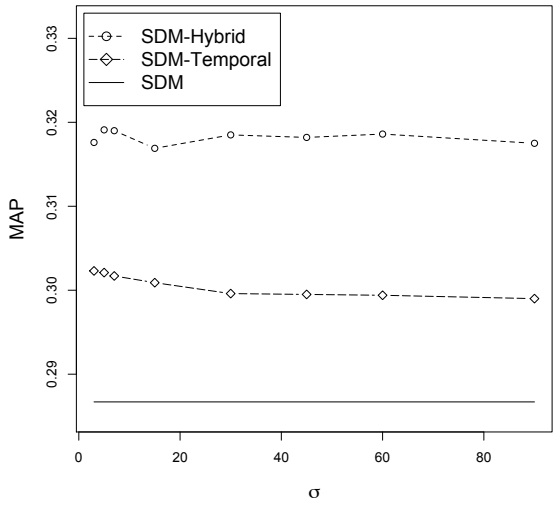

Figure 6.8. Effect of the parameter σ in the performance over TREC'07 data set.

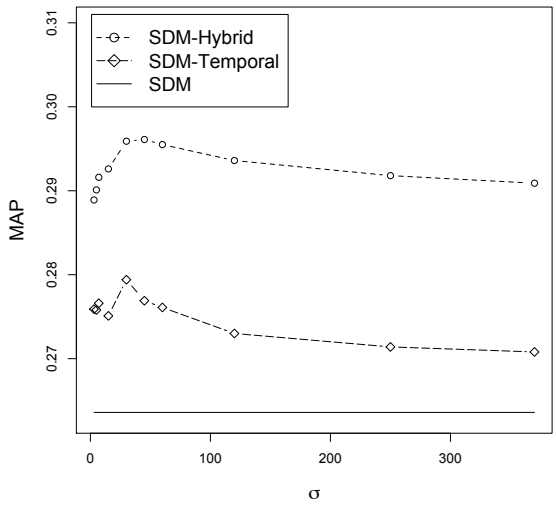

Figure 6.9. Effect of the parameter σ in the performance over TREC'09 data set

topical diversity scores. The third three rows show the performance of the methods using the temporal diversity scores followed by the last rows that show the performance of the hybrid diversity scores.

Statistical significance tests are performed using the paired T-test at 0.05 level of significance. The symbol ↑ indicates that a diversity method has a statistically significant improvement over its corresponding baseline. The symbols △ and ▲ show that the hybrid diversity method has a statistically significant improvement over the temporal diversity and topical diversity methods respectively. The bold values in each column indicate the best performance for the corresponding evaluation measure.

As we can see in the tables, diversity-based methods generally improve their corresponding baseline methods. In most cases, these improvements are at a statistically significant level. The temporal diversity methods are not as effective as their topical diversity counterpart. However, when combined together, the resulting hybrid diversity methods usually produce the best results. In some cases, combining temporal diversity with topical diversity results in a statistically significant improvement over each one of them.

An interesting observation is that the *SDM* method, as the strongest baseline, benefits the most from the diversification. As a result, the *SDM-hybrid* method significantly outperforms the *SDM* method in most of the collections and evalu-

ation metrics.

We previously mentioned that spam blogs are one category of non-relevant blogs that would be filtered using diversity-based methods. It is interesting to see what portion of blogs that are affected by diversity methods are in fact spam blogs. To this end, we manually checked non-relevant blogs that were initially retrieved by SDM and were removed from the ranked list after applying hybrid diversity measure. We chose ten random queries from the TREC'09 query set and compared their ranked lists before and after considering diversity. Among 106 non-relevant blogs that were removed from the ranked list, 27 of them (25%) were spam blogs and the rest were non-spam. This shows that considering diversity does not just filter out spam blogs, but also removes other non-relevant blogs from the ranked list. On the other hand, our examination shows that the method promotes relevant blogs in the ranking. For the examined queries, there were 30 new relevant blogs retrieved after applying diversity and only 6 relevant blogs were removed from the ranked list.

In order to test the robustness of proposed approaches, we analyze the sensitivity of their retrieval performance to the parameters. We analyze the results over the TREC'09 query set, as an example of a large collection, and the TREC'07 query set, as an example of a small collection. For simplicity, we only consider the *SDM* method and the corresponding diversity-based methods in this analysis. To study each of the parameters, we fix the other parameters with values that were learnt from the training set in the previous experiments.

Figures 6.4 and 6.5 show the effect of n, the number of initially retrieved posts, on the performance of retrieval systems. The diversity-based methods outperform the baselines at all the values of n. This confirms our analysis in section 6.2 that the difference between on-topic diversity of the relevant blogs and the non-relevant blogs does not change by increasing n. We can see that by increasing the number of posts, the performance increases and the difference of performances for values higher than 5000 are insignificant.

Figures 6.6 and 6.7 show the effect of λ when we fix the values of n and σ. We can see that the performance of the hybrid and topical diversity methods generally increase by increasing λ. In both collections, the best performance of these methods is achieved when λ has a value close to one. This shows that the maximum penalty for those blogs that publish repetitive information produces the best results. On the other hand, temporal diversity is less robust and the performance decreases for λ values higher than 0.5.

Finally, figures 6.8 and 6.9 show the effect of σ on the performance of temporal and hybrid diversity methods. It can be seen that σ is less influential on the performance and it has little negative effect when its value increases. It is

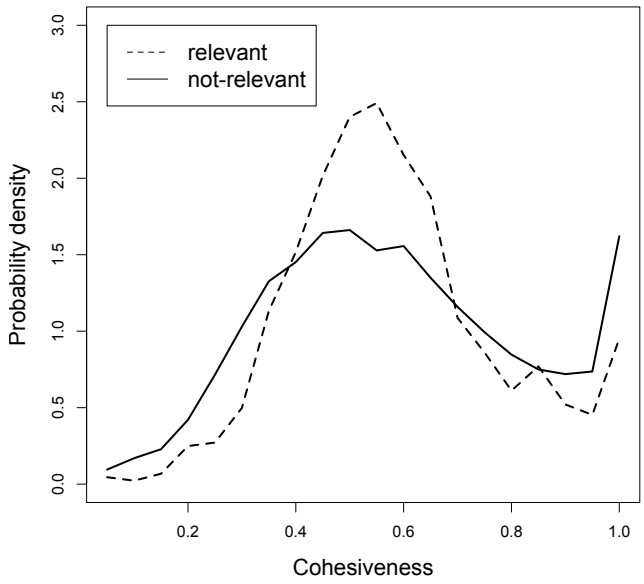

Figure 6.10. Cohesiveness distribution of relevant and non-relevant blogs.

interesting to see the relation between the size of the collection and the best value of σ. While for the TREC'07 collection the best performance is achieved when σ is around 5, this value for the TREC'09 collection is around 40.

Considering runtime of the diversity algorithms, the main overhead of the proposed algorithm is calculating the similarity between every pair posts of the blog. This calculation might take long time if number of posts in each blog is high, however we perform calculation only on those posts that are related to the query topic. Those topic related posts are extracted from an initial ranked list of posts for the query. We saw that by limiting the size of the initial rank list to less than 2000, we get fairly good improvement and would not need to have a longer initial list. Using the short initial ranked list, number of posts that are retrieved from each blog in the initial list would be very small and calculating similarity between them would not take long time. In our experiment over the TREC'09 query set, each blog has on average less than 10 posts in the initial list that results in 45 similarity calculation for each blog. Although calculating similarity can increase the runtime of the algorithm,

6.3.1 Diversity vs. Cohesiveness

So far, we showed that blog post diversity is a discriminative feature in blog retrieval that can improve retrieval performance. In this section, we investigate a comparison between blog post diversity and the previously proposed blog cohesiveness measures. The cohesiveness of a blog is assumed to show whether the blog focuses on a specific topic or not. While this assumption seems reasonable, the proposed methods in previous work did not show any positive effect on retrieval effectiveness:

- MacDonald and Ounis suggested that high cohesiveness of a blog shows that most of the blog posts are about similar topics and thus we should give higher score to that blog. Adding their cohesiveness measure to the weighting model not only did not improve their results but also decreased the results in some cases [MO08].

- Seo and Croft defined a penalty factor based on a clarity score [SC08]. They assumed that a blog that covers many different topics has a language model similar to that of the collection and thus should be penalized. They added the clarity-based penalty to two weighting models and in both cases it decreased the performance of the system.

- Elsas *et al.* proposed a centrality measure for each post to calculate $P(p_{ji}|b_i)$ in equation 2.1 [EACC08]. The centrality captures how similar the post is to the blog as a whole. The assumption is that if the blog is cohesive, then the centrality of the posts is high and thus the overall score of the blog should be higher. Our experiments show that replacing the centrality measure with a simple uniform distribution over blog posts improves the retrieval performance[1].

All these works assumed cohesiveness to be a positive feature of blog relevance and did not investigate if this is a valid assumption or not. In order to verify the validity of this assumption, we use the following cohesiveness measure to compare the relevant and non-relevant blogs:

$$cohesiveness(b_i, q) = \frac{\sum_{j=1}^{|b_i \cap R(q)|} sim(p_{ji}, VD(b_i))}{|b_i \cap R(q)|}$$

[1]For clarity of the paper we do not report this comparison here.

where $VD(b_i)$ is a virtual document created by concatenating all the posts of b_i. We use the cosine similarity measure for calculating $sim(p_{ji}, b_i)$, as the similarity between a post and a blog. This measure of cohesiveness compares the initially retrieved posts of a blog to the blog as a whole. In other words, it captures the cohesiveness of the blog with respect to the topic. It is similar to those measures used in other works mentioned earlier [MO08; EACC08].

Figure 6.10 shows the distribution of cohesiveness values for the relevant and non-relevant blogs in TREC'09 data set. The value of n, the size of $R(q)$, is set to 15000 for this analysis. The mean value of cohesiveness for the relevant blogs is 0.56 while for the non-relevant blogs, it is 0.55. The difference between the mean values is not statistically significant and the p-value of the Student's t-test is 0.66. This shows that the cohesiveness is not a strong discriminative feature and explains why adding cohesiveness did not improve the performance of the retrieval systems in the previous works.

It is interesting to verify if cohesiveness is correlated with *IOS*, as defined in equation 6.1. If a blog had high cohesiveness, it is not surprising that it would also have high similarity among its posts and thus high *IOS* value. However if the top posts of a blog are very similar (high *IOS*), one can not directly conclude that the blog has high cohesiveness. In order to examine such a correlation we calculate the Pearson correlation between the cohesiveness and the *OIS* values. The correlation is statistically significant with a value of `0.83`. The high correlation shows that blogs with high *OIS* are very likely to also have high cohesiveness.

6.3.2 Diversity in Facet Detection

In 2009, a more complex and refined version of the blog distillation task was introduced in TREC, named "Faceted Blog Distillation" [MOS09]. The new task, which was first introduced by Hearst *et al.* [HHD08], aims to consider not only the blog relevance to the topic but also the "quality aspects" (facets) of the blog. For each query, a specific facet is determined and only blogs that satisfy that facet are considered to be relevant. In this section, we discuss the effect of diversity on the performance of the system for different facets. Different facets can be seen as different categories of blogs that users might search for. It is important to be able to decide for what type of information needs we can use diversity to get the maximum overall performance.

We use the facets introduced in the TREC'09 and TREC'10 data collections which include Opinionated vs. Factual, Personal vs. Company and In-depth vs. Shallow facets [MOS09]. Tables 6.6 and 6.7 compare the performance of the diversity method with the baseline method for each facet. Without further

Facet	Model	MAP	P@10	Bpref
Opinionated (13 queries)	SDM	0.1153	0.1615	0.0998
	SDM-hybrid	0.1236 (+7%)	0.1923 (+19%)	0.1132 (+13%)
Factual (13 queries)	SDM	0.1749	0.1692	0.1570
	SDM-hybrid	0.1788 (+2%)	0.1538 (-9%)	0.1661 (+5%)
Personal (8 queries)	SDM	0.1840	0.2000	0.1384
	SDM-hybrid	0.2201 (+19%) ↑	0.2375 (+18%)	0.1756 (**+26%**) ↑
Official (8 queries)	SDM	0.1876	0.1750	0.1537
	SDM-hybrid	0.2319 (**+23%**)	0.1750 (0.0%)	0.1819 (+18%)
Indepth (18 queries)	SDM	0.2723	0.2222	0.2624
	SDM-hybrid	0.2872 (+5%)	0.2667 (**20%**) ↑	0.2526 (-3%)
Shallow (18 queries)	SDM	0.1348	0.0889	0.1118
	SDM-hybrid	0.1540 (+14%) ↑	0.1056 (+18%)	0.1378 (+23%)

Table 6.6. Effect of diversity on different facets over TREC'09 data set.

Facet	Model	MAP	P@10	Bpref
Opinionated (15 queries)	SDM	0.1061	0.1800	0.1257
	SDM-hybrid	0.1260 (+18%) ↑	0.2000 (+11%)	0.1369 (+8%)
Factual (15 queries)	SDM	0.1028	0.1000	0.0820
	SDM-hybrid	0.1190 (+15%)	0.1067 (+6%)	0.0985 (**+20%**)
Personal (15 queries)	SDM	0.1290	0.1533	0.1095
	SDM-hybrid	0.1426 (+10%)	0.1467 (-4%)	0.1060 (-3%)
Official (15 queries)	SDM	0.2057	0.1267	0.1697
	SDM-hybrid	0.2274 (+10%)	0.1400 (+10%)	0.1928 (+13%)
Indepth (16 queries)	SDM	0.2396	0.1250	0.2079
	SDM-hybrid	0.2953 (**+23%**) ↑	0.1625 (**+30%**) ↑	0.2430 (+16%)
Shallow (16 queries)	SDM	0.0833	0.1250	0.0818
	SDM-hybrid	0.1003 (+20%)	0.1313 (+5%)	0.0838 (+2%)

Table 6.7. Effect of diversity on different facets over TREC'10 data set.

Model	MAP	P@10	Bpref
SDM	0.2867	0.4444	0.3439
SDM-hybrid	0.3191 ↑	**0.5156** ↑	0.3622 ↑
SDM-hybrid-BL	**0.3233** ↑	0.4933 ↑	**0.3693** ↑
BloggerModel	0.3441	0.5333	0.3903
BloggerModel-BL	0.3581 ↑	0.5689 ↑	0.4001 ↑

Table 6.8. Evaluation results for blog-level penalty over TREC'07 data set.

Model	MAP	P@10	Bpref
SDM	0.2096	0.3740	0.2699
SDM-hybrid	0.2329 ↑	**0.3840**	**0.2777**
SDM-hybrid-BL	**0.2424** ↑	0.3740	**0.2812**
BloggerModel	0.2511	0.4180	0.3006
BloggerModel-BL	0.2625	0.4280	0.3004

Table 6.9. Evaluation results for blog-level penalty over TREC'08 data set.

tuning, we use the same parameter values as the previous section.

The statistically significant improvements are shown by ↑. Since each facet has very few queries, the statistical significance tests will be very sensitive and will not always show a difference. Thus we report also the percentage of improvement over the baseline method. The maximum improvement for each measure is shown in bold.

As we can see, diversity improves the performance of the system for almost all of the facets and evaluation measures. This shows that the effectiveness of diversity is not restricted to specific categories of topics. As a result, one can expect to have improvements by applying diversity for any type of information needs in blog search.

It is interesting to notice the improvement of P@10 for the in-depth queries which is the maximum among all the facets. For the in-depth queries, a relevant blog is expected to have in-depth thoughts and analyses about the topic [MOS09]. As we previously saw in example 3 in table 6.1, one of the scenarios where diversity is effective is in retrieving blogs with such an in-depth property. The obtained improvements for in-depth queries confirm our previous observation. The results show that a diversity-based method retrieves in-depth blogs at the top ranks and significantly improves the P@10 measure.

Model	MAP	P@10	Bpref
SDM	0.2636	0.3821	0.2858
SDM-hybrid	**0.2961** ↑	0.4308 ↑	**0.3125** ↑
SDM-hybrid-BL	**0.2909** ↑	0.4385 ↑	**0.3066**
BloggerModel	0.2764	0.4154	0.2899
BloggerModel-BL	0.2896 ↑	0.4205	0.3003 ↑

Table 6.10. Evaluation results for blog-level penalty over TREC'09 data set.

Model	MAP	P@10	Bpref
SDM	0.2273	0.3022	0.2273
SDM-hybrid	**0.2571** ↑	**0.3500** ↑	**0.2649** ↑
SDM-hybrid-BL	**0.2443** ↑	0.3239 ↑	**0.2545** ↑
BloggerModel	0.2251	0.3000	0.2344
BloggerModel-BL	0.2355 ↑	0.3087	0.2401

Table 6.11. Evaluation results for blog-level penalty over TREC'10 data set.

6.3.3 Diversity as an Independent Component

In the previous experiments, we employed diversity in the methods that estimate blog relevance score as an aggregation of the post-level relevance scores. Consequently we penalize the post scores in order to take diversity into account. However, there are some methods that do not perform aggregation over posts score. For example, the Blogger Model aggregates term probabilities in posts and estimate the term probabilities in a blog level. Based on the resulted term probabilities, it estimates the query likelihood directly at the blog level without computing posts relevance score. In such a situation, we can not penalize post scores and thus we need a more general method for integrating diversity into those blog retrieval methods.

One possible choice is to penalize blog score as a whole. To this end, we penalize blog relevance scores based on the average similarity of their posts. Average similarity of posts is calculated by OIS measure as defines in equation 6.1. The new score of a blog is given by:

$$Score_{diversity}(b_i, q) = score_{ini}(b_i, q)(1 - \gamma OIS(b_i, q)) \tag{6.6}$$

As apposed to previous methods, this method does not explicitly model the novelty of each post. However it gives us a measure for evaluating the average

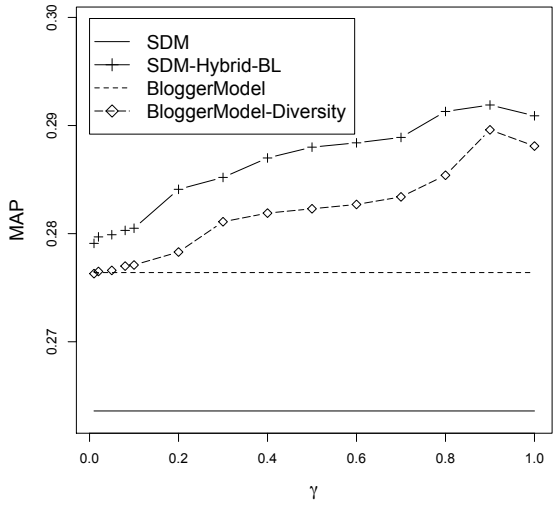

Figure 6.11. Effect of the diversity parameter λ in the performance over TREC'07 data set.

novelty of posts in the blog. This method can be seen similar to approaches based on portfolio theory in information retrieval where they take into account the risk of a ranked list in retrieving documents [WZ09]. Similar to the risk-aware ranking methods, we can assume the initial score of a blog as the gain of retrieving that blog and the *OIS* value as a measure of risk of retrieving a non-informative blog.

We apply this method on the scores of the Blogger Model for all the available data sets. In order to have a better comparison between the blog-level penalty and the post-level penalty, we apply this method to SDM scores. The results compared to the previous diversity methods can be seen in tables 6.8 to 6.11. We can see that the blog-level method can improve the results in most of the cases although it is not as effective as post-level penalty.

Same as before, the γ parameter is tuned using each of the data sets as the training set for its counterpart. Sensitivity of the system to γ is shown in figures 6.11 and 6.12. We can see that the proposed method can improve the Blogger Model for almost all values of γ while the best results are obtained by γ values close to one.

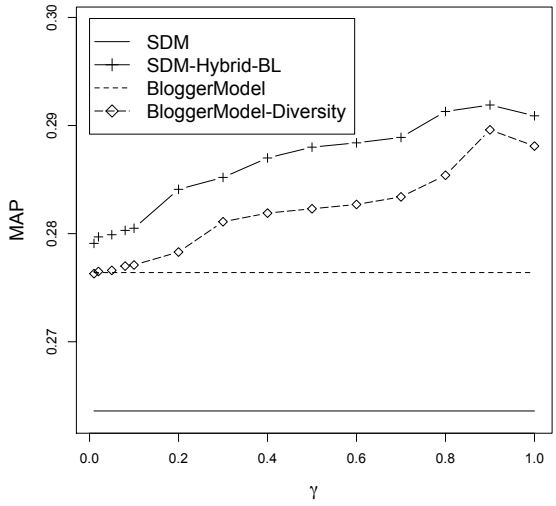

Figure 6.12. Effect of the diversity parameter γ in the performance over TREC'09 data set

6.4 Conclusion

In this section we studied the effect of employing different diversity measures on blog retrieval. We showed that diversity is an important feature that can help in distinguishing relevant blogs from non-relevant ones. We introduced three types of diversity and investigated their effect on the performance of the retrieval systems. Our experiments on standard blog retrieval collections showed improvements over different baseline methods.

Our main finding in this study suggest that topical and temporal diversities are positive indicator of relevant blogs. The topical diversity favors blogs that discuss about different subtopics and thus cover multiple subtopics of the query. On the other hand, temporal diversity favors blogs that discuss about query topic over entire time span, not just at some specific time windows. Overall, those blogs that discuss different subtopics with long span coverage are better choices for blog feed retrieval.

We further showed that the common assumption of cohesiveness is not an indicative feature of blog relevance and it is in fact the reason that there is no improvement observed in previous work. Finally, we investigated the effect of diversity on different types of queries and showed that diversity can be beneficial to all query types.

Chapter 7

Conclusions and Future Work

This chapter concludes our study and summarizes the answers to the research questions that were asked in chapter 1. Finally, it presents some possible directions for future work in this line of research.

7.1 Conclusions

This thesis investigated the problem of blog feed retrieval with an emphasis on temporal properties of blogs. We have proposed different approaches that can be applied on any stream ranking problem and evaluated their effectiveness on ranking of blogs over available standard data sets. Although we applied our methods on ranking blogs, they are applicable to other text streams that have similar properties to blogs, e.g twitter data. In summary, the findings of our research can be summarized as follows:

- Our study of smoothing techniques showed that similarity between blog posts can provide us with useful information that can be used to better estimate the relevance of a post to a given topic. This study was presented in chapter 4 and answers the research question RQ1 to RQ5.

  RQ1 examines whether the similarity between posts can be used in blog retrieval. We found that posts from relevant blogs are more similar to each other than to a post from a non-relevant blog. This is a good indicator that similarity between posts can be useful in distinguishing relevant blogs from non-relevant ones (RQ1). Based on this observation, we proposed methods by which we answered RQ2 and RQ3. We employed two smoothing methods in which we used similarity between posts as extra information for estimating their relevance score to a given topic. Both methods use

graph-based representations to capture the relation between posts. The first method changes the relevance score of a post based on its similarity to other posts (RQ2). This approach assumes that similar posts should have similar scores for a given topic. The second method smooths the probability estimation of terms in each post based on its terms in common with other posts (RQ3). We showed that both of these methods outperform the baseline methods. However, there is not a significant difference between the cost and performance of the two smoothing methods. The first method generates a smaller graph but needs similarity between posts to be provided. The second method does not need separate similarity values but generates bigger graphs due to adding extra node for each term.

Nest, we explored the temporal similarity between posts in order to answer RQ4 and RQ5. Besides content-based similarity between posts, we found that temporal similarity between posts can further help us in improving the retrieval performance of smoothing techniques (RQ5). By adding temporal information into the graph representations, we smooth the score of each post using other posts that have similar content and are published around the same data. In other words, we changed the similarity functions and only consider two posts similar when they have high values for both topical and temporal similarity measures (RQ4).

- In chapter 5, we investigated on the pseudo relevance feedback techniques for blog feed retrieval in which we integrated time as an important factor in the term selection process. In this chapter we tried to answer RQ6 to RQ8.

  In RQ6 we were interested in selecting query expansion terms based on time. We realized that the time factor can improve the query expansion process and the resulting expansion terms have higher qualities than traditional expansion methods. Our method assumes that for a given topic, a blogger picks a date and the selected date affects the term distribution of documents. Based on this assumption, once we estimated the importance of each day for the query, the expansion terms are selected proportional to two factors: 1) the term weight in a day and 2) the importance of the day for the query. We estimated the date importance based on the number of top posts published in each day. The term distribution of each day is estimated using the term distribution of posts published on that specific day.

  We further explored time-based query expansion techniques and proposed

a framework that answers RQ7 and RQ8. Our proposed framework, named TEMPER, not only selects expansion terms based on time but also finds blogs that use those terms at the corresponding time points. As opposed to traditional single expanded query, TEMPER generates multiple expanded queries for different time points. To this end, similarly to other pseudo relevance feedback techniques, this technique assumes that the top scored posts at each time are relevant to the query and use those posts for extracting expansion terms. Once term distributions are estimated for the dates of observed data, TEMPER uses kernel methods to estimate term distribution for all other dates. Using kernel methods, we assume that term distribution in each date is more similar to term distribution of closer dates than dates that are far away. The result of our estimations is a time-based representation of both blogs and queries (RQ7).

Having the new time-based representations enables us to use time-series similarity measures to calculate similarity of a blog to a query. We used two simple similarity measures for this purpose. The fist measure sums the daily similarities between the two series to calculate the similarity score. The second measure considers each time-series as a point in a multi-dimensional space with days as dimensions. The Euclidian length of the similarity vector between the two time series is then considered as the similarity value (RQ8). Our experiments showed that the Euclidian similarity measure is a better measure for blog retrieval.

TEMPER enables us to retrieve blogs that are both relevant and up-to-date for a given topic. Our experiments across different data sets with different settings showed improvement over existing query expansion methods for blog feed retrieval.

- Finally we studied diversity and novelty of blog posts and their effect on blog relevance in chapter 6. This chapter answered research question RQ9 to RQ12.

  In RQ9 we asked whether blog posts diversity is an important factor in blog relevance. We found that, as opposed to the usual assumption of cohesiveness of relevant blogs, relevant blogs are more likely to have high diversity and thus low cohesiveness among their posts. Based on this observation, we proposed methods that consider diversity of blog posts in the scoring model. Our main assumption was that novel information should contribute more in the score of the blog than repetitive already seen information.

We introduced three different diversity measures including topical diversity, temporal diversity and hybrid diversity (RQ10). Topical diversity examines whether the content of blog posts are similar to each other. Temporal diversity measures if blog posts are published all over the time or concentrated in some specific time intervals. Hybrid diversity considers both temporal and topical diversities and only penalizes blogs that publish similar content repetitively at some limited time points. We showed that diversity measures can be easily integrated into existing blog retrieval methods. To this end, we either penalize post scores before aggregating them for blog score or directly penalize blog score (RQ11). Our experiments over different blog collections and different baseline methods showed that diversity-based methods can improve the performance of the retrieval system. Among the three types of diversity, hybrid diversity, that considers both topical and temporal diversities, achieves the best performance. We also found that while diversity-based methods are effective for all types of queries, they work the best for in-depth queries (RQ12).

During a manual examination of results, we found that diversity-based methods implicitly filter out spam blogs since they are very likely to publish similar content. However, spam blogs are not a large portion of the results and these methods also remove some non-spam blogs that are not relevant to the query.

7.2 Future Work

User generated data and specifically user generated streams have not sufficiently been explored yet. There are different directions related to these problems that seem promising and are worth investigating in the future. Among all the possible directions, the most attractive ones are the following:

- **Discriminative Models for Blog Feed Retrieval:** One of the immediate steps of the future work can be that of using discriminative models to combine all the available evidence for predicting relevance of a blog to a given query. We have multiple scores for each blog using different baseline methods and settings, e.g., one score using SDM with top 500 posts and another using 1000 posts. Also we have scores using our own methods like regularization, TEMPER or diversity-based methods. We can assume each of these scores as a feature of a blog that shows the strength of the blog relevance to the query. Thus we can use existing learning to rank methods,

like SVM-Map, to combine all those features and generate a final ranking of blogs. Since we have multiple data sets and relevance judgments, we can train our learning method using some of these data sets and test it on other data sets. A similar approach has been used by Macdonald and Ounis where they combine different outputs of voting techniques and generate a final ranked list of blogs [MO11b]. However, they do not consider other existing methods and use limited features that are only based on the voting models. Each method ranks blogs from different aspects and combining them might add extra information that improves the ranking system.

- **Similarity Measures for Blogs:** Like any other information retrieval system, the goal of a blog retrieval system is to rank some items in response to a given query. In any retrieval system having similarity between items can help us to extract useful information and improve the retrieval performance. Similarity between items can be used in different scenarios like clustering items, ranked list fusion, score regularization or smoothing.

  One of the interesting future works can be to investigate similarity measures between complex objects like blogs. Blogs, as collections of documents, are complex objects and calculating the similarity between them is more challenging than the similarity between two single documents.

  For any similarity measure, first we need to properly represent items. In case of blogs, the representation can be based on any of the existing models like the small document model, the large document model or the time series model. After having the representation we need to define a function that calculates similarity between two representations. For the similarity function we can choose one of the existing functions or define new methods that consider blog-specific properties into account. Finding the best representation and best similarity function and apply them to blog retrieval systems is an interesting direction for future work.

- **Exploiting other evidence like comments:** One of the information that we did not use in our methods are comments. Comments are important part of each post that can be used to enhance the representation of posts. For example, comments can help to better estimate term probabilities in posts or give us an idea how attractive the post is. These type of information can then be integrated in the retrieval system.

- **Analysing User-Generated Short Documents:** The task of analysing blog data can be expanded to shorter user generated contents like Twitter or

other conversational texts like chat rooms or forums. Besides designing specific retrieval methods for this data, analyzing the temporal dynamics of these streams is not well-explored yet.

- **Comparison of Streams Evolution:** Analysing the relationship between different streams like blogs, tweets, query logs and news streams can provide us with useful knowledge about the evolution of data on the web. For a given topic, we can estimate and compare the evolution of topics in different streams. It would be interesting to study which of the streams update faster in response to an external event and how it affect other streams.

  Knowing the relation between streams and the ability to predict changes in one of them based on the other streams can have different applications. For example if query logs are predictable based on other streams, a search engine can cache the results for possibly popular queries and decrease the response time for future requests. It can also help news agencies to select the best headlines based on the popular stories in social media. A similar problem is introduced in the Top Stories Identification (TSI) task as part of the blog track in TREC'09 where the goal was to determine how well the news stories are represented in the blogosphere [MOS09].

  The possibility to predict news headlines can be used in more complicated tasks like stock market forecast. Some studies show that news headlines can be used to predict stock market [LSL+00]. However, if we find that user generated data can predict news headlines, it can be a sign that user generated data can also predict stock market. In that case, by using user generated data instead of news headlines, we can save some time that is very valuable in stock market and we might even get more accurate predictions.

- **Beyond Content and Temporal Dependency:** The idea of exploiting dependency between documents can be directed toward considering other information, such as hyperlinks or user comments, as relations between posts. One can integrate these type of relations into smoothing frameworks and investigate their effect on retrieval performance.

- **Time and Diversity in Expert Search:** The developed models in this thesis can be applied on other retrieval applications where time plays an important role like expert search (ranking people based on their expertise in the given area). Similar to blogs, the interest of an expert for a given area can

change over time and we need to consider these dynamics in the ranking system.

Beside temporal analysis, diversity of documents can also be informative in expert search. In chapter 6 we showed that high diversity is a positive feature in blog relevance which means relevant blogs are more likely to publish novel and diverse posts. It is interesting to investigate this property in expert search problem. Although it might have different meaning in expert search collections, like academic publications. In those collections we can assume that every publication provides new information and thus low diversity does not necessarily mean that the author publishes repetitive information. In this situation, cohesiveness (low diversity) might show that expert is focused on the given topic and it can be a positive indicator of the expertise.

- **Multi-stream Summarization:** With the huge amount of information available on the Web, summarization becomes more important everyday. Multi-stream summarization is another direction that seems interesting. Having different types of streams like news, blogs and tweets, it would be interesting to have a summarized stream that would contain all the important information from different streams and reasonably integrate them together. Since time is an influential feature in all the mentioned streams, the generated summary should also take time into account and consider the dependency between consecutive documents.

- **Time-based Opinion Mining:** Opinion mining and polarity detection in user generated content have attracted interest from many researchers lately. One of the interesting directions that we would like to further explore is the temporal analysis of user opinions for a given topic. It is worthwhile to study how the user opinions for the topic change over time and how those changes correspond to external events. For example, companies would be interested to find how people react when the company makes a change in its services. Such studies can be done by analysing user opinions over time and can help companies to make better decisions.

Bibliography

[ACD+98] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, 1998.

[AECC08] J. Arguello, J. Elsas, J. Callan, and J. Carbonell. Document representation and query expansion models for blog recommendation. In *Proceedings of ICWSM'08*, pages 231–240, 2008.

[AEY+08] J. Arguello, J. L. Elsas, C. Yoo, J. Callan, and J. G. Carbonell. Document and Query Expansion Models for Blog Distillation. In *In Proceedings of TREC'08*, 2008.

[APL98] J. Allan, R. Papka, and V. Lavrenko. On-line new event detection and tracking. In *Proceedings of SIGIR'98*, pages 37–45, 1998.

[BAdR06] K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of SIGIR'06*, pages 43–50, 2006.

[BdRW08] K. Balog, M. de Rijke, and W. Weerkamp. Bloggers as experts: feed distillation using expert retrieval models. In *Proceedings of SIGIR'08*, pages 753–754, 2008.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[BSAS95] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using smart: Trec 3. *NIST SPECIAL PUBLICATION SP*, pages 69–69, 1995.

[BSdV10] K. Balog, P. Serdyukov, and A. P. de Vries. Overview of the trec 2010 entity track. In *Proceedings of TREC'10*, 2010.

[BV04] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of SIGIR'04*, pages 25–32, 2004.

[BYRN99] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[CdVR10] M. Clements, A. P. de Vries, and M. J. T. Reinders. The task-dependent effect of tags and ratings on social media access. *ACM Transaction on Information Systems*, 28(4):21, 2010.

[CG98] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR'98*, pages 335–336, 1998.

[CLC95] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In *Proceedings of SIGIR'95*, pages 21–28, 1995.

[CMS10] W. Croft, D. Metzler, and T. Strohman. *Search engines: Information retrieval in practice*. Addison-Wesley, 2010.

[CNGR08] G. Cao, J.-Y. Nie, J. Gao, and S. Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of SIGIR'08*, pages 243–250, 2008.

[Cre00] F. Crestani. Exploiting the similarity of non-matching terms at retrieval time. *Information Retrieval*, 2(1):23–43, 2000.

[CS07] N. Craswell and M. Szummer. Random walks on the click graph. In *Proceedings of SIGIR'07*, pages 239–246, 2007.

[CTZC02] S. Cronen-Townsend, Y. Zhou, and W. Croft. Predicting query performance. In *Proceedings of SIGIR'02*, pages 299–306. ACM, 2002.

[DD10] N. Dai and B. D. Davison. Freshness matters: in flowers, food, and web authority. In *Proceedings of SIGIR'10*, pages 114–121, 2010.

[DGI08] W. Dakka, L. Gravano, and P. G. Ipeirotis. Answering general time sensitive queries. In *Proceedings of CIKM'08*, pages 1437–1438, 2008.

[Dia05] F. Diaz. Regularizing ad hoc retrieval scores. In *Proceedings of CIKM'05*, pages 672–679, 2005.

[DJ04] F. Diaz and R. Jones. Using temporal profiles of queries for precision prediction. In *Proceedings of SIGIR'04*, pages 18–24, 2004.

[dJRH05] F. de Jong, H. Rode, and D. Hiemstra. Temporal language models for the disclosure of historical text. In *Proceedings of AHCÕ05*. Royal Netherlands Academy of Arts and Sciences, 2005.

[EACC08] J. L. Elsas, J. Arguello, J. Callan, and J. G. Carbonell. Retrieval and feedback models for blog feed search. In *Proceedings of SIGIR'08*, pages 347–354, 2008.

[ED10] J. L. Elsas and S. T. Dumais. Leveraging temporal dynamics of document content in relevance ranking. In *In Proceedings of WSDM'10*, pages 1–10, 2010.

[EG11] M. Efron and G. Golovchinsky. Estimation methods for ranking recent information. In *Proceeding of SIGIR'11*, pages 495–504, 2011.

[ETO07a] M. Efron, D. Turnbull, and C. Ovalle. University of Texas School of Information at TREC 2007. In *Proceedings of TREC'07*, 2007.

[ETO07b] M. Efron, D. Turnbull, and C. Ovalle. University of Texas School of Information at TREC 2007. In *In Proceedings of TREC'07*, 2007.

[EWdR07] B. Ernsting, W. Weerkamp, and M. de Rijke. Language modeling approaches to blog postand feed finding. In *Proceedings of TREC'07*, 2007.

[Fag89] J. Fagan. The effectiveness of a nonsyntactic approach to automatic phrase indexing for document retrieval. *Journal of the American Society for Information Science*, 40(2):115–132, 1989.

[GCC10] S. Gerani, M. J. Carman, and F. Crestani. Proximity-based opinion retrieval. In *In Proceedings of SIGIR'10*, pages 403–410, 2010.

[HHD08] M. A. Hearst, M. Hurst, and S. T. Dumais. What should blog search look like? In *Proceedings of SSM'08*, pages 95–98, 2008.

[HMP+07] D. Hannah, C. Macdonald, J. Peng, B. He, and I. Ounis. University of Glasgow at TREC 2007: Experiments in Blog and Enterprise Tracks with Terrier. In *Proceedings of TREC'07*, 2007.

[HT05] D. Hawking and P. Thomas. Server selection methods in hybrid portal search. In *Proceedings of SIGIR'05*, pages 75–82, 2005.

[HTBO09] B. Hedin, S. Tomlinson, J. R. Baron, and D. W. Oard. Overview of the trec 2009 legal track. In *Proceedings of TREC'09*, 2009.

[Hul96] D. Hull. Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society for Information Science*, 47(1):70–84, 1996.

[ICC10] G. Inches, M. J. Carman, and F. Crestani. Statistics of online user-generated short documents. In *Proceedings of ECIR'10*, pages 649–652, 2010.

[JD07] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Transactions on Information Systems*, 25(3):14, 2007.

[JKT05] A. Jatowt, Y. Kawai, and K. Tanaka. Temporal ranking of search engine results. *Web Information Systems Engineering*, pages 43–52, 2005.

[JW03] G. Jeh and J. Widom. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*, pages 271–279. ACM, 2003.

[KGC11] M. Keikha, S. Gerani, and F. Crestani. Temper: A temporal relevance feedback method. In *Proceedings of ECIR'11*, pages 436–447, 2011.

[KL05] O. Kurland and L. Lee. Pagerank without hyperlinks: structural re-ranking using links induced by language models. In *Proceedings of SIGIR'05*, pages 306–313, 2005.

[KP99] E. J. Keogh and M. J. Pazzani. Relevance feedback retrieval of time series data. In *Proceedings of SIGIR'99*, pages 183–190, 1999.

[KZ10] M. Karimzadehgan and C. Zhai. Estimation of statistical translation models based on mutual information for ad hoc information retrieval. In *In Proceedings of SIGIR'01*, pages 323–330, 2010.

[LC01a] V. Lavrenko and W. B. Croft. Relevance-based language models. In *In Proceedings of SIGIR'01*, pages 120–127, 2001.

[LC01b] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proceedings of SIGIR'01*, pages 120–127, 2001.

[LC03] X. Li and W. B. Croft. Time-based language models. In *Proceedings of CIKM'03*, pages 469–475, 2003.

[Liu09] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3:225–331, March 2009.

[LNK+08] Y. Lee, S.-H. Na, J. Kim, S.-H. Nam, H.-Y. Jung, and J.-H. Lee. Kle at trec 2008 blog track: Blog post and feed retrieval. In *Proceedings of TREC'08*, 2008.

[LNL09] Y. Lee, S.-H. Na, and J.-H. Lee. An improved feedback approach using relevant local posts for blog feed retrieval. In *Proceedings of CIKM'09*, pages 1971–1974, 2009.

[LSL+00] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, and J. Allan. Mining of concurrent text and time series. In *In proceedings of SIGKDD'00 Workshop on Text Mining*, pages 37–44, 2000.

[LZ01] J. D. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR*, pages 111–119, 2001.

[LZ09] Y. Lv and C. Zhai. Positional language models for information retrieval. In *Proceedings of SIGIR'09*, pages 299–306, 2009.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297. California, USA, 1967.

[McC02] A. McCallum. Mallet: A machine learning for language toolkit. In *http://mallet.cs.umass.edu*, 2002.

[MdR06] G. Mishne and M. de Rijke. A study of blog search. In *Proceedings of ECIR'06*, pages 289–301, 2006.

[MO06a] C. Macdonald and I. Ounis. The TREC Blogs06 collection: Creating and analysing a blog test collection. *Department of Computer Science, University of Glasgow Technical Report*, 2006.

[MO06b] C. Macdonald and I. Ounis. Voting for candidates: adapting data fusion techniques for an expert search task. In *In Proceedings of CIKM'06*, pages 387–396, 2006.

[MO08] C. Macdonald and I. Ounis. Key blog distillation: ranking aggregates. In *Proceedings of CIKM'08*, pages 1043–1052, 2008.

[MO11a] C. Macdonald and I. Ounis. Learning models for ranking aggregates. In *ECIR*, pages 517–529, 2011.

[MO11b] C. Macdonald and I. Ounis. Learning models for ranking aggregates. In *Proceefings of ECIR'11*, pages 517–529, 2011.

[MOS07] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC 2007 Blog Track. In *Proceedings of TREC'07*, 2007.

[MOS09] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC-2009 Blog Track. In *Proceedings of TREC'09*, 2009.

[MOS10] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC-2010 Blog Track. In *Proceedings of TREC'10*, 2010.

[MRS08] C. Manning, P. Raghavan, and H. Schutze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[MSOS10a] C. Macdonald, R. Santos, I. Ounis, and I. Soboroff. Blog track research at trec. In *ACM SIGIR Forum*, volume 44, pages 58–75. ACM, 2010.

[MSOS10b] C. Macdonald, R. L. Santos, I. Ounis, and I. Soboroff. Blog track research at trec. *SIGIR Forum*, 44(1):58–75, August 2010.

[MZZ08] Q. Mei, D. Zhang, and C. Zhai. A general optimization framework for smoothing language models on graph structures. In *Proceedings of SIGIR'08*, pages 611–618, 2008.

[New03] M. Newman. Mixing patterns in networks. *Physical Review E*, 67(2):26126, 2003.

[NRD08] S. Nunes, C. Ribeiro, and G. David. Feup at trec 2008 blog track: Using temporal evidence for ranking and feed distillation. In *Proceedings of TREC'08*, 2008.

[ODRM+06] I. Ounis, M. De Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC-2006 blog track. In *Proceedings of TREC'06*, pages 15–27, 2006.

[PBT05] J. Perkiö, W. Buntine, and H. Tirri. A temporally adaptive content-based relevance ranking algorithm. In *Proceedings of SIGIR'05*, pages 647–648, 2005.

[PC98] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR'98*, pages 275–281, 1998.

[PCG08] B. Poblete, C. Castillo, and A. Gionis. Dr. searcher and mr. browser: a unified hyperlink-click graph. In *Proceedings of CIKM'08*, pages 1123–1132, 2008.

[PLCB10] J. Parapar, J. López-Castro, and Á. Barreiro. Blog Posts and Comments Extraction and Impact on Retrieval Effectiveness. In *Proceedings of Spanish Conference on Information Retrieval*, 2010.

[PLR07] B. Pfahringer, C. Leschi, and P. Reutemann. Scaling Up Semi-supervised Learning: An Efficient and Effective LLGC Variant. In *Proceedings of PAKDD'07*, pages 236–247, 2007.

[RJ76] S. Robertson and K. Jones. Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146, 1976.

[RMC82] S. E. Robertson, M. E. Maron, and W. S. Cooper. The unified probabilistic model for ir. In *Proceedings of SIGIR'82*, pages 108–117, 1982.

[Rob77] S. Robertson. The probability ranking principle in ir. *Journal of documentation*, 33(4):294–304, 1977.

[Roc71] J. Rocchio. Relevance feedback in information retrieval. *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323, 1971.

[SA99] R. Swan and J. Allan. Extracting significant time varying features from text. In *Proceedings of CIKM'99*, pages 38–45, 1999.

[Sal71] G. Salton. *The SMART Retrieval System—Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.

[SAT09] M. Shokouhi, L. Azzopardi, and P. Thomas. Effective query expansion for federated search. In *Proceedings of SIGIR'09*, pages 427–434, 2009.

[SC03a] L. Si and J. P. Callan. Relevant document distribution estimation method for resource selection. In *Proceedings of SIGIR'03*, pages 298–305, 2003.

[SC03b] L. Si and J. P. Callan. Relevant document distribution estimation method for resource selection. In *Proceedings SIGIR'03*, pages 298–305, 2003.

[SC08] J. Seo and W. B. Croft. Blog site search using resource selection. In *Proceedings of CIKM'08*, pages 1053–1062. ACM, 2008.

[SCGJ95] B. Shaparenko, R. Caruana, J. Gehrke, and T. Joachims. Identifying temporal patterns and key players in document collections. In *Proceeedings of AMAST'95*, pages 165–174, 1995.

[SdVC06] I. Soboroff, A. de Vries, and N. Craswell. Overview of the TREC 2006 Enterprise Track. In *Proceedings of TREC'06*, 2006.

[SHL08] A. Sun, M. Hu, and E.-P. Lim. Searching blogs and news: a study on popular queries. In *Proceedings of SIGIR'08*, pages 729–730, 2008.

[Sin01] A. Singhal. Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, 24(4):35–43, 2001.

[SJ71] K. Sparck Jones. *Automatic keyword classification for information retrieval*. Butterworths London, 1971.

[SJ00] R. Swan and D. Jensen. Timemines: Constructing timelines with statistical models of word usage. In *Proceedings of KDD'00 Workshop on Text Mining*, pages 73–80, 2000.

[SMM+12] R. L. T. Santos, C. Macdonald, R. M. C. McCreadie, I. Ounis, and I. Soboroff. Information retrieval on the blogosphere. *Foundations and Trends in Information Retrieval*, 6(1):1–125, 2012.

[Sob04] I. Soboroff. Overview of the trec 2004 novelty track. In *Proceedings of TREC'04*, 2004.

[SRH08a] P. Serdyukov, H. Rode, and D. Hiemstra. Modeling expert finding as an absorbing random walk. In *Proceeding of SIGIR'08*, pages 797–798, 2008.

[SRH08b] P. Serdyukov, H. Rode, and D. Hiemstra. Modeling multi-step relevance propagation for expert finding. In *Proceeding of CIKM'08*, pages 1133–1142, 2008.

[SWY75] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communication of the ACM*, 18:613–620, November 1975.

[SZ08] A. Shakeri and C. Zhai. Smoothing document language models with probabilistic term count propagation. *Journal of Information Retrieval*, 11:139–164, 2008.

[WBdR08] W. Weerkamp, K. Balog, and M. de Rijke. Finding key bloggers, one post at a time. In *Proceedings of ECAI'08*, pages 318–322, 2008.

[WBdR10a] W. Weerkamp, K. Balog, and M. de Rijke. A two-stage model for blog feed search. In *Proceedings of SIGIR'10*, pages 877–878, 2010.

[WBdR10b] W. Weerkamp, K. Balog, and M. de Rijke. A two-stage model for blog feed search. In *In Proceeding of SIGIR*, pages 877–878, 2010.

[WBdR11] W. Weerkamp, K. Balog, and M. de Rijke. Blog feed search with a post index. *Inf. Retr.*, 14(5):515–545, 2011.

[WZ09] J. Wang and J. Zhu. Portfolio theory of information retrieval. In *Proceedings of SIGIR'09*, pages 115–122, 2009.

[XC99] J. Xu and W. B. Croft. Cluster-based language models for distributed retrieval. In *Proceedings of SIGIR'99*, pages 254–261, 1999.

[YLL04] P. S. Yu, X. Li, and B. Liu. On the temporal dimension of search. In *Proceedings of WWW'04 Alternate track*, pages 448–449, 2004.

[ZCL03] C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proceedings of SIGIR'03*, pages 10–17, 2003.

[Zhu04a] M. Zhu. Recall, precision and average precision. *Technical report, Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2004.

[Zhu04b] M. Zhu. Recall, precision and average precision. *Technical report, Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2004.

[ZL01a] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of CIKM'01*, pages 403–410, 2001.

[ZL01b] C. Zhai and J. D. Lafferty. Document language models, query models, and risk minimization for information retrieval. In *In Proceedings of SIGIR'01*, pages 403–410, 2001.

[ZL04] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22(2):179–214, 2004.