Proximity-based Approaches to Blog Opinion 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

Shima Gerani

under the supervision of

Fabio Crestani

September 2012

Dissertation Committee

Dissertation accepted on 13 September 2012

Research Advisor	PhD Program Director
Fabio Crestani	**Antonio Carzaniga** ***pro tempore***

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.

Shima Gerani
Lugano, 13 September 2012

Abstract

Recent years have seen the rapid growth of social media platforms that enable people to express their thoughts and perceptions on the web and share them with other users. Many people write their opinion about products, movies, people or events on blogs, forums or review sites. The so-called User Generated Content is a good source of users' opinion and mining it can be very useful for a wide variety of applications that require understudying of public opinion about a concept.

Blogs are one of the most popular and influential social media. The rapid growth in the popularity of blogs, the ability of bloggers to write about different topics and the possibility of getting feedback from other users, makes the blogosphere a valuable source of opinions on different topics. To facilitate access to such opinionated content new retrieval models called *opinion retrieval* models are necessary. Opinion retrieval models aim at finding documents that are relevant to the topic of a query and express opinion about it.

However, opinion retrieval in blogs is challenging due to a number of reasons. The first reason is that blogs are not limited to a single topic, they can be about anything that is of interest to an author. Therefore, a large number of blog posts may not be relevant to the topic of query. The second reason is that a blog post relevant to a query, can be also relevant to a number of other topics and express opinion about one of the non-query topics. Therefore, an opinion retrieval system should first locate the document relevant to a query and then score documents based on the opinion that is targeted at the query in a relevant document. Finally, blogs are not limited to a single domain, an opinion retrieval model should be general enough to be able to retrieve posts related to different topics in different domains.

In this thesis, we focus on the opinion retrieval task in blogs. Our aim is to propose methods that improve blog post opinion retrieval performance. To this end, we consider an opinion retrieval model to consist of three components: relevance scoring, opinion scoring and the score combination components. In this thesis we focus on the opinion scoring and combination components and propose methods for better handling these two important steps. We evaluate our propose methods on the standard TREC collection and provide evidence that the proposed methods are indeed helpful and improve the performance of the state of the art techniques.

Acknowledgements

First and foremost, I am grateful to my advisor, Dr. Fabio Crestani. I am really thankful to Fabio for guiding me while giving me the freedom to choose my research topic and explore different ideas. His constant encouragement, trust and support helped me overcome difficulties throughout this work. I am specially thankful to Fabio for providing the possibilities to attend conferences, summer schools and workshops whenever possible and necessary. He has also helped me to get in touch and collaborate with the most helpful and well known researchers in the field of Information Retrieval.

I am deeply grateful to Dr. Mark Carman who co-advised me during the first years of my Ph.D. Mark was my primary source for getting my science questions answered. He devoted a lot of time discussing different ideas around my research. I am very thankful to Mark for patiently and generously teaching me every basic knowledge I needed to carry out this research.

During my Ph.D. I visited the TIMan group at the University of Illinois at Urbana-Champaign where I had the honor of getting advise from Dr. ChengXiang Zhai. Research under Cheng's supervision has been one of my most fruitful experiences. His passion in research, teaching and advising students is really admirable. I am really thankful to Cheng for the time he devoted to the part of this thesis regarding score transformation and the innumerable lessons he though me.

I am grateful to the dissertation committee members, Dr. Fernando Pedone, Dr. Marc Langheinrich, Dr. Maarten De Rijke, and Dr. Mohand Boughanem, for their feedback and the time they spent examining this thesis. Their valuable comments made this thesis more complete.

I gratefully acknowledge the funding sources that made this work possible. This research was partly funded by the "Secrétariat d'état à l'Éducation et à la Recherche (SER)" and COST IC0702 "Combining Soft Computing Techniques and Statistical Methods to Improve Data Analysis Solutions". Part of this work that has been carried out in UIUC was funded by the Swiss National Science Foundation.

Former and current members of the Information Retrieval Group at the University of Lugano have contributed immensely to my personal and professional time in Lugano. The group has been a source of friendships as well as good advice and collaboration. I would like to specially thank Mostafa Keikha with whom I greatly enjoyed collaboration and discussing research problems. I am also very thankful to Dr. Monica Ladoni, Giacomo Inches, Ilya Markov, Parvaz Mahdabi, and Az Azrinudin Alidin who have always been present to discuss ideas and help me in solving different problems that I encountered throughout this research.

I would like to thank the following graduate students at the University of Lugano for all the good times we shared and for their friendship: Parisa Jalili Marandi, Amirhossein Malekpour, Navid Ahmadi, Cyrus Hall, Marco D'Ambros, Nicolas Schiper, Nemanja Memarovic, Marcello Paolo Scipioni, Paolo Bonzini, Giovanni Ciampaglia, Mircea Lungu, Alex Tomic, Alessio Gambi and Alessandra Gorla, Marco Pasch, Lile Hattori, Konstantin Robinov, and many others.

I would like to thank Maryam Karimzadehgan for the great time we had and all the help she gave me while I was at UIUC. My thanks also go to Yuanhua Lv, Vinod Vydiswaran, and Duo Zhang for the useful discussions about my work at UIUC.

A special thank goes to the deans, Ph.D. directors, and staff members of the Faculty of Informatics at University of Lugano for providing information and assistance in arranging many things in the years I was a Ph.D. candidate.

I owe my warm gratitude to my family. To my mom, dad and my brother for their unconditional love and support throughout my years of studying. My gratitude goes also to my parents, sisters and brother in laws for their sympathy and support through out my Ph.D studies. I also thank my extended family who has encouraged me throughout this endeavor.

Most importantly, none of this would have been possible without my dear husband, Mehdi Mirzaaghaei. His constant love, encouragement and support during the past years helped me pass the difficult stages of this journey. Mehdi has shared the ups and downs of the entire Ph.D. process with me while working on his own doctoral thesis. He has given up so much to give my happiness and success a priority in our lives. This work is dedicated to him with all my love.

Preface

This thesis concerns the Ph.D. work done under the supervision of Prof. Fabio Crestani at the University of Lugano, from 2008 to 2012. It focuses on blog post opinion retrieval. The work presented in this thesis resulted in a number of publications which are listed below:

P1. Shima Gerani, Mostafa Keikha, Mark James Carman, Robert Gwadera, Davide Taibi, and Fabio Crestani. **University of Lugano at TREC 2008 Blog Track**. In Proceedings of TREC, 2008.

P2. Shima Gerani, Mark J. Carman, and Fabio Crestani. **Investigating learning approaches for blog post opinion retrieval**. In Proceedings of the 31th European Conference on IR Research, ECIR 09, pages 313–324, 2009.

P3. Mostafa Keikha, Mark James Carman, Robert Gwadera, Shima Gerani, Ilya Markov, Giacomo Inches, Az Azrinudin Alidin, and Fabio Crestani. **University of Lugano at TREC 2009 blog track**. In Proceedings of TREC, 2009.

P4. Shima Gerani, Mark James Carman, and Fabio Crestani. **Proximity-based opinion retrieval**. In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 10, pages 403–410, 2010.

P5. Mostafa Keikha, Parvaz Mahdabi, Shima Gerani, Giacomo Inches, Javier Parapar, Mark James Carman, and Fabio Crestani. **University of Lugano at TREC 2010**. In Proceedings of TREC, 2010.

P6. Shima Gerani, Mostafa Keikha, and Fabio Crestani. **Aggregating multiple opinion evidence in proximity-based opinion retrieval**. In Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 11, pages 1199–1200, 2011.

P7. Shima Gerani, ChengXiang Zhai, and Fabio Crestani. **Score transformation in linear combination for multi-criteria relevance ranking**. In Proceedings of the 34th European Conference on IR Research, ECIR 12, pages 256–267, 2012.

P8. Shima Gerani, Mark James Carman, and Fabio Crestani. **Aggregation Methods for Proximity-based Opinion Retrieval**, ACM Transactions on Information Systems (TOIS), Volume 30, Issue 4, 2012.

Contents

Figures

Tables

Chapter 1

Introduction

The advent of Web 2.0 had a huge influence on the way people communicate and exchange information. Social media platforms enable users to publish information on the web, connect to other users and communicate with them. Today, users can write about everything they like in blogs. People can connect to their friends and people of similar interests through social networking (e.g. Facebook, MySpace) or micro-blogging platforms (e.g. Twitter). People are no longer limited to their close friends and family for getting advice about their decisions as to what to buy, where to go, what to eat, etc. They can learn about others' experience and opinion by reading their blogs or reviews, checking their Facebook status updates or reading their tweets.

Mining the massive amount of opinion that is present in User Generated Content (UGC) on different platforms is very useful for a wide range of applications. Below we list some examples:

Product/Service Review Search: People usually seek for others' opinion online before buying a product or receiving a service. Opinion Mining can help in extracting the vast amount of opinion scattered in review sites, blogs or forums [6, 71] and make it accessible to users.

Business Intelligence: Researching the market to find product features that are attractive for customers or those that made the product or service unpopular is very helpful for manufacturers [40, 46, 83]. Opinion Mining can help in finding the relevant opinions of users to the product or feature of interest. Sale prediction [67] and reputation management [36] can be considered as other applications of tracking customers opinion [81].

Government Intelligence: Opinion mining on UGC can help government to better understand what the citizen's really need and want [4]. Government can track citizen's opinion on new systems or rules and change accordingly.

Politics: Understanding people's opinion is an important factor in political analysis. One popular application of opinion mining in politics is understanding what voters think [18, 31, 48, 73].

Applications mentioned above are all possible today due to the presence of social media. Blogs are one of the most popular and influential social media. According to the blog search engine Technorati.com, the blogosphere (i.e. the collection of all blogs on the internet) is doubling its size every six months. According to Technorati[1], from July 2006, over 175,000 new blogs and 1.6 million new posts are created each day. Bloggers (i.e. people who write blogs) write about their thoughts, experiences and opinions, on different topics. People who read the blogs can put comments and discuss their viewpoints with other users.

The rapid growth in popularity of blogs, the ability of bloggers to write about different topics and the possibility of getting feedback from other users, makes the blogosphere a valuable source of opinions on different topics. In fact, according to the analysis of a blog search engine query log, many blog search queries are related to uncovering opinion about a given subject [66]. To facilitate access to the opinionated content of blogs, new retrieval models called *opinion retrieval* models are indeed necessary. An opinion retrieval model ranks documents according to the degree that they are relevant to a topic and express opinion about it. Traditional Information Retrieval (IR) models are not sufficient for retrieving documents that express opinion on a topic since no opinion finding component is considered in those models. Using such IR models may result in retrieving blog posts that are relevant to the users's query but are not expressing any opinion about it, such as blog posts that are reporting a relevant news.

Opinion retrieval is more challenging in blogs compared to the other types of UGC such as products or movie reviews due to a number of reasons. The first reason is that a large number of blog posts are not relevant to the topic of user's query. Therefore, it is important to consider a topical relevance scoring component in an opinion retrieval system. The opinion retrieval model can use the topical relevance scores to filter non-relevant (very low scored) documents and to differentiate between documents with the same opinion scores but different topical relevance to the query.

The second reason is that blogs are not limited to a single topic. A blog post may be relevant to multiple topics but express opinion about only one of the topics. For example, consider a blog post in which the blogger writes about different Apple products such as "macbook pro", "iPad" and "iPhone" but only

[1]Technorati annual report on blogosphere's growth: http://www.sifry.com/alerts/archives/000436.html.

express his/her opinion about the iPad. Now consider a user that is looking for opinion about "macbook pro". In such scenario, the blog post is relevant to the user's query and contains some opinion, but the opinion is not about the topic of the user's query. Therefore, this document should not get a high score from a good opinion retrieval system.

Finally, blogs are topically diverse; they can be about any topic in any domain. Therefore, it is important for a blog post opinion retrieval system to be general and not specific to a single domain.

The need for new models of opinion retrieval that facilitate access to opinions about a topic, from topically diverse and unstructured UGC of blogs, is the main motivation for the research in this thesis.

In the rest of the chapter, we first present the research questions guiding this thesis. Next, we present the main contributions of this thesis. Finally we present an overview of the thesis and the publications resulted from the research presented in this thesis.

1.1 Research Outline and Questions

The opinion retrieval problem is usually tackled in 3 phases: in the first phase a standard IR system is used to rank posts by relevance, and the highest ranking posts are selected as a candidate set of documents. In the second phase, opinion scores are calculated for each candidate document, and in the third phase, the opinion and relevance scores are combined so as to produce a single ranking. A variety of different techniques have been employed in previous studies for realizing each of these phases and it is not clear from these studies to what extent learning (either in the opinion scoring phase or the ranking phase) or the feature selection method or the use of external datasets of opinionated terms/sentences/documents is responsible for the performance improvements demonstrated.

We start our research by focusing on the use of learning in the second and third phases of opinion retrieval and comparing learning techniques directly with baseline (non-learning) techniques. We limit our study to use only data available in TREC blog collection and ask:

RQ1 How can we make use of the information available in the collection to improve the opinion retrieval performance? In such settings, are learning methods more effective than non-learning methods in handling different steps of an opinion retrieval system?

This general research question leads to the following detailed questions:

- Is it possible to build an opinion lexicon based on the frequency of terms occurrences in relevant and opinionated documents compared to non-relevant ones?
- Having an opinion lexicon, are learning methods like SVM classification more effective than non-learning approaches for opinion scoring of documents?
- What is a proper way of combining relevance and opinion score of documents?
- Being limited to the information available in the collection, is it possible to achieve improvements in opinion retrieval performance over the topical-relevance retrieval baseline?

In Chapter 4 we seek answer to these questions by investigating different models of weighting terms based on the statistics of their occurrence in relevant and opinionated versus non-relevant documents. We also try simple learning and non-learning models of scoring documents regarding the amount of opinion expressed in them. Finally we try different methods for combining relevance and opinion scores to produce the final ranking. We evaluate different models experimentally and compare the performance with topical relevance retrieval.

To answer RQ1, we assume that the initial filtering of non-relevant documents to a topic is enough and all opinion expressions in a relevant document can be used to score the document. Next, we explore the validity of this assumption with the following research question:

RQ2 Assuming a document to be relevant to a query, can we consider all opinion expressions occurring in the document as targeted to the topic of the query? If no, how can we identify the relevant opinion expressions to the topic of the query?

More detailed questions are:

- How can we identify opinion expressions that are targeted at the topic of query?
- Is proximity of opinion expressions to the query terms helpful in identifying relevant opinion expressions?
- What would be a proper way to model proximity?

- How can we integrate proximity information into the document opinion scoring models?
- How does the proximity-enhanced opinion retrieval system perform compared to its non-proximity counterpart?

In Chapter 5 we explain the models, experiments and analysis that we carry out to answer the above mentioned questions.

The last step in an opinion retrieval process is ranking documents based on their score in terms of two criteria: 1) relevance to the topic expressed in the query and 2) opinion about the query. We look into the linear combination which is the most common approach for combining the scores of multiple criteria in multi-criteria IR. We investigate the necessity of score transformation before combination to increase compatibility between scores of different criteria and ask:

RQ3 What is an optimal way of transforming scores of different criteria before linear combination to increase their compatibility? To answer RQ3, we seek answers for the following more detailed questions:

- How can we formally characterize the optimality of transformation functions?
- How can we computationally achieve an optimal transformation in general for any multi-criteria ranking problem?
- Is the necessity of compatibility between different relevance aspect scores in linear regression supported by the experimental results in real applications?

In Chapter 6 we discuss these issues and propose a new principled approach to score transformation in linear combination, in which we would learn a separate non-linear transformation function for each relevance criteria. We then compare the proposed method with the state of the art score normalization methods. We also evaluate the method in terms of robustness with respect to score transformation.

In the rest of this chapter, we first list the contributions of this thesis. Then, we present the outline of the thesis.

1.2 Main Contributions

The main contributions of this thesis are as follows:

- **Comparing the effect of learning and non-leaning approaches in opinion retrieval:** by fixing different components of an opinion retrieval framework between learning and non-leaning approaches, we investigate the advantage of either approach in opinion scoring and score combination.
- **A novel probabilistic opinion retrieval model:** we propose an effective probabilistic model which uses the proximity between opinion lexicons and query terms as an indicator of their relatedness. We investigate the impact of different types of proximity functions in our model.
- **Different aggregation models to assign opinion score to documents:** we investigate different models for aggregating opinion evidence in order to assign an opinion score to a document.
- **A general principled approach to score transformation:** we analyze the incompatibility problem of scores in multi-criteria IR and discuss the necessity of scores transformation before applying the linear combination. We propose a general principled approach to score transformation based on well-studied and theoretically justified statistical models. We investigate the impact of different score transformations on the opinion retrieval task.

1.3 Outline of the Thesis

This thesis consists of three main research chapters, Chapters 4–6, each addressing a set of research questions mentioned earlier. Before presenting the research chapters, we have two general chapters which discuss the background, related work and experimental setup. Finally we present the concluding remarks in Chapter 7:

Chapter 2 - Background and Related Work: Here we present an introduction to IR in general and opinion retrieval in particular as a specific task which we address in this thesis. We explain the related work in the area of opinion retrieval and highlight the challenges that were not properly addressed in previous works.

Chapter 3 - Experimental Setup: In this chapter we provide information on the experimental settings used in the rest of this thesis such as test collections, topic sets and relevance judgements that we use in this thesis. We also explain different evaluation metrics that are commonly used to evaluate the quality of the rank list returned by a retrieval system. We further give details on the significance test that we use to compare different methods throughout this thesis.

Chapter 4 - Investigating Learning Approaches: In the first research chapter, we compare different models of weighting terms based on the statistics of their occurrence in opinionated and relevant documents compared to their occurrences in relevant but non-opinionated documents. Fixing the opinion lexicon, we compare learning or non-learning approaches in scoring documents for the opinion criterion. At the end, by fixing the opinion lexicon and opinion scoring method, we explore and compare different models of combining topical relevance and opinion scores.

[This chapter addresses RQ1.]

Chapter 5 - Exploiting Proximity Information: Getting insights from Chapter 4, we propose a novel probabilistic model that takes advantage of proximity information of opinion expressions to the query terms to better score the documents regarding their opinion targeted at the query. We highlight the importance of score normalization before combining relevance and opinion scores.

[This chapter addresses RQ2.]

Chapter 6 - Score Transformation for Multi-Criteria Relevance Ranking: In this chapter, we emphasize the compatibility of scores of different criteria before linear combination and propose a general model of score transformation in multi-criteria IR. We evaluate our proposed model and show its superiority to the state-of-the-art normalization methods.

[This chapter addresses RQ3.]

Chapter 7 - Conclusions: In the last chapter we provide tentative answers to the research questions introduced in this chapter. Finally, we discuss future directions of research.

Chapter 2

Background and Related Work

This chapter discusses the background and previous works related to the opinion retrieval task. We start with a general introduction to Information Retrieval in Section 2.1. We then give a brief description of common relevance retrieval models in Section 2.1.1. Relevance retrieval models are used in the first step of opinion retrieval to obtain a candidate set of relevant documents to be later scored and ranked based on their opinionatedness about the query. In Section 2.1.2, we discuss the evaluation methodology of Information Retrieval systems and give an introduction to the Text REtrieval Conference (TREC) which is a platform for evaluating Information Retrieval systems. The material until Section 2.2 provide a general background to the Information Retrieval and the evaluation mechanism used for evaluating Information Retrieval systems. To provide a more specific background of the topic and the proposed approaches in thesis, in Section 2.2, we explain the machine learning techniques that are used in the models proposed in different chapters of this thesis. We give a brief history of opinion retrieval, which is the main task addressed in this thesis, in Section 2.3. The opinion retrieval problem has been addressed as one of the tasks in the TREC Blog track. Thus, in Section 2.4 we explain the TREC Blog track, its different tasks and specifically the *blog post opinion retrieval task*. Finally we review the previous approaches to blog post opinion retrieval. In Section 2.5 we review the previous works for score combination in multi-criteria Information Retrieval. We notice that linear combination of criteria scores is the common approach. We review state-of-the-art score normalization methods for normalizing scores before combination.

2.1 Information Retrieval

Gerard Salton, a pioneer in information retrieval, defines information retrieval as follows [92]:

> Information retrieval is a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information.

The field of Information Retrieval (IR) deals with different applications related to search on a wide variety of information types such as text, image, video, audio and music. The most common application of IR is *web search* where someone types a query to a search engine and receives a ranked list of documents, present on the web, in response. Other examples of applications are *Vertical search*, *Enterprise search*, *Desktop search* and *Peer-to-peer search*. These are described below:

Vertical search is a specialized form of web search which focuses on a specific type of online content such as news, shopping, travel, etc.

Enterprise search involves searching information among computer files scattered across the intranet of an enterprise.

Desktop search is about searching among files in a personal computer.

Peer-to-peer search is about searching information in a network of nodes or computers without any centralized control.

Besides *search*, *filtering, classification* and *question answering* are other text-based tasks that are addressed in the field of IR [16].

Filtering is about finding and tracking information that can be interesting for a person and providing an alert using mechanisms such as email.

Classification automatically assigns documents to a predefined set of classes or labels.

Question answering aims at finding and returning an answer to a specific question instead of returning the whole document that contains the answer.

In this thesis, we work with text-based documents, mainly blog posts, and the task that we address is a *search-based* task.

An important concept in IR is *relevance*. A relevant document is a document that contains information that a person is looking for, when submitting the query to the search engine [16]. One may think that relevant documents can be retrieved by looking for exact match of the query terms in a document. However, there are some issues that make the task more difficult than what it looks like. The first issue is the *query mismatch problem*: different words can be used to express the same concept. Therefore, by looking for exact match of query terms some relevant documents can be missed. The other important issue is to differentiate between *topical relevance* and *user relevance*. A document is topically relevant to a query if it is on the same topic. However, a person who looks for the information, often called *user*, may not find every topically relevant document as relevant. User relevance may take other factors such as *recency*, *language* or *having opinion expressed about the query* into account. For example, if a user is looking for the homepage of the next upcoming conference, retrieving a five years old homepage of that conference is not satisfactory and the homepage is considered as non-relevant by the user. In this thesis we refer to such IR tasks as *multi-criteria* IR, since besides topical relevance, other criteria need to be satisfied in a document to be relevant.

Researchers propose *retrieval models* for retrieving documents relevant to the user's *information need*. According to Croft et al. [16], a retrieval model can be defined as a formal representation of the process of matching a query and a document. Traditional retrieval models focus on topical relevance of documents while other criteria may be also important to satisfy user's need. In the next section we review some of the main topical relevance retrieval models.

2.1.1 Topical Relevance Retrieval Models

Retrieval models can be classified into a number of classes called: *Boolean Model*, *Vector Space Model*, *Probabilistic Models* and *Language Models*. Below we give a brief description of each class:

Boolean Model The Boolean retrieval model, also known as *exact match retrieval*, was the first retrieval model used in the search engines. It specifies queries using Boolean logic operators AND, OR and NOT. The AND operator requires that a document contain all query terms to be relevant. The OR operator means that a document should contain at least one of the query terms to be relevant and the NOT operator specifies a term that is indicator of irrelevant documents and so, a relevant document should not contain it. For each document, the model returns TRUE or FALSE depending on whether it is relevant or irrelevant to the query. Thus, the output of a boolean retrieval model can not be directly used for ranking since it does not differentiate between relevant documents. However, an extension of the Boolean model is proposed by Joyce and Needham [44] that takes the term frequency of query terms into account to produce a ranking in a Boolean retrieval system. The limitations of the Boolean model in partial matching of a query and document, and also its inability to return a rank list of documents lead to the advent of new generation of retrieval models such as *vector space model*.

Vector Space Model The vector space model [94, 93] allows the partial matching of a query and a document; it represents both document and query in a t-dimensional vector space, where t is the number of terms in the vocabulary. Documents are then ranked based on the similarity with the query in that space. Different similarity measures have been proposed in the literature. The *cosine similarity*, i.e. the cosine of the angle between the two vectors of document and query, is the most successful one. Elements of the vectors can take binary as well as real values. Binary vectors indicate the presence or absence of terms in a query or a document. Different weighting methods have been proposed to represent documents or query vectors with real values. These weighting methods are based on the importance of the term in discriminating between relevant and non-relevant documents and its frequency in the document [95].

Probabilistic Models One of the most effective family of retrieval models is the *probabilistic models* [15]. It started from an idea in Maron and Kuhns' paper [65], where they mention that "since no retrieval system can be expected to predict *with certainty* which documents are relevant to the user's information need, systems must be dealing with *probabilities*" [88]. Based on this idea, Robertson developed the *Probability Ranking Principle (PRP)* [88] which encouraged the development of probabilistic retrieval models. The PRP states that:

> If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose, the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data.

On the basis of some assumptions, such as that the relevance of a document to a query is independent of other documents, it is possible to show that the above statement is true and the produced ranking is optimum. Ranking based on PRP requires estimation of the probability of relevance. Different models have been proposed for estimating the probability of relevance of a document to a query. The *binary independence model* [88, 89] is the first retrieval model in this direction. To estimate the probability of relevance, it assumes term independence and binary features in documents. These two assumptions are the intuitions behind the name of the model. The model was later extended to include term frequencies [90]. One of the most effective and popular probabilistic retrieval models is BM25 [101]. It extends the scoring function of the binary independence model by including the document and query term weights.

Language Models A Statistical Language Model (SLM) [91] is simply a probability distribution $P(s)$ over all possible s, where s can be any linguistic unit such as a word, sentence or document. Application of SLM started in speech recognition but later became popular in machine translation, document classification and routing, handwriting recognition, spelling correction, IR and many more. Using SLM in IR was first suggested by Ponte and Croft [84]. Based on their idea, both documents and the topic of query can be represented as language models. Therefore, we have three possibilities for retrieval models based on language models. The first is based on the probability of generating the query terms from a document language model. The second is based on generating the document based on a query language model and the third is based on comparing the language model of the query and documents. The three variations lead to three types of retrieval models named *query likelyhood*, *document likelihood* and *KL-divergence* retrieval models [47] respectively.

2.1.2 Evaluation of IR Systems

An important issue in IR research is *evaluation*. Evaluating the quality of IR systems is usually done thorough experiments; e.g. for a query, the ranked list of documents retrieved by a system from a collection is checked to see if the documents are relevant to the information need behind the query. Therefore, an important component in evaluating a retrieval system is a *test collection* composed of *a set of queries*, *a set of documents* and their *relevance judgment*; i.e. information about documents' relevance to each query.

Until 1992, different test collections such as the Cranfield collection [12], CACM collection [24] and NPL collection [104] were built and used by different research groups for evaluation purposes. However, available collections were relatively small. Therefore, the performance of systems on those small collection couldn't reflect the performance in real applications. In fact, a realistically-sized collection was missing [32].

Text REtrieval Conference

In 1992 the US Department of Defense and the National Institute of Standards and Technology (NIST), cosponsored the Text REtrieval Conference (TREC) [32, 33]. The goal was to provide the infrastructure that was necessary for the large scale evaluation of text retrieval technology and so faster transfer of retrieval technology to the commercial sector [107].

Now, TREC is an ongoing series of workshops, focusing on different text retrieval research areas called *tracks*. The main focus in the first years of TREC was on ad-hoc retrieval (i.e., given a query return a ranked list of relevant documents). However, new tracks were added as new research needs were identified. A complete list of current and past TREC tracks is available in [20]. Below we list some examples of TREC's tracks:

Blog Track: the purpose is to explore information seeking behavior in the blogosphere.

Legal Track: the goal is to develop search technology that meets the needs of lawyers to engage in effective discovery in digital document collections.

Microblog Track: the goal is to examine search tasks and evaluation methodologies for information seeking behaviors in microblogging environments.

Session Track: aims at providing the necessary resources in the form of test collections to simulate user interaction and help evaluate the utility of an IR system over a sequence of queries and user interactions, rather than for a single"one-shot" query.

Web Track: explores Web-specific retrieval tasks, including diversity and efficiency tasks, over collections of up to one billion Web pages.

Entity Track: the aim is to perform entity-oriented search tasks in the World Wide Web. These tasks are better answered by returning specific objects instead of just any type of document.

For every track, a list of tasks is defined that address different challenges related to the track. In particular, the following steps are done:

1. Track organizers provide a test set of documents and queries.

2. Participants try new retrieval methods and submit top-ranked documents, returned by their system, to NIST.

3. NIST judges the retrieved documents for relevance and evaluates the results.

4. NIST organizes the TREC workshop for participants to share their experiences.

Since it is very difficult to judge the relevance of all documents in the huge TREC collections, the pooling technique [100] is used when providing the relevance judgment of documents to queries for TREC. For pooling, the top X documents are retrieved by each system for a given topic and are merged into the pool for assessment. Note that in all ranked lists, those documents most likely to be relevant returned first.

2.2 Machine Learning and IR

IR provides a perfect class of applications for Machine Learning (ML) methods. Key IR processes are somehow based on classification tasks that are well explored in machine learning. Learning algorithms use examples, features and values, which IR tasks can provide. Documents can serve as examples, many linguistic features can be used and different weighting methods can serve as the feature values. An important factor in the success of ML techniques is the existence of adequate training data. Although for some ML approaches, called *unsupervised* approaches, we may not need labeled examples, for many applications we need to use *supervised* approaches which take advantage of labeled training data. For example, to classify review documents as positive or negative, we need to have examples of positive and negative reviews to train a classifier. Having adequate *labeled* training data is not always easy and this may limit the application of supervised ML techniques. We expect learning methods to perform better than non-learning methods, if enough training data is available.

In this thesis, we take advantage of a number of ML techniques namely *Logistic Regression*, *Support Vector Machine* and *Learning to Rank* which we explain below:

2.2.1 Logistic Regression

In statistics, regression analysis includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a response (dependent) variable and one or more predictor (independent) variables. In ML, regression is usually used to predict the value of the response variable given the value of predictor variables. Logistic regression is a type of regression used when the response variable is binary (0 or 1). In such cases, using regression to estimate the value of y is not useful. The reason is that the predicted value will be continuous and can get real values other than 0 or 1 which is hard to interpret. What can be useful though, is to predict the probability of value 1 for the response variable. For that, the predictor variables are regressed against log of the odd ratio of the probability of y taking value one. For example in case of having two predictor variable x_1 and x_2 and a response variable y, we have:

$$ln\frac{p(y)}{1-p(y)} = \alpha + \beta.x_1 + \gamma.x_2 \tag{2.1}$$

Based on Equation (2.1), and the estimated values of parameters α, β and γ, the probability of $y = 1$ can be obtained as follows:

$$P(y) = \frac{e^{\alpha+\beta.x_1+\gamma.x_2}}{1+e^{\alpha+\beta.x_1+\gamma.x_2}} \tag{2.2}$$

Due to the binary nature of relevance judgment that is usually available for IR tasks (i. e. a document is judged 1 meaning relevant and 0 meaning non-relevant), logistic regression seems like a right choice to predict the probability of relevance given some feature values of a document. In Chapters 5 and 6 of this thesis, we use logistic regression to predict the probability of relevance of a document based on its score from a retrieval model. We then use this probability instead of the original score from the retrieval models. Our aim is to normalize scores of different relevance criteria to the probability of relevance before combining them.

2.2.2 Support Vector Machine

Support Vector Machines, often called SVMs, are a group of supervised learning methods that can be applied to classification or regression. They treat data such as documents as points in some geometric space and then find a *hyperplane* (i.e. a generalized plane in N-dimensional space) that best separates the data points of two classes. Intuitively, the best separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class. To classify a new document, it is mapped into that same space and predicted to belong to a category based on which side of the hyperplane it falls.

If data points of two classes can be separated using a hyperplane, the data is called *linearly separable*. In case of having non-linearly separable data, the *Kernel trick* [1] can be used to simulate the computation in a higher dimensional space where data is linearly separable.

In the Chapter 4 of this thesis, we use a SVM classifier to classify blog posts as “opinionated and relevant” versus “relevant but not opinionated”.

2.2.3 Learning to Rank

The huge amount of data that is available these days, makes it possible to apply ML techniques to the problem of ranking documents. Learning to rank approaches [53] try to learn the best way of combining features. Features are usually related to document content, meta data, anchor text, links, etc. Scores of documents from different retrieval models like BM25 or Language Models can also serve as useful features. The training data for learning to rank methods is a set of queries and the desired ranking for each query. The advantage of the learning approach is that we do not need to decide explicitly how best to combine the forms of evidence, but can rely on historical data for fine tuning the retrieval. The best known learning to rank approaches such as *Ranking SVM* [42] and *SVMmap* [113] are based on an SVM classifier. Please note that the combination method in *Ranking SVM* and *SVMmap* is linear combination of feature values. The learning to rank methods help us in finding the best weight (i.e. coefficient) for each feature in linear combination.

In Chapter 4 of this thesis, we use SVMmap for combining the relevance and opinion scores of documents. In the rest of this chapter, we first give a brief history of opinion retrieval. We then continue with explaining the TREC Blog Track and the specific task of Opinion Retrieval which is the focus of this thesis in more details.

2.3 History of Opinion Retrieval

Research on opinion mining and sentiment analysis started mostly on review-type data with the intention to classify documents as expressing either a positive or negative opinion. The proposed approaches can be categorized into two main groups: lexicon-based [103, 102, 112, 37] and classification-based

[82, 17, 72, 26]. Both of these approaches rely on word occurrences. The first approach (lexicon based), uses a manually or automatically built list of subjective words, such as ‘good’ and ‘like’, and assumes that the presence of these words in a document provides evidence of document opinionatedness. A term’s opinion score can be used in different ways to assign an opinion score to the whole document. The second approach (classification-based) utilizes word occurrence and sometimes linguistic features and builds a classifier to classify positive (opinionated) and negative (non-opinionated) documents using Machine Learning techniques. Nevertheless, most of the early research in this area neglected the problem of retrieving documents that are related to the topic of the user’s query. It also did not target the problem of ranking opinionated documents according to the degree with which they were opinionated (either in positive or negative way). Relevance of an opinion to a topic was considered for the first time in Yi et. al [112] and then in Hurst and Nigam’s work [39] but they did not consider the ranking of documents. Instead they only classified documents as to whether they expressed an opinion about the topic. Opinion ranking was considered for the first time in Eguchi and Lavrenko’s work [19]. It then became an interesting research direction in TREC and other conferences such that a separate task at TREC Blog track was related to opinion retrieval and many papers were published that tried to address different challenges in this area.

In the next section, we first explain the TREC blog track and different tasks that were defined. We then focus on the opinion retrieval track and explain the challenges and proposed approaches in more details.

2.4 The TREC Blog Track

The Blog Track was introduced at TREC with the aim to explore the information seeking behavior in the blogosphere. It was introduced in TREC 2006 and continued to 2010. Various tasks have been investigated in the Blog Track namely *baseline adhoc (blog post) retrieval task, (blog post) opinion retrieval task, polarized (blog post) opinion finding task, blog distillation, faceted blog distillation* and *top news identification*. Below we can see a brief description of these tasks:

Baseline adhoc retrieval: aims at finding relevant blog posts to a query.

Opinion retrieval: aims at finding blog posts that are both relevant and opinionated about a query.

Polarized opinion retrieval: aims at finding blog posts that are relevant and express (positive/negative/mixed) opinion about a query.

Blog distillation: aims at finding blogs about a topic such that users can subscribe to and read from them regularly.

Faceted blog distillation: is an extension of the blog distillation task where besides relevance to the topic, the quality of blog such as being opinionated/factual, written by individual/company or in-depth/shallow discussion of topic, is taken into account.

Top news identification: investigates whether the blogosphere can be used to identify the most important news stories for a given day [59].

Since Opinion Retrieval task is the focus of this thesis, below we give a more detailed description of this task and review the proposed approaches in the literature for addressing this task.

2.4.1 The TREC Blog Post Opinion Retrieval Task

In TREC 2006, the *Opinion Retrieval* task appeared in the blog track. It considered the opinion mining problem from an IR perspective and introduced the problem of retrieving and ranking blog posts that were relevant and opinionated to a given topic [78]. From the above definition it is clear that a system should consider two aspects in retuning a blog post as relevant: 1) blog's *topical relevance* to the query and 2) its *opinionatedness* toward the topic of the query. The blog post opinion retrieval task ran as part of the TREC blog track from 2006 to 2008. A blog collection named Blog06 which consists of about 3.2 million blog post was made available to the community by the Blog track organizers and used for this task. At every year of running the opinion retrieval task, a set of 50 new topics were chosen by TREC organizers and added to the query set along with their relevance judgments. Relevance judgement indicate whether a blog post is relevant to a topic and if so, whether it expresses opinion (with a positive, negative, mix polarity) toward the query. More details about the Blog06 collection are provided in Chapter 3.

After three years of running the opinion retrieval task, the Blog06 collection with the set of 150 queries and their relevance judgment seemed to be good enough for assessing the quality of an opinion retrieval system. Therefore, the opinion retrieval task was stopped and did not continue during the TREC 2009 and TREC 2010 runs of TREC blog track. However, the resulting collection and query sets are still under active use by the researchers on the opinion retrieval problem [60]. There has been lots of research on blog opinion retrieval in TREC

[78, 58, 80] and other venues [120, 111, 117, 97, 106, 68] in which people follow the opinion retrieval definition used in the TREC blog track.

A common approach in blog post opinion retrieval is to first rank documents based on their topical relevance to the query and then re-rank the top *n* retrieved documents by taking their opinion score into account. Studies have shown that the performance of opinion retrieval systems is strongly dependent on the performance of the underlying topical relevance retrieval method. A strong topical relevance retrieval system leads to a stronger opinion retrieval system than a system that has a weak topical relevance component. In fact, a system without any opinion finding component, only based on topical relevance retrieval, can still perform well in terms of finding opinionated blog posts. The reason is that blogs are opinionated in nature and the majority of blog posts that are relevant to a topic are also expressing an opinion about it. Thus, in order to evaluate the performance of opinion retrieval systems, it is natural to compare the performance of the final system with performance of the underlying topical relevance method. It has been shown to be very difficult to improve the ranking produced by a strong topical relevance method, using any opinion finding technique [56]. This means that an opinion scoring technique may be able to improve the performance of a topical relevance ranking method remarkably, while fail to improve the performance of another relevance ranking system. Therefore, comparing the performance of opinion scoring methods is not fair, if systems are based on different relevance retrieval methods.

In order to facilitate direct comparison between systems during TREC 2008, five relevance retrieval baselines were provided by the TREC organizers, selected from the best performing retrieval runs. Each of these baselines covers all 150 topics and contains a ranking of the top 1000 relevant documents for each topic [80]. Having a common relevance baseline, researches can deploy their opinion finding model on top of a TREC relevance retrieval baseline and easily compare the effectiveness of different opinion finding models.

In the following, we review the effective techniques published in the literature for the opinion retrieval task. Here we do not provide details on the first relevance ranking step of opinion retrieval. Instead we focus on an overview of the techniques that were used in opinion finding and in producing the final ranking which is based on the combination of relevance and opinion scores. These approaches can be categorized to two classes of lexicon-based and classification-based.

2.4.2 Classification-based Opinion Retrieval

Classification-based opinion retrieval systems use resources that are known to be subjective(i.e. opinionated) or objective(i.e. non-opinionated) as training data to build a subjectivity classifier.

W. Zhang et al. [120] decompose documents that are retrieved in the topical relevance step, into sentences. They learn an SVM classifier which labels each sentence as opinionated or not and gives an opinion score to the sentence. They propose different methods for scoring documents for their opinion aspect based on the number and score of opinionated sentences that they have. To find relevant opinionated sentences to the query, they use a NEAR operator which considers opinionated sentences that are close to query terms as relevant opinionated sentences. Finally relevance and opinion scores of documents are combined linearly to ensure satisfying the both aspects of topical relevance and opinionatedness in every document. To build the subjectivity sentence classifier, they use resources other than the Blog06 data for training, including subjective documents collected from review web sites like epinion.com[1] and rateitall[2]. They also submitted queries to a Web search engine containing opinion indicator phrases such as “reviews”, “don’t like", etc. and considered the top retrieved documents as opinionated documents. They used the same approach for collecting not opinionated documents from dictionary sites like Wikipedia, and by submitting queries to a search engine for documents that did not contain the terms “review", “comments", etc. They used uni-grams and bi-grams in the collected documents as the features for the SVM classifier, and used the Chi-Square method to reduce the number of features. Later, W. Zhang et al. refined their work by extracting some document level features for a decision tree classifier based on the output of a SVM sentence classifier. The decision tree classifier was then used to score the whole document [119]. Q. Zhang et al. [118] used the CME (Classification by Minimizing Error) method to assign an opinion score to each sentence of a blog post. They then defined some features based on subjectivity and relevance of all sentences in the post. An SVM classifier is used to assign an opinion score to each document based on the values of the defined features. A subjectivity data set of movie-review data containing 5000 subjective and 5000 objective sentences was used for training the CME classifier, but for the SVM and classifying documents they just used the Blog06 collection. The blog posts were ranked by the final score that was calculated as the product of the relevance and the opinion scores.

[1]http://www.epinions.com
[2]http://www.rateitall.com

Another classification based opinion finding system is proposed by He et al. [35]. They use OpinionFinder [108] to classify sentences of a blog post as subjective or objective. They then use the proportion of subjective sentences in a blog post and the confidence of the classifier to assign opinion score to documents. A function of the opinion score is then linearly combined with the relevance score to produce the relevant opinion score for a final ranking. OpinionFinder is a freely available toolkit for identifying subjective sentences in text. It uses two Naïve Bayes classifiers to distinguish between subjective and objective sentences using a variety of lexical and contextual features. The classifiers have been trained using subjective and objective sentences, which have been automatically generated from a corpus of un-annotated data by rule-based classifiers [86].

Classification-based opinion retrieval methods require a lot of training data to learn a precise opinion scoring model. However, they have the advantage of being very effective in case enough training data is available and good discriminative features are used in the model.

2.4.3 Lexicon-based Opinion Retrieval

In the Lexicon-based approach to opinion retrieval, a list of opinion terms called *opinion lexicon* is used to find opinion expressions in a document. Different methods are proposed to create a general, domain or topic-specific opinion lexicon. Different methods are also proposed to assign opinion score to a document that take advantage of different factors such as the frequency and position of opinion terms in a document and their relatedness to the topic of interest. In Chapter 5 of this thesis we present a lexicon-based opinion retrieval model that uses an external opinion lexicon and takes advantage of the proximity of opinion terms to the query terms to estimate their relatedness. We then incorporate this information in assigning an opinion score to a document. Below we briefly explain some of the previous works following the lexicon-based approach and in case of similar models to our proposed model in Chapter 5, we highlight the difference of models with our proposed model.

Yang et al. [111] proposed a method that uses multiple modules each considering a different source of evidence in their opinion retrieval model. The first module identifies opinion based on common opinion terms. A lexicon is built by identifying terms that occur frequently in opinionated blog posts and infrequently in non opinionated blog posts. The resulting terms are manually filtered and assigned an opinion weight. A second module uses the standard set of Wilson subjective terms as a collection independent source of evidence. Since in blogs opinion is often expressed in a creative and non standard form, a third module looks at the low frequency terms as opinion evidence. Pronouns such as “I” and “you” appear very frequently in opinionated posts, so a fourth module recognizes certain n-grams which begin or end with these terms. The last module recognizes acronyms, specifically a subset of Netlingo’s chat acronyms and text message shorthands. The scores from each module are first normalized using MinMax method [50] and then combined using a weighted summation, where the weights are estimated from training data. At the end, the linear combination of relevance and opinion scores is used to score and rank documents.

Amati et al. [2] proposed an approach for automatically generating opinion word list from the TREC 2006 relevance judgment. Candidate opinion terms are selected according to a method that is similar to the Weighted Log-Likelihood Ratio [76, 75]. The list of opinion terms is then refined according to the term’s document frequency in the set of relevant opinionated documents. The assumption is that the best opinion bearing terms occur uniformly in the set of opinionated documents. Therefore, the number of opinionated documents that a term occurs in, is used as a threshold to select terms from the candidate set of opinion terms and obtain a smaller opinion lexicon. Based on the value of a threshold parameter, opinion lexicons with different sizes can be obtained. After building the opinion lexicon, opinion terms are submitted as a query to a search engine to get opinion scores for the relevant documents. Ranking is done in two steps. In the first step, documents are ranked according to opinion score divided by the topical relevance rank. In the second step, ranking is obtained by the topical relevance score divided by the ranking of the previous step.

He et al. [34] proposed a similar approach in which they build an opinion lexicon from the Blog06 collection by first removing very frequent and very rare terms from the vocabulary of the collection. Using a set of queries for training, they weight terms based on the divergence of their occurrence in the opinionated and relevant document compared to their occurrence in the relevant set of documents. The top weighted terms are considered as opinion bearing terms and are submitted as a query. The score assigned by the retrieval system for a blog post in response to this opinion query is considered as the opinion score of the post. Opinion and relevance score are normalized via dividing them by the maximum opinion and relevance score respectively. Two models are tried for combining the normalized opinion and relevance scores of documents. The first model is the linear combination of scores. The second model first transforms opinion score to probability by dividing the score by the sum of scores of all documents. It then uses a function of the log of the probability score and linearly combines it with the relevance score. Experimental results showed the superiority of the log-based model.

Lee et al. [52] use a generative model of words from subjective and objective resources: the Amazon review corpus as a subjective and Amazon specification corpus as an objective resource. The subjective weight of terms in SentiWordNet[3] [21] was considered as the prior weight of terms. The obtained opinion lexicon only contains words from the review corpus that are also present in SentiWordNet (i.e. the intersection of the two word sets). After building the lexicon, length-normalized sum of opinion terms are used to score documents for the opinion aspect. Relevance and opinion scores are then linearly combined to produce the final ranking.

Seki and Uehara [45] propose a language modeling approach based on the notion of subjective triggers. They differentiate between two classes of words in a subjective document: 1) the subject of the opinion or the object that the opinion is about, and 2) a subjective expression. They regard the former as a triggering word and the latter as a triggered word and automatically identify trigger patterns characteristic of subjective opinions using customer reviews collected from Amazon.com. They interpolate the resulting trigger model with the n-gram language model baseline. As potential triggering words, they experimentally chose pronouns such as: I, my, you, it, its, he, his, she, her, we, our, they, their, and this, and identify trigger pairs using some criterion. In building the model, they limit the history to the prior context (preceding words) in the same sentence. They use a subjective trigger language model to assign opinion score to documents. The opinion score of a document is then linearly combined with the relevance score of the document in the log space to produce the final score.

The above mentioned methods do not take the relatedness of opinion terms to a query into account when calculating an opinion score for a document. These methods assume that all opinion expressions in a document are targeted toward the topic of interest. An assumption that does not hold in a all document. Even a relevant document to a topic of a query can be relevant to multiple topics and some opinion expressions be targeted toward other topics than the topic of the query. Some previous works tried to address this limitation by using a topic-specific opinion lexicon and so ignore (or weight less) the opinion terms that are rarely used to express opinion about the topic of the query.

Na et al. [74] propose a method for creating a topic specific opinion lexicon via a feedback-style learning on the top retrieved documents in response to a query. The method starts by assigning opinion scores to the top retrieved doc-

[3]http://sentiwordnet.isti.cnr.it

uments for the query, using a general opinion lexicon such as SentiwordNet. It then updates the opinion weight of lexicon terms to be the average of the opinion score of the feedback documents in which the term has occurred. Thus, a term that occurred in documents with higher opinion score has its weight increased. This method showed slight improvement over using the general lexicon.

Huang and Croft [38] propose a single-stage opinion retrieval model where they use a small set of opinion terms as expansion to the query. They propose different ways of selecting a small number of (query dependent and query independent) opinion terms to expand a query. They use the expanded query to retrieve relevant opinionated documents. For query dependent opinion expansion, they proposed using a variation of the relevance model [49] to rank opinion terms from a general lexicon. They then used the top n opinion terms to expand the query. Their experiments showed that using an interpolated model in which both query independent and query dependent selected opinion terms are used, leads to the highest performance. However, using just the query dependent opinion terms did not have any advantage compare to the query independent variations.

Jijkoun et al. [41] proposed a bootstrapping method for obtaining a topic-specific lexicon from a general opinion lexicon. Jijkpun's method starts with a general opinion lexicon. It then finds the syntactic context of the opinion words in the top retrieved documents to a query, as well as in a background collection. Frequency of the syntactic contexts is then compared between the top relevant documents and the background collection using χ^2 statistics. The T different syntactic contexts with the highest χ^2 are then considered as the focused topic-specific lexicon. This method is useful for identifying relevant opinion expressions for a topic, thereby producing smaller opinion lexicons without significantly hurting the performance of the system.

Another approach in assigning topic-specific opinion score to documents is to consider the position of the opinion terms in a document and their distance with the topic of the query as an indicator of their relatedness. Our proposed model in Chapter 5 follows this approach. Below we briefly explain some previous works that follow this approach and highlight their difference with our model.

Santos et al. [97] use a divergence from randomness proximity model to integrate the proximity of query terms to the opinionated sentences identified by a general opinion finding system. They further combine the scores of opinionated sentences by the relevance score of the document using a linear combination. Our work is similar to this method in the sense that we also use a general opinion lexicon without refining it with query specific opinion terms, but our method differs in that we do not work on the sentence level but use the opinion weights and proximity of terms to the query directly. We also consider a proximity-based opinion density functions to capture the proximity information that has not been used in previous studies in opinion retrieval. The way we incorporate the relevance score in our model is also different from previous studies in that we investigate different ways of normalizing the relevance score.

M. Zhang et al. [117] proposed a formal generative model for opinion retrieval that considers the relevance score as a weight for the opinion score of a document. They consider the proximity of opinion terms to query terms by computing the probability of query term and opinion term co-occurrence within a window.

Vechtomova [106] builds an opinion lexicon using several manually built subjective resources. She weights each opinion term from the lexicon based on the Kullback-Liebler (KL) divergence between the set of relevant opinionated documents and relevant documents in Blog06 collection. She then uses a BM25 in which she calculate a pseudo-frequency value for a query term in a document. The pseudo-frequency of a query term is calculated based on the occurrence and proximity of subjective terms and their KL weight around them. The pseudo-frequency of query terms is used instead of its actual frequency in the BM25 model. Therefore, the occurrence of a query term is counted in the retrieval model weighted by its surrounding subjective terms and so documents are scored just based on their subjective and relevant content.

Although Zhang and Vechtomova considered proximity information in their models, Zhang et al. [117] did not find any advantage in using the proximity information while Vechtomova [106] did show some improvement in terms of opinion MAP. In this paper we introduce the proximity information in a more principled way and show that it can improve the performance over a non proximity-based opinion retrieval baseline.

Lee et al. [51] generate a *sentiment-relevance flow (SRF)* graph for each document, based on the relevance and opinion scores of every sentence, regarding their position in the document. To calculate the opinion score of every sentence, a fixed number of sentences before and after the sentence were taken into account. A regression model is trained using maximum entropy to predict the relevance of documents based on some features of the *SRF*. Interesting features such as the variance of sentence scores, the fraction of sentences with normalized scores more than 0.5, called the *peak*, and the first peak position are taken into account. Our model is similar to the SRF in a sense that we both use positional information in obtaining the opinion score. However the SRF model was mainly used to re-rank the top 15 documents and improve very early precision, while we use the proximity method to find the opinion score for all relevance-retrieved documents.

2.5 Score Transformation in Multi-criteria IR

In Chapter 6, we will consider the *topical relevance* and *opinionatedness about the query* as two criteria that should be satisfied in a document to be considered as relevant in an opinion retrieval system. In this section, we first review the previous works in the area of multi-criteria IR and look for common approaches for score combination of different criteria. We will show that linear combination of scores is the common approach. We then look into the previous works in multi-criteria IR and meta search and review the score normalization methods that has been proposed to normalize scores before combination.

Pereira et. al [13] considered relevance as a multidimensional property of document and proposed four relevance criteria namely, "aboutness", "coverage", "appropriateness" and "reliability" to improve *personalized IR*. They proposed an ordered aggregation method which linearly combines the score of documents for each criterion. A recent study [5] defined a set of document quality features related to different aspects of document quality such as "content readability", "provision of useful links" and "ease of navigation". A ranking function based on the linear combination of the quality features and topicality is used to promote high quality relevant web documents in the ranking. Blog post opinion retrieval can also be considered as a multi-dimentional relevance retrieval in which "topical relevance to the query" and "expressing opinion about the query" are the two facets that should be considered. Most of the proposed methods for solving this problem linearly combined the relevance score of blog with its opinion score. These studies mostly focused on proposing techniques to score documents for every aspect and did not considered the comparability of the scores to be combined.

Crawell et. al. [14] considered some query independent aspects of documents such as PageRank, URL length and link in-degree beside document content. They tried to find the optimum combination function of the query independent (static) feature values and document content in order to produce the final score of documents. They followed the Probability Ranking Principle, while representing a document by two components: the content score and the static feature score. Their model led to two additive scores: BM25 score and $log\frac{P(S|R)}{P(S|\bar{R})}$ which they call *indep* score since it is obtained after independence assumption between the content and static feature. They further adjusted the *indep* score by comparing the retrieved and relevant set of documents, in order to compensate the possible dependency of the results obtained by the two components. At the end, they guessed the functional form of the adjusted score called *floe*. Our work in Chapter 6 is similar to this study as we are also looking for the optimum transformation of features before combination. However, our model is more general since we are not limited to using BM25 as the relevance score and also we propose an automatic way of estimating the functional form of the score transformation functions, which is guaranteed to be optimum.

Multi-critera IR is closely related to meta-search, where the goal is to combine the result of multiple retrieval systems over a common document collection, to produce a more effective ranking. The difference between meta-search and multi-criteria IR can be explained as follows: in multi-criteria IR, independent criteria need to be satisfied at the same time but maybe with different weight, in every document. Therefore, one approach to this problem would be to score documents based on every criterion separately and then combine the scores of every document regarding different criteria to produce a final score which can be used for ranking. Similar to multi-criteria, in meta-search document scores produced by different retrieval systems need to be combined to produce a final ranking. However, usually the goal of every retrieval system is scoring documents for the same criterion (e.g. topical relevance), but different representation of query and documents or different retrieval functions are used in every system.

The incomparability of scores produced by different systems, was noticed in meta-search and other related tasks such as distributed IR. In previous studies, different methods have been proposed to normalize scores prior to combination [3]. We explain these methods in more details in the rest of this section.

Lee [50] tried to address the incomparability of scores by shifting and scaling them to the range [0,1]. We call this method *MinMax* in the rest of this thesis.

Montague and Aslam [69] proposed two normalization methods named *Sum* and *Z-score*. *Sum* normalization shifts minimum score to zero and scales sum to 1. *Z-score*, shifts mean to zero and scales variance to one. Experimental comparison between *Sum*, *Z-score* and *MinMax*, showed that *Sum* was the most effective and robust method between the three. The superiority of *Sum* in meta-search was confirmed in [99].

Later, normalization techniques were proposed that consider the shape of score distributions [61, 62, 99]. Manmatha et. al [61] showed that distribution of scores in relevant and non-relevant set of documents for a single query can be modeled by normal and exponential distributions respectively. They propose using EM algorithm to estimate the parameters of this mixture model in the absence of relevance judgement. The posterior probability of relevance given the score is then calculated based on the mixture components. It has been noticed that the posterior probability of relevance does not satisfy the monotonicity of transformation from original score, such that after a maximum probability, the higher original score leads to lower posterior probability. To resolve this issue, authors suggested considering a straight line form the maximum probability to the point (1,1). This way of fixing the problem ensures monotonicity but does not have a theoretical justification. The final score of document was calculated as the average posterior probability obtained from different engines.

In another work [62], Manmatha and Sever followed the normal-exponential mixture model of score distributions. They proposed equalizing non-relevant score distributions of different search engines in order to normalize their scores. The intuition was that non-relevant distribution of scores provides information on how a search engine maps a random set of documents to scores for a given query. In their view, score normalization should ensure that random documents are mapped in the same manner. For non-relevant distribution, this can be done by setting the minimum score to 0 and the mean of exponentials to be the same. In the absence of relevance judgement, non-relevant score distribution can be estimated using EM algorithm. It also can be approximated by the distribution of all scores. Therefore, normalization can be done by setting the minimum to 0 and equalizing the mean of all scores which is equivalent (in ranking) to the *Sum* normalization method discussed earlier. Authors mentioned that normalizing scores to the posterior probability of relevance, as was suggested in [61], did not performed as well as equalizing the non-relevant distribution. It was believed to be due to the error involving the estimation of the mixture model. In Chapter 6, we consider *MinMax*, *Z-score* and *Sum* methods as baselines.

Arampatzis and Robertson [3] provide a complete survey on modeling score distributions in IR. In this survey, the following problems has been reported for the normalization techniques that rely on score distributions. First, it has been discussed that the normal-exponential mixture model is not universal in modeling SDs in IR and some retrieval models such as KL-Dirverence may be better fitted with different mixtures. Second, the model does not satisfy the convexity condition (i. e. the normalized score does not increase monotonically with the original score). This problem mostly occurs at the top of the ranking, where it is very important for meta-search or multi-critera methods to have a precise relevance estimation. The proposed methods for resolving this issue were mostly heuristic and not theoretically justified. Also estimating the parameters of the mixture model was shown to be difficult, specially in the case of few relevant document.

Fernandez et. al. [22, 23] estimate a single cumulative density function (CDF) for every search engine based on historical queries. The normalized score is obtained as follows:

$$s' = F^{-1}(P(S \leq s)) \tag{2.3}$$

where $P(S \leq s)$ is the CDF of the probability distribution of all scores aggregated for a number of submitted query to the search engine. F is the CDF of an ideal scoring function which is common between all engines. As a simple approximation to the ideal system, F was estimated based on aggregated results of several good search engines for the historical queries and F^{-1} was obtained numerically based on F. Transformation using F^{-1} was called *Standardization* which aims at reducing any bias related to individual search engines. This method does not use training data and does not assume any parametric model for the distribution of relevant or non-relevant scores. It also does not introduce non-convexity since it results in monotonic transformation of original scores. This method is based on the assumption of comparability of scores across queries which may not be true specially for systems that use query features in the retrieval function. However, it does have the advantage that the transformation function can be calculated offline and may only have to change with significant collection change [3].

In order to accommodate combination of arbitrary features, machine learning techniques have been applied, leading to the recent work on learning to rank [53]. Many of these works rely on a linear combination of multiple features, but have not looked into the issue of compatibility; instead, the focus is on developing different loss functions and learning algorithms by assuming linear combinations. Normalization has been done on features, but mostly to bring the values to comparable ranges. Our work in Chapter 6 is complementary with this line of work in that we develop a general transformation strategy that can be applied to features in any learning to rank framework where linear combination of features is used.

2.6 Summary

In this chapter, we gave an introduction to the IR field. We briefly explained the retrieval models and machine learning techniques that we use throughout this thesis. We explained the opinion retrieval task which is the main focus of this thesis. Finally, we gave an overview of the previous works in the area of opinion retrieval and related tasks.

Chapter 3

Experimental Setup

The evaluation of information retrieval effectiveness is based on two factors: a test collection and an evaluation measure. In the standard evaluation setting we need a test set consisting of three things:

1. A document collection

2. A test suite of information needs, expressible as queries

3. A set of relevance assessments, standardly a binary value indicating relevant or non-relevant for each query-document pair.

Below we give a detailed description of the test sets and the evaluation measures that we used to report the performance of the models presented in this thesis.

3.1 Test Sets

We evaluated our proposed methods based on the test sets provided by TREC 2006-2008 blog tracks for the *Opinion Blog Post Retrieval* task. To the best of our knowledge the Blog06 collection is the only standard collection available for the opinion retrieval task with a reasonable collection size and number of topics and relevance judgements. An important advantage of evaluating our proposed methods on the standard TREC collection is that we will be able to compare the performance of our proposed methods with the state of the art approaches, tested on the same data set, without the need to implement and tune them on a different data set. Apart from facilitated comparison of the final results of different systems, we will be able to compare the effectiveness of our opinion scoring and final ranking methods with state-of-the-art systems by using the same standard relevance retrieval runs provided by the TREC Blog Track organizers. This way, the effect of the initial relevance retrieval step will be filtered and comparisons will be more focused on the two most challenging steps: 1) opinion scoring, 2) combination of the relevance and opinion scores to provide final ranking.

In the rest of this section we give a brief description of the test sets.

3.1.1 Document Collection

Our experiments are based on the Blog06 collection[1], created and used as a test collection in the Blog Track of TREC 2006-2008. The collection is crawled over an eleven weeks period from 6th Dec. 2006 until 21st Feb. 2006 and contains the XML feeds, the permalinks as well as the homepage of blog at the time of the crawl. The total size of the collection is 148GB, consisting of 38.6GB of feeds, 88.8GB of permalink documents and 20.8GB of homepages.

In order to provide a more realistic search environment, some spam, non-English and non-blog documents are included in the Blog06 collection [57, 79]. In fact, 13.1% of the permalinks in the collection are non-English. However, only English documents were evaluated as if they were relevant or not, while the non-English documents were assumed to be non-relevant. The collection also contains 509,137 spam blog posts.

In the experiments reported in this thesis, we only indexed the permalink components of blogs as retrieval units. Each permalink constitutes a post and its comments. The actual number of permalink documents in the Blog06 collection is 3,215,171. We did not perform any spam filtering or language detection. The preprocessing of the collection was minimal and involved only stopword removal.

3.1.2 Topic sets

There are three sets of topic available for the opinion retrieval task. Each topic set consists of 50 topics and is related to a specific year of having the opinion retrieval task from TREC 2006 through 2008. Topics were selected by the NIST assessors from a query log of a commercial blog search engine. Selected queries were expanded by the assessors and presented in the TREC topics format, where,

[1]`http://ir.dcs.gla.ac.uk/test_collections`

every topic contains three fields: title (the actual query), description and narrative. An example of a TREC topic is shown in Fig. 3.1.

```
<top>
<num> Number: 856 </num>

<title> macbook pro </title>

<desc> Description:
What has been the reaction to the Macbook Pro laptop computer?
</desc>

<narr> Narrative:
General statements of liking or disliking the Macbook Pro are
relevant.  Value comparisons to earlier versions of Macintosh laptops
or to other companies laptops are relevant.  Product reviews are
relevant if they contain opinions.  Speculation about unreleased
laptops is not relevant.
</narr>

</top>
```

Figure 3.1. Example of a TREC topic

In our experiments we considered the title part of every topic as the query. In order to be able to compare our results with TREC 2008 participants, we used the 100 topics from TREC 2006 and TREC 2007 (numbered 851 to 950) as our training set and the 50 topics from TREC 2008 (numbered 1001 to 1050) for testing.

3.1.3 Assessments

The standard approach to information retrieval system evaluation revolves around the notion of relevant and non-relevant documents. With respect to a user information need, a document in the test collection is given a binary classification as either relevant or non-relevant.

In the Blog06 collection, the relevance assessments for the opinion retrieval query sets provide information about whether a given blog post is relevant to a topic and also reflects the opinionatedness of the relevant posts. Table 3.1 summarizes the assessment scale used. Due to the large collection size, the assessments were done over the Blog06 collection with the pooling technique.

Scale	Meaning
-1	Not judged
0	Not relevant
1	Relevant to the target
2	Relevant and expresses negative opinion about the target
3	Relevant and expresses both positive and negative opinion about the target
4	Relevant and expresses positive opinion about the target

Table 3.1. Assessment scale in Blog06 collection for the opinion retrieval task

3.2 Evaluation

Measuring the performance of different systems is an important aspect of IR studies. In this section we first explain the common IR metrics for evaluating the performance of document retrieval, and in particular the opinion retrieval task. We then explain the significant testing that we perform to compare the results of different systems in this thesis.

3.2.1 Evaluation Metrics

Finding relevant documents is the basis of many IR tasks. Therefore, the two most used evaluation metrics are *precision* and *recall*. Precision is the fraction of retrieved documents that are relevant, and recall is the fraction of relevant documents that are retrieved [63]. However, precision and recall are computed without considering the rank of documents. Based on these two measures, different IR evaluation measures have been proposed that extend these two notions to evaluate ranked retrieval results.

Here, we explain a set of common IR metrics that has been widely used to report the performance of opinion retrieval systems. This helps us to compare the effectiveness of our models with the previous works in the area of opinion retrieval:

Precision at K documents (P@K) Precision at the point when K documents have been retrieved. This measure is particularly good for web search applications to measure how good are the results in the first page (p@10) or first three pages (P@30).

$$P@K = \#\{r \in R_i | rank(q_i, r) \leq K\}/K \tag{3.1}$$

where R_i shows the set of relevant documents to the query q_i. $rank(q_i, r)$ is the rank of document r for the query q_i and P is the precision at that rank.

Average Precision (AP) Precision is calculated at every point in the rank list where a relevant document have been retrieved (using zero as the precision for relevant documents that are not retrieved), and then averaged [11]. AP for query q_i can be written as:

$$AP_i = \frac{1}{|R_i|} \sum_{r \in R_i} P@rank(q_i, r) \tag{3.2}$$

R-Precision (R-Prec) Precision at the point when R relevant documents have been retrieved, where R is the number of relevant documents for a given topic. This measure is in fact the P@R.

$$P@R = \#\{r \in R_i | rank(q_i, r) \leq R\}/R \tag{3.3}$$

Binary Preference (bPref): The previous measures make no distinction in pooled collections between documents that are explicitly judged as non-relevant and documents that are assumed to be non-relevant because they are un-judged. The bPref measure considers the possible incompleteness of the relevance judgments and evaluates a ranking based on the number of judged non-relevant documents [10]. For query q_i with R_i as its set of relevant documents,

$$bPref = \frac{1}{|R_i|} \sum_{r \in R_i} 1 - \frac{\#\{n \in NR_i | rank(q_i, n) \leq rank(q_i, r)\}}{|R_i|} \tag{3.4}$$

where NR_i is the set of first $|R_i|$ judged non-relevant documents as retrieved by the system. R_i and $rank(q_i, r)$ have show the set of relevant documents to q_i and the rank of document r respectively.

Mean Average Precision (MAP): To report the performance of a retrieval system, the above measures are commonly averaged over a number of queries. For example, MAP measure is the mean of the AP over all topics in a given topic set Q:

$$MAP = \frac{1}{|Q|} \sum_{i=1}^{|Q|} AP_i \tag{3.5}$$

Among evaluation measures, MAP has been shown to have good discrimination and stability [9]. We use MAP as the main evaluation measure in this thesis. Other measures such as P@10, R-Prec and bPref are also used to report and compare the performance of different systems in this thesis.

3.2.2 Significance Test

In the experiment part of sections 4, 5 and 6 of this thesis, we compare the proposed models with some baseline systems. We may also compare two alternative variations of a proposed model. To check for "significant" difference between the two models, we use non-directional paired t-test since it has been shown to be more reliable than the Wilcoxon or signed test [96]. We report the statistical significance difference at level 0.01.

3.3 Unsupervised Normalization Methods

In the experimental parts of the research chapters of this thesis, we usually use normalization methods to make relevance and opinion scores of documents comparable before combining them. In Chapter 6, we propose a new supervised technique for normalizing the scores. We then use the unsupervised normalization methods as baseline to show the effectiveness of our proposed method. In this section, we explain the baseline unsupervised normalization methods that we use in this thesis as introduced in Chapter 2, in more details:

MinMax Normalization

MinMaxed normalization, shifts and scales scores of every query to the range [0,1] as follows:

$$MinMax(score \in S_q) = \frac{score - \min(S_q)}{\max(S_q) - \min(S_q)} \tag{3.6}$$

where, S_q is the set of scores that a retrieval system assigns to documents in response to the query q. $max(S_q)$ and $\min(S_q)$ indicate the maximum and minimum scores in set S_q.

SUM normalization

For every query, Sum normalization [69] shifts minimum score to zero and scales sum to 1:

$$Sum(score \in S_q) = \frac{score - \min(S_q)}{\sum_{score \in S_q}(score - \min(S_q))} \tag{3.7}$$

Sum normalization is used as a baseline in Chapter 6.

The score combination method in Chapter 5 is multiplication. Therefore, we do not report Sum normalization results since it is rank equivalent to the MinMax normalization when the scores are multiplied together.

Z-score normalization

For every query, the *Z-score* normalization[69] shifts the mean to zero and rescales the variance to one:

$$Z - score(score \in S) = \frac{score - mean(S)}{stdev(S)} \tag{3.8}$$

Z-score normalization is used in Chapter 5 to normalize relevance scores before combing with opinion scores. We also use z-score as a baseline normalization method in Chapter 6.

Logistic Regression

In Chapters 5 and 6, we use *Logistic Regression* for learning a transformation from relevance scores to probability estimates. We train the model using the relevance judgements from the training set. We used variations on the score or rank of the documents in returned by a retrieval model as a feature for logistic regression.

In Chapter 5, we use the following model to estimate the relevance probability of a document in each baseline:

$$P(d \text{ is relevant}|TREC\_baseline_j) = \frac{e^{\alpha+\beta.x}}{1+e^{\alpha+\beta.x}} \tag{3.9}$$

where x is one of the normalized scores, the rank or the log of the rank (or score) of document d. In order to estimate this probability, we learn values for α and β. We used the logistic regression implementation provided in LingPipe[2], the TREC 2006 topics, and the set of relevant and non-relevant documents for learning these parameters.

In Chapter 6, we again use the Logistic Regression to estimate the probability of relevance given topical score. We also use it to estimate the probability of

[2]http://alias-i.com/lingpipe/

relevance given the opinion score. We consider the transformation of scores obtained using Logistic Regression as another baseline to our proposed models.

HIS Normalization

HIS normalization is a variation of the method proposed in [22, 23]. It estimates a single cumulative density function (CDF) for every retrieval system based on historical queries. The normalized score is obtained as follows:

$$HIS(score \in S) = P(S \leq score) = \frac{|S \leq score|}{|S|} \tag{3.10}$$

where S is the set of all scores aggregated for a number of historical[3] queries submitted to a retrieval system.

In Chapter 6 we consider *HIS* method as a baseline normalization technique and use to normalize topical relevance and opinion scores before combination.

3.4 Summary

In this chapter, we introduced the Blog06 collection and the set of queries that we use for evaluating our proposed models in the context of opinion retrieval. We explained the common IR measures for evaluating the performance of opinion retrieval systems. We reported the significance test that we use for checking if our proposed models significantly improve the baselines. Finally, we introduced the normalization techniques that we use throughout this thesis as part of the proposed models or as baselines.

[3]We can consider training queries as historical queries

Chapter 4

Investigating Learning Approaches

4.1 Introduction

The opinion retrieval problem is usually tackled in 3 stages. In the first stage, a standard information retrieval system is used to rank posts by relevance, and the highest ranking posts are selected as a candidate set of documents. In the second stage, opinion scores are calculated for each candidate document, and in the third stage, the opinion and relevance scores are combined so as to produce a single ranking. A variety of different techniques have been employed in previous studies for realizing each of these stages and it is not clear from these studies to what extent learning (either in the opinion scoring stage or the ranking stage) or the feature selection method or the use of external datasets of opinionated terms/sentences/documents is responsible for the performance improvements demonstrated. Our intention in this chapter is to remedy this situation by focusing our investigation on the use of learning in the different stages of opinion retrieval and comparing learning techniques directly with baseline (non learning) techniques. We limit our study to use only data available in opinion retrieval relevance assessments in order to focus our investigation on what can be learnt automatically from the assessment data.

This chapter is structured as follows. We first define formally the problem and describe different learning approaches for solving it. Finally, we describe the empirical comparison we performed between different techniques for learning in opinion retrieval.

4.2 Problem Definition

The *blog post opinion retrieval* problem is the problem of developing an effective retrieval function that ranks blog posts according to the likelihood that they are *expressing an opinion about a particular topic*. To define this problem more formally, we introduce some notation. Assume that we have a set of labeled training documents, denoted D, and a set of training queries, denoted $q_1, ..., q_n$. For each query q_i we have a set of assessed documents $A_i \subset D$, a subset of which were considered relevant to the query $R_i \subset A_i$, and a subset of the relevant documents have also been marked as opinionated documents $O_i \subset R_i$. We now define the blog post opinion retrieval problem as the problem of learning a retrieval function $f : Q \times D \rightarrow \mathbb{R}$ such that the ranking of documents for each query $q_i \in Q$ is optimal (on average) with respect to a particular performance measure over rankings, $\mathscr{M} : \mathbb{B}^l \rightarrow \mathbb{R}$, (where $l \leq |D|$ is the maximum length of a ranking).

Note that we have deliberately defined the opinion retrieval problem to include only a collection of documents, a set of queries, corresponding relevance and opinion judgements, and an evaluation function $\langle D, [q_i, A_i, R_i, O_i]^n, \mathscr{M} \rangle$. Our aim is to understand how well a learning system can tackle opinion retrieval problem without needing to recourse to external sources of information such as opinionated term lists, product review corpora and so on.

4.3 Generating a Blog Post Opinion Score

Obviously if a blog post does not mention a particular topic then the author cannot be expressing an opinion about it. Thus the first step in most opinion retrieval systems is to rank posts by their relevance to the query. Once we have a set of relevant documents, the problem becomes that of estimating a score for the "opinionatedness" of each document. We investigate both learning and non-learning (baseline) approaches for doing this. In the first (non-learning) approach we calculate an opinion score for each term in the document and then combine the score over all terms in the document. In the second we train a classification system to distinguish between opinionated and non-opinionated posts using the opinionatedness of each term for feature selection. We then use the confidence of the classifier as an opinion score for the document.

4.3.1 Discovering Opinionated Terms

Before defining different opinion scores for terms we introduce some more notation. Let $O = \cup_i O_i$ be the set of all opinionated documents in our training set and $R = \cup_i R_i$ be the set of all relevant documents, (note that $O \subset R$). The relative frequency of a particular term t in the set O is denoted $p(t|O)$ and calculated as:

$$p(t|O) = \frac{\sum_{d \in O} c(t,d)}{\sum_{d \in O} |d|} \tag{4.1}$$

where $c(t,d)$ denotes the number of occurrences term t in document d and $|d|$ denotes the total number of words in the document. The relative frequency of terms in the relevant set $p(t|R)$ is defined analogously.

One simple score for comparing the likely "opinionatedness" of terms is the ratio of relative frequencies in the opinionated and relevant sets, which we call the Likelihood Ratio:

$$opinion_{LR}(t) = \frac{p(t|O)}{p(t|R)} \tag{4.2}$$

A second formula is the technique proposed by Amati et al. [2], which is a slight variation on the standard Weighted Log-Likelihood Ratio [76, 75] feature selection technique, (where the relevant set R replaces the "non-opinionated" set $\bar{O} = R$).

$$opinion_{WLLR}(t) = p(t|O) \log \frac{p(t|O)}{p(t|R)} \tag{4.3}$$

Note that the summation over all terms of the opinion score gives the well-known Kullback-Leibler divergence [64] between the opinionated document set and the relevant document set. This measure quantifies the dissimilarity between the two sets of documents. Terms that cause high divergence are therefore good indicators of opinionatedness.

We investigate also another formula for estimating the opinionatedness of a term, based on the concept of Mutual Information (MI) [64]. Mutual Information is often used for feature selection in machine learning [98].

$$opinion_{MI}(t) = \sum_{x \in \{t \in d, t \notin d\}} \sum_{y \in \{d \in O, d \notin O\}} p(x,y) \log \frac{p(x,y)}{p(x), p(y)} \tag{4.4}$$

Here the joint and marginal probabilities are calculated using the document frequency in the sets O, $\bar{O}$ and R as follows:

$$p(t \in d, d \in O) = df(t,O)/|R|$$
$$p(t \in d, d \notin O) = df(t,\bar{O})/|R|$$
$$p(t \in d) = df(t,R)/|R|$$
$$p(d \in O) = |O|/|R|$$

where $df(t,O) = \#\{d \in O | t \in d\}$ is the number of relevant and opinionated documents that contain the term t. Similarly, $df(t,\bar{O}) = \#\{d \in \bar{O} | t \in d\}$ is the number of relevant but not opinionated documents containing the term t and $df(t,R) = \#\{d \in R | t \in d\}$ is the number of relevant documents that contain the term t.

A related feature selection measure, also based on document frequencies, is the χ^2 score:

$$opinion_{\chi^2}(t) = |D| \frac{[p(t \in d, d \in O)p(t \notin d, d \notin O) - p(t \in d, d \notin O)p(t \notin d, d \in O)]^2}{p(t \in d)p(t \notin d)p(d \in O)p(d \notin O)} \tag{4.5}$$

4.3.2 Document Opinion Scores

In order to calculate an opinion score for an entire document, we can simply calculate the average opinion score over all the words in the document:

$$opinion_{avg}(d) = \sum_{t \in d} opinion(t)p(t|d) \tag{4.6}$$

where $p(t|d) = c(t,d)/|d|$ is the relative frequency of term t in document d.

Alternatively, as stated previously, we can train a classifier and in particular a Support Vector Machine (SVM) to recognize opinionated documents. Training data in this case will be the set of all relevant documents R, with the set of opinionated documents O being positive examples of the class. We can then use the confidence of the classifier (i.e. the distance from the hyperplane) as the opinion score for each document[1]. The per term opinion score is used in this case only for feature selection, with the m highest scoring terms being selected as features for the classifier. We use the relative frequency of each of these terms in the document as the feature weight:

$$opinion_{SVM}(d) = f_{SVM}(p(t_1|d), ..., p(t_m|d)) \tag{4.7}$$

where the function $f_{SVM}()$ is the output (confidence) of the trained SVM for a particular document and m is the size of the feature set (vocabulary).

[1]Note that the classifier is *not* being used to perform regression estimation as the training examples have categorical class labels $c \in \{0,1\}$ as opposed to a range of value, e.g. $c \in [0,1]$.

4.4 Combining Relevance and Opinion Scores

In order to combine the relevance score with the opinion score we again investigate both learning and non-learning approaches.

4.4.1 Simple Combinations

In the baseline approach we investigate simple algorithms for combining the two scores. Our first approach is simply to rerank the top N results according to the opinion score:

$$basicScore(q_i, d) = opinion(d) \tag{4.8}$$

The next approach is to simply weight the opinion score by the relevance score for the document:

$$product(q_i, d) = relevance(q_i, d) * opinion(d) \tag{4.9}$$

where $relevance(q_i, d)$ is the relevance score that was given to document d for query q_i. This score could be a simple content-based retrieval function like BM25 [101, 87] and Language Modeling [116], or it could be combined using a Learning to Rank approach [53] with additional information including the content of incoming hyperlinks and tag data from social bookmarking websites.

A third approach is to take the rankings produced by the opinion score and the relevance score and merge them. The simplest way to do that is to use Borda Fuse [70], which is simply to calculate a score based on the sum of the individual ranks.

$$borda(q_i, d) = rank_{opin}(q_i, d) + rank_{rel}(q_i, d) \tag{4.10}$$

where $rank_{opin}(q_i, d)$ and $rank_{rel}(q_i, d)$ is the rank of document d in the opinion and relevance rankings for query q_i respectively. Whenever there are ties in the ranking (i.e. two or more documents have the same opinion or relevance score), the rank is the mean of the tied rankings (e.g. two documents tied at position 10 would both be given the rank 10.5).

4.4.2 Using a Rank Learner

Our final approach to ranking blog posts by their opinionatedness relies on a learning framework. In this case we train a Learning to Rank system named SVMmap [114] to take both the relevance score and the opinion score for each document into account when producing an output ranking. Note that because of having opinion and relevance scores as the two features, SVMmap is simply helping us to find the coefficients in the linear combination of the two scores. Training data in this case is the set of assessed documents A_i for each query q_i, where positive examples are those documents judged to be both relevant and opinionated, O_i.

4.5 Experimental Results

In this section we discuss the experiments we conducted in order to determine the usefulness of different learning frameworks for performing blog post opinion retrieval.

In order to avoid overfitting the data we perform 10 fold cross-validation on the 150 queries. Thus for each fold we had 135 queries in the training set and 15 queries in the test set. The training and test data was kept separate *in all stages* of the experiment. Training data was used in the feature selection phase in order to determine highly opinionated terms, in the classification phase to learn a model for detecting opinionated documents, and in the ranking stage to train the rank learner to best combine opinion and relevance scores for each document.

4.5.1 Discovering Opinionated Terms

In this section we investigate the different techniques discussed in section 4.3.1 for discovering terms which are good indicators of opinionated content. Table 4.1 shows a list of the most opinionated terms in the collection according to the different weighting schemes outlined in section 4.3.1. It would appear from the list that the Likelihood Ratio (LR) is not a useful weighting scheme. On the other hand, the Weighted Log Likelihood (WLLR) does a reasonable job of discovering terms that are likely to indicate opinionated content, such as “think” and “like”, but the document frequency based Mutual Information (MI) appears to do even better. Not surprisingly, the χ^2 measure seems to rank terms almost identically to MI.

4.5.2 Detecting Opinionated Documents

Here we investigate the use of term scores for feature selection when we want to use the learning methods for calculating the opinion score for documents.

Rank	LR	WLLR	MI	χ^2
1	jeffgannon	8217	just	just
2	ruberri	who	think	think
3	opotho	peopl	know	know
4	mtlouie	muslim	like	like
5	mcclatchi	like	don	don
6	spaeth	think	ve	who
7	alschul	just	who	ve
8	tomwright	don	love	sai
9	hairlessmonkeydk	2006	sai	love
10	snoopteddi	sai	littl	littl
11	amlp	februari	well	well
12	mccammon	can	am	am
13	quicklink	post	actual	peopl
14	rrlane	comment	didn	actual
15	deggan	12	peopl	didn
16	wpf	right	ll	ll
17	fatimah	know	thought	thought
18	pixnap	pm	let	let
19	onim	cartoon	wai	see
20	martouf	decemb	see	wai
21	perfess	islam	still	still
22	pemolin	see	feel	feel
23	bka	rsquo	reason	reason
24	suec	film	hate	hate
25	thecitizen	time	god	go
26	tvtattl	want	go	want
27	scocca	will	sure	sure
28	rutten	onli	yes	god
29	fws	wai	believ	yes
30	pav	am	want	try

Table 4.1. The highest scoring opinion bearing terms according to different opinion scores. The scores were calculated over the training data for a single fold, consisting of the opinionated and relevant sets of documents for 135 queries.

We only compare the WLLR and ML term weighting methods, because the LR does not seem a good weighting method and χ^2 ranks term pretty much the same as MI. We rank terms according to these weighting methods and choose the top 7000 terms as features. In order to train a classifier we use the SVM-light tool [43] with the linear kernel and other default parameters. We use the TREC qrels to evaluate the classifier in terms of *precision, recall* and combined *F1* score, which are defined as follows, where C is the set of documents classified as opinionated:

$$Precision = |C \cap O|/|C|$$
$$Recall = |C \cap O|/|O|$$
$$F1 = 2|C \cap O|/(|C| + |O|)$$

The results for this experiment are shown in Table 4.2.

	Weighted Log Likelihood (WLLR)	Mutual Information (MI)
Precision	66.5%	63.8%
Recall	81.0%	95.2%
F1	72.6%	75.7%

Table 4.2. Comparing the effect of different feature selection methods in the classification of opinionated blog posts

The precision values are relatively low for both feature selection methods, because the ratio of negative to positive examples is more than five to one. Hence the false positive rate is high. MI-based feature selection achieved a very high recall rate of 95% and despite slightly lower precision than WLLR, achieved a better $F1$.

4.5.3 Opinionated Document Retrieval

For coming up with the final ranking of documents, we inspected simple combinations of opinion and relevance score and also a particular rank learner named SVM-map [114] which is designed to optimize rankings for the Mean Average Precision (MAP) performance measure.

The results for the experiment are given in Table 4.3. We report both the opinion retrieval MAP scores as well as the percentage change with respect to the opinion retrieval score for the baseline (relevance retrieval) system. The different ranking methods are listed vertically, while the techniques for generating a document opinion score are divided horizontally.

	Expected Value		Learning (SVM)	
	WLLR	MI	WLLR	MI
opinion	0.281 (-26.11%)	0.291 (-23.50%)	0.284 (-25.00%)	0.285 (-25.00%)
opinion*relevance	0.30(-20.99%)	0. 349(-8.44%)	0.312(-18.01%)	0.333 (-12.49%)
Borda Fuse	0.381(-0.0%)	0.381(0.0%)	0.342(-10.13%)	0.362(-4.83%)
Rank Learning	0.390(2.39%)	0.389(2.07%)	0.386(1.34%) $*$	0.384(0.88%) $*$

Table 4.3. opinion finding MAP on the Blog06 dataset using different methods of opinion scoring for documents and different methods of combining opinion and relevance scores. Percentages show improvements over the best baseline opinion retrieval ranking. Symbol $*$ indicates statistically significant improvement over the baseline.

One can see that the final performance of the opinion retrieval system is very sensitive to the method used for combining the relevance and opinion scores. Results show that only the Rank Learning using SVMmap is useful for combining the opinion and relevance scores. In fact, all other combination methods produce ranking which is less effective than the simple ranking based on the relevance score.

Comparing the lexicon-based and learning-based opinion scoring methods shows that there is no statistically significant difference between them. However, unlike lexicon-based opinion scoring methods, learning-based methods lead to rankings which are statistically significantly better than the relevance baseline.

The comparison between the two feature selection (weighting methods), WLLR and MI, did not provide a conclusive result. According to the paired T-test their difference is not significant.

4.6 Conclusions

In this chapter we limited ourselves to the information that can be obtained from collection's relevance judgment and investigated different models of weighting terms based on the statistics of their occurrence in relevant and opinionated versus non-relevant documents. Based on terms' weights, we selected terms as features for learning an opinion classifier. We also used weighted terms to create an opinion lexicon. We assumed that the whole content of documents that are retrieved in the relevance retrieval step is relevant to the query. Therefore, we used every opinion expression that occur in any position of a document in the opinion scoring. We then tried simple learning and lexicon-based methods for assigning opinion scores to documents. Finally we tried different methods for combining relevance and opinion scores to produce a final ranking. We evaluated different models experimentally and compared the performance with the topical relevance retrieval baseline.

Results showed that the best term weighting methods are mutual information and weighted log likelihood ratio. We did not notice any statistical significant difference between the two methods. We also did not notice a statistical significantly difference between the learning-based and lexicon-based methods of opinion scoring. However, unlike the lexicon-based approach, the learning-based methods showed statistical significantly improvement over the topical relevance retrieval baseline. Finally, comparison between different methods of combining the relevance and opinion scores showed that SVMmap, as a learning to rank method is the only combination method that leads to improvement over the topical relevance baseline. We have shown that learning can be effective both for generating opinion scores for individual documents and also for learning a retrieval function that combined opinion evidence with relevance into a single retrieval function. A distinct advantage of the approach proposed in this chapter is that by performing multiple levels of learning, we maximize the use of available training data while not relying on external sources. However, the performance improvement compared to the topical relevance baseline was not very high. The proposed models also did not performed well compared to the state of the art opinion retrieval methods which use informations such as externally built opinion lexicon or those that consider the relatedness of opinion terms to the query in their opinion scoring method. We believe the main reason behind this poor performance is the simple assumption that all opinion expressions in a topically relevant document are targeted at the query. In the next chapter we will use external opinion lexicon that has been shown to be effective in other studies and consider the relatedness of opinion terms to the query in the model by taking their proximity into account. The analysis in this chapter also showed the sensitivity of the performance of the system to the methods that we use for combining the relevance and opinion scores. the best performing setting was using the learning to rank method for combination which is based on the linear combination of the features. In the next chapters, specially in Chapter 6, we will focus on the combination step more deeply.

Chapter 5

Exploiting Proximity Information

5.1 Introduction

In previous chapter we limited ourselves to the information that can be obtained from collection's relevance judgment. We also assumed that the whole content of documents that are retrieved in the relevance retrieval step is relevant to the query. Therefore, we used every opinion expression that occur in any position of a document in the opinion scoring. However, the performance improvement was not very high compared to other opinion retrieval methods which use informations such as externally built opinion lexicon or those that consider the relatedness of opinion terms to the query in their opinion scoring method. In previous chapter we also showed the sensitivity of the performance of the system to the methods that we use for combining the relevance and opinion scores.

In this chapter we use an externally built opinion lexicon that has been shown to be effective in previously studies. The lexicon is used to identify the opinionated content of a document. We propose a novel probabilistic model which uses proximity-based density functions to model the notion of query-relatedness for opinionated content. Our model ranks documents based on the product of relevance and opinion probabilities. In previous chapter, multiplication was shown to be the least effective technique. However, here we show the importance of score normalization before multiplying the two scores and show that using proper normalization very high performance can be achieved.

The remainder of the chapter is structured as follows. Section 5.2 describes the proposed probabilistic relevant opinion retrieval system. Estimating the opinion density at a document position is explained in Section 5.3. Different methods for aggregating the opinion density information to estimate the opinion score of documents are explained in Section 5.4. Different opinion lexicons are investigated in Section 5.5 and their effectiveness is compared. Section 5.6 presents the results of experiments and investigates the effectiveness of different setting of the proposed methods. We analyze the results on a per topic and per document basis in Section 5.7. Finally conclusions are given in the last section. Sections 5.2, 5.3 and 5.4 constitute the core contribution of this chapter, while the rest of this chapter provides thorough analysis of the results and compare the effectiveness of different proposed techniques with other proposed models and the state of the art approaches.

5.2 Problem Definition

Blog post opinion retrieval aims at developing an effective retrieval function that ranks blog posts according to the likelihood that they are *expressing an opinion about a particular topic*. We follow the typical generative model of IR that estimates the likelihood of generating a document given a query, $P(D|Q)$. In opinion retrieval we also need to estimate the probability of generating an opinion about the query. We introduce the random variable O which denotes the event that the document expresses an opinion about the query. Thus, for opinion retrieval we can rank documents by their likelihood given the query and opinion, $P(D|Q,O)$:

$$P(D|O,Q) \propto P(D,O,Q) = P(D)P(Q|D)P(O|Q,D) \tag{5.1}$$

We can see two components in this ranking formula: $P(D)P(Q|D)$ which considers the relevance of document to the query, and $P(O|D,Q)$ which scores a document based on the amount of opinion expressed in the document about the query. The relevance probability can be estimated using any existing IR method such as language models [85] or classical probabilistic models [25].

In this chapter, we focus on the second component of the model which addresses the opinion scoring of documents. In the rest of this section, we explain the possible options for estimating this probability.

5.2.1 Query-independent Opinion Score

The first studies on opinion retrieval assumed conditional independence between O and Q given the document D. So, $P(O|D,Q)$ in those models was calculated as $P(O|D)$. Such models assume that each document discusses only one topic and so if a document is relevant to a query, all opinion expressions in the document are about the query. Therefore, to estimate a query-independent (general) opinion score for a document, we can simply calculate the average opinion score over all terms in the document:

$$P(O|D,Q) = P(O|D) = \sum_{t \in D} P(O|t)P(t|D) \tag{5.2}$$

where $P(t|D) = c(t,D)/|D|$ is the relative frequency of term t in document D and $P(O|t)$ shows the probability of opinionatedness of the term.

5.2.2 Query-dependent Opinion Score

The assumption that a document is only relevant to a single topic and that all opinion expressions are about that topic is overly simplistic. In fact, a document can be relevant to multiple topics and just be opinionated about one of them. Therefore, for assigning opinion scores to documents, we need to identify opinion expressions that are directed toward the query topic. One possible approach is to find opinion lexicons that are mostly used to express opinion about the query topic. For instance, the word “delicious” may be used more for expressing opinion about a food type query than an electronic product. Having access to query-related opinion lexicons, we can either ignore the word delicious or give it a low weight if the query is about electronics. For example Na et al. [74] used an opinion lexicon refinement via pseudo relevance feedback in order to build a query-related opinion lexicon.

Another approach is to use the documents’ structure. In this approach the distance of an opinion term to the query term is used as a measure of their relatedness. Accordingly, we assume that an opinion term refers with higher probability to the terms closer to its position. On the other hand, opinion terms can refer not only to the entities they proceed or follow, but also to the entities which may be a couple of words, or even sentences, before or after. Bi-gram or tri-gram models have limitations in capturing such dependencies between opinion and topic terms. In order to model this dependency, we propose considering proximity-based density kernels.

5.3 Estimating the Opinion Density at a Specific Position of a Document

To model the dependency between opinion terms and query terms, we propose considering proximity-based density kernels, centered at each opinion term, which favor positions closer to the opinion term’s position. As a kernel we can

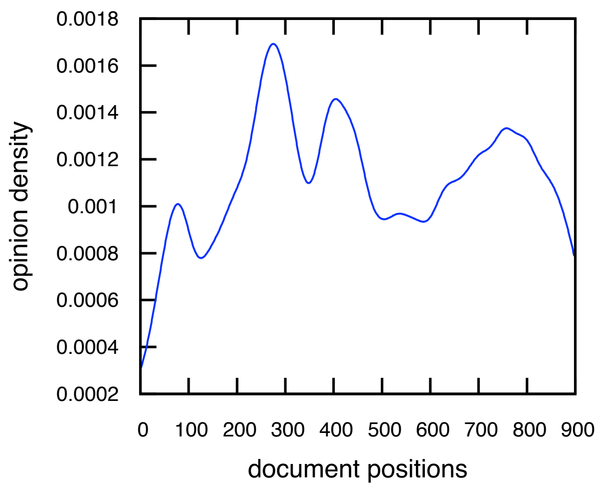

Figure 5.1. Example of opinion density at different word positions of a document. The first word is at position 1, second word is at position 2 and so on.

use any non-increasing function of the distance between the position of an opinion term and any other position in a document [55]. We weight this kernel by the probability of opinionatedness of the term. Therefore, the opinion density at each position in the document is the accumulated opinion density from different opinion terms at that position. We define this accumulated probability to be the probability of opinion expressed in the document about the term at that position. Figure 5.1 shows the opinion density at different positions in a sample document.

In order to present our model more formally, we first introduce some notation. We denote a document with the vector $D = (t_1, ..., t_i, ..., t_j, ..., t_{|D|})$ where the subscripts i and j indicate positions in the document and t_i indicates the term occurring at the position i. To find the opinion probability at i, we calculate the accumulated opinion probability from all positions of the document at that position. So, for every position j in a document we consider the opinion weight of the term at that position which we denote by $P(O|t_j)$, and we weight it by the probability that a term at that position j is about a term at position i. We represent this probability by $P(j|i,D)$ and calculate it as follows:

$$P(j|i,D) = \frac{k(j,i)}{\sum_{j'=1}^{|D|} k(j',i)} \tag{5.3}$$

here $k(i,j)$, is the kernel function which determines the weight of propagated

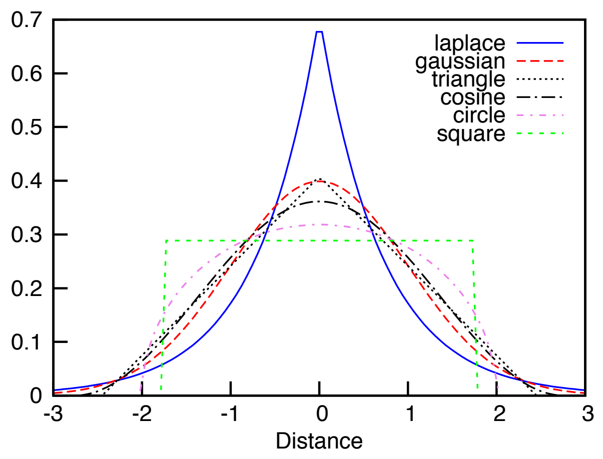

Figure 5.2. Proximity kernel functions with the same variance.

opinion from a term at position j to position i. Thus the probability of opinion at position i in the document can be estimated as:

$$P(O|i,D) = \sum_{j=1}^{|D|} P(O|t_j)P(j|i,D) \tag{5.4}$$

In the rest of this section we present the different kernels used in our experiments. We investigate the five different density functions used in [55], namely the Gaussian, Triangular, Cosine, Circle and Rectangular kernel. We also present the Laplace kernel as an additional kernel in our experiments. Figure 5.2 shows the different kernels all with the same variance.

In the following formulas, we present normalized kernel functions with their corresponding variances (denoted σ^2).

1. Gaussian Kernel

$$k(i,j) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[\frac{-(i-j)^2}{2\sigma^2}\right] \tag{5.5}$$

2. Laplace Kernel

$$k(i,j) = \frac{1}{2b} \exp\left[\frac{-\left|i-j\right|}{b}\right] \text{ where } \sigma^2 = 2b^2 \tag{5.6}$$

3. Triangular Kernel

$$k(i,j)=\begin{cases}\frac{1}{a}\left(1-\frac{|i-j|}{a}\right) & \text{if}\,|i-j|\leq a\\ 0 & \text{otherwise}\end{cases}\qquad \text{where } \sigma^2=\frac{a^2}{6} \tag{5.7}$$

4. Cosine Kernel

$$k(i,j)=\begin{cases}\frac{1}{2s}\left[1+cos\left(\frac{|i-j|.\pi}{s}\right)\right] & \text{if}\,|i-j|\leq s\\ 0 & \text{otherwise}\end{cases}\qquad \text{where }\ \sigma^2=s^2\left(\frac{1}{3}-\frac{2}{\pi^2}\right) \tag{5.8}$$

5. Circle Kernel

$$k(i,j)=\begin{cases}\frac{2}{\pi r^2}\sqrt{r^2-(i-j)^2} & \text{if}\,|i-j|\leq r\\ 0 & \text{otherwise}\end{cases}\qquad \text{where }\ \sigma^2=\frac{r^2}{4} \tag{5.9}$$

6. Rectangular Kernel

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

As one baseline, we compare proximity kernels to the uniform kernel which gives the same importance to all positions in the document and simulates the non proximity-based opinion retrieval presented in section 3.1. Our aim is to investigate whether it is better to use kernels which favor opinion occurrence in close proximity of query term or not.

5.4 Estimating Documents' Relevant Opinion Score

Now that we can compute the probability of opinion about a term at each position of the document, we need to calculate an overall probability that the document is expressing an opinion about the query, $P(O|D,Q)$.We estimate this probability with respect to the position information of the document as follows:

$$\begin{aligned} P(O|D,Q) &= \sum_{i=1}^{|D|} P(i,O|D,Q) \\ &= \sum_{i=1}^{|D|} P(i|D,Q)P(O|i,D,Q) \end{aligned} \tag{5.11}$$

We assume that O and Q are conditionally independent given the position in the document, $O \perp Q|(i,D)$. Thus $P(O|i,D,Q)$ reduces to $P(O|i,D)$ which can be estimated using methods proposed in the section 5.3. The conditional independence between O and Q comes from the assumption we made in this thesis that information about position of query and opinion terms and therefore their proximity is enough to estimate the probability that the opinion is targeted at the query. Note that a more complex system can take advantage of extra information such as query-specific opinion lexicons. A query-specific opinion lexicon assigns different weight to an opinion term depending on the query, that is $P(O|t,Q)$. For example the term "delicious" has more opinion weight if the query is "pasta" compared to the case when the query is "macbook pro" (i.e. $P(O|delicious, pasta) > P(O|delicious, macbook\ pro)$). In such system, we do not consider the conditional independence of O and Q. We can estimate $P(O|i,D,Q)$ using the method proposed in Section 5.3, by substituting $P(O|i,D)$ with $P(O|i,D,Q)$ and substituting $P(O|t_j)$ with $P(O|t_j,Q)$ in Equation (5.4). This way, we take advantage of both proximity information and topic-specific lexicon.

Plugging Equations (5.11) and (5.4) into Equation (5.1), we have the following model for ranking documents:

$$P(D|O,Q) \propto P(D)P(Q|D)\sum_{i=1}^{|D|} P(i|D,Q)P(O|i,D) \tag{5.12}$$

$$\propto P(D)P(Q|D)\sum_{i=1}^{|D|} P(i|D,Q)\sum_{j=1}^{|D|} P(O|t_j)P(j|i,D) \tag{5.13}$$

In the remainder of this section we suggest different methods for estimating $P(i|D,Q)$, the probability of position i given the query Q and the document D.

5.4.1 Average

One strategy is to estimate $P(i|D,Q)$ based on the occurrence of the query terms as follows:

$$P(i|D,Q) = \begin{cases} \frac{1}{|pos(Q,D)|} & \text{if } t_i \in Q \\ 0 & \text{otherwise} \end{cases} \tag{5.14}$$

where $pos(Q,D)$ is the set of all query term positions in document D, (where $t_j \in Q$).

Equation (5.14) assigns equal weight to all query term positions. Therefore, it does not differentiate different query terms and even an occurrence of a single term from a *multi-term*[1] query is considered an indicator of the query and so a relevant position.

Using Equation (5.14), we have the following formula for estimating $P(O|D,Q)$:

$$P(O|D,Q) = \sum_{i=1}^{|D|} P(i|D,Q)P(O|i,D) = \frac{1}{|pos(Q,D)|} \sum\nolimits_{i \in pos(Q,D)} P(O|i,D) \tag{5.15}$$

Intuitively, Equation (5.15) states that, an opinion score of a document should be based on the average of the opinion expressed toward any query term occurrences. However, this assumption may be too restrictive. Consider a document which first contain some objective sentences describing the topic and then discusses its opinion about it. Although the document contains some very subjective sentences about the query, the *Average* opinion scoring method, may assign low score to this document due to the objective sentences about the query. This can be more problematic when the objective sentences contain only a subset of the query terms, while the document contains some very subjective sentences which contain *all* of the query terms (or vice versa).

5.4.2 Relevance Weighted Average (RWA)

One possible solution to the problem mentioned for the *Average* variation is to weight the query term occurrences according to their "relevance" to the query Q using the positional language model [55]. By doing this, we give more importance to those query term positions that are close to other query term occurrences and so are more likely to be relevant to the query topic. Therefore, we estimate $P(i|D,Q)$ as follows:

[1]A multi-term query is a query that contains more than one term.

$$P(i|D,Q) = \begin{cases} \frac{P(Q|D,i)}{\sum_{i \in pos(Q,D)} P(Q|D,i)} & \text{if } t_i \in Q \\ 0 & \text{otherwise} \end{cases} \tag{5.16}$$

$$\begin{aligned} P(O|D,Q) &= \sum_{i=1}^{|D|} P(i|D,Q) P(O|D,i) && (5.17) \\ &= \sum_{i \in pos(Q,D)} \frac{P(Q|D,i)}{\sum_{i \in pos(Q,D)} P(Q|D,i)} P(O|D,i) && (5.18) \end{aligned}$$

In this method, we take the weighted average of opinion density at query term positions such that the more relevant query position is given more weight. This method can be useful in cases where a subset of query terms may be used to refer to a different topic/entity from that of the "whole query". For example, consider the query *"March of the Penguins"*. Intuitively, we would like to count more the opinion density at positions close to *all of the query terms* (i.e. positions that are likely to be referring to the movie as a whole) as opposed to positions close to the occurrence of a single query term like *Penguin*, (which might be a reference to a character from the movie). This method has the danger, however, of underestimating the opinion expressed in a document in cases where a subset of query terms can indeed be used as a reference to the whole query. For example, in case of queries referring to person entities, such as *"Steve Jobs*", the first name of the person, *Steve*, may be used in the later parts of a document to refer to the entity after the first mention of the entire name. In this case, any occurrence of *Steve* is in fact a reference to the whole query, but the model can not recognize this and thus weights less the opinion expressed at these later positions compared to the first occurrence which contains the whole query.

This model is able to differentiate different query positions inside a document with respect to their relevance to the query. However, it can not differentiate between a document which contains all query terms compared to the one that just has one of the query terms. This is assumed to be handled in the relevance scoring component of the model.

Also, due to the averaging nature, this method can have the same problem of the first method in assigning lower opinion scores to documents that contain also objective sentences about the query.

5.4.3 Maximum

Since the Average-based methods proposed previously may be too restrictive, we propose another variation called *Max*. This model is based on the intuition that a relevant document is considered subjective toward a query if it has at least one subjective passage about the query. Therefore, we find the opinion density at every query term positions and then assign the maximum opinion density as the opinion score of the document. Formally, we assume that only the query term position where $P(O|D,i)$ is maximum is important. Thus:

$$P(i|D,Q) = \begin{cases} 1 & \text{if } i = \arg\max_{i' \in pos(Q,D)} P(O|D,i') \\ 0 & \text{otherwise} \end{cases} \tag{5.19}$$

Thus, the opinion density at the maximum subjective query position, is considered as the opinion score of the document:

$$P(O|D,Q) = \sum_{i=1}^{|D|} P(i|D,Q)P(O|D,i) = max_{i \in pos(Q,D)} P(O|D,i) \tag{5.20}$$

Like other previously mentioned variations, the *Max* variation can not differentiate between a document that has all query terms compared to a document with just some of the query terms.

5.4.4 Ordered Weighted Averaging (OWA)

The drawback of the *Max* method is that it relies only on the highest opinion evidence of every document and misses other available evidence that might be useful. As a more general aggregation method which takes into account more opinion evidence, we use the ordered weighted averaging (OWA) operator, which was introduced by Yager [110]. OWA provides a parametrized class of mean type aggregation operators, that can generate the *OR* operator (Max), the AND operator (Min) and any other aggregation operator between them like the $Mean$. An OWA operator of dimension K is a mapping $F : \mathbb{R}^K \to \mathbb{R}$ with associated weight vector W,

$$W = [w_1, w_2, ..., w_K]$$

such that

$$\sum_{i=1}^{K} w_i = 1, \;\; 0 \le w_i \le 1,$$

and where

$$F(a_1,...,a_K) = \sum_{i=1}^{K} w_i a_{ind(i)} \tag{5.21}$$

here $a_{ind(i)}$ is the ith largest element for whatever measure we are using in the collection $a_1,...,a_K$. OWA operators have different behaviours based on the weighting vector associated with them. Yager introduced two measure for characterizing OWA operators [110]. The first one is called *orness* which characterizes the degree to which the operator behaves like an *OR* operator (Max). The second measure is *dispersion* which measures the degree to which the OWA operator takes into account all available information in the aggregation. To determine the weighting vector, we used the method proposed by O'Hagan [77], who defined a constrained non-linear optimization problem, where orness is constrained to be a predefined value and dispersion is the objective function to be maximized. Based on the setting of the weight vector, different aggregation methods are possible. For example when combining five values, the weight vectors $[1,0,0,0,0]$, $[0,0,0,0,1]$ and $[1/5,1/5,1/5,1/5,1/5]$ produce the maximum, minimum and average of the values respectively (the corresponding *orness* values are 1, 0 and 0.5). In our opinion retrieval system, we consider the K query positions with highest opinion density as the operands of the OWA operator. The value of K and the degree of $orness$ are the parameters of the model.

5.4.5 Average of Maximums

This variation uses the *Max* method to estimate the opinion score of a document with respect to each query term separately and then assigns the average of the opinion scores over the distinct query terms as the opinion score for the document with respect to the whole query:

$$P(i|D,Q) = \begin{cases} \frac{1}{|Q|} & \text{if } t_i \in Q \text{ and } i = \arg\max_{i' \in pos(t_i,D)} P(O|D,i') \\ 0 & \text{otherwise} \end{cases} \tag{5.22}$$

$$P(O|D,Q) = \sum_{i=1}^{|D|} P(i|D,Q)P(O|D,i) = \frac{1}{|Q|}\sum_{q \in Q} max_{i \in pos(q,D)} P(O|D,i) \tag{5.23}$$

A document that does not contain some of the query terms will have zero opinion density over the absent query terms and eventually a lower opinion score will be assigned to that document.

5.4.6 Smoothed Proximity Model

The proximity-based estimate can be further refined by smoothing it with the non-proximity-based estimation as follows:

$$p_{\lambda}(O|D,Q) = (1-\lambda)P(O|D,Q) + \lambda P(O|D) \tag{5.24}$$

Smoothing the proximity model with the non-proximity score lets us capture the proximity at different ranges. This can be useful because there are some documents in which the exact query term occurs rarely. In such documents opinion expressions refer to the query indirectly through anaphoric expressions such as *he, she, it, the film, etc*. Since we don't do any query expansion or reference resolution in our model, we investigate whether smoothing the proximity scores with the non-proximity scores helps us capture further related opinion expressions in the document.

5.5 Opinion Lexicon

As is the case for all lexicon-based opinion retrieval models, the choice of opinion lexicon can have a significant effect on the performance of the opinion retrieval system. In fact, if the opinion lexicon is not complete or does not assign proper opinion weights to terms, the model is not able to distinguish the opinionated parts of a document correctly.

In our experiments we used the opinion lexicon that was proposed by Lee et al. [52], called the KLE lexicon in this thesis. The KLE lexicon was built using both sentiWordNet [21] and an automatically learned model from the Amazon.com product review corpus. The opinion lexicon only contains terms from the review corpus that are also present in sentiWordNet (i.e. the intersection of Amazon and sentiWordNet word sets). The lexicon gives us the probability of opinionatedness of each term, $P(O|t)$, which can be used in our model.

We also considered using SentiWordNet. It assigns a positive, negative and objective scores, all between zero and one and sum to one, to every WORDNET synset s of a terms with a specific Part of Speech (POS) tag. The sum of the positive and negative scores can be used as the opinion score for a term with a specific sense s and POS tag a.

$$\text{opinion-score}(t,s,a) = \text{positive-score}(t,s,a) + \text{negative-score}(t,s,a) \tag{5.25}$$

Since we did not do any POS tagging or word sense disambiguation, we tried different methods for estimating an opinion score for a term without considering its POS tag or synset as follows:

We used the maximum opinion score of a term-sense pair across its different POS tags, as its opinion score.

$$P(O|t,s) = \max_{a} \text{opinion-score}(t,s,a) \tag{5.26}$$

We then tried various ways of estimating the probability of opinionatedness of a term based on its opinion score across different synsets. One method was to consider the maximum value across the different senses (denoted $syn(t)$) of a term:

$$p_{\text{max}}(O|t) = \max_{s \in syn(t)} P(O|s,t) \tag{5.27}$$

The intuition behind this method is that in the absence of information about the sense of a term in a context, we assume the term belongs to the most opinionated synset. Subscript max of $p_{\text{max}}(O|t)$ in Equation (5.27) indicates the usage of maximum opinion score of a term across its senses.

Since a term is not always used with its most opinionated sense in a document, the max method may lead to overestimating the opinion score of the term in a document. As an alternative, we considered a weighted average of the opinion score of a term across its different senses, where we give more weight to the most probable synset than the second and so on. In the formula below we use s_i to denote the i^{th} synset for a term:

$$p_{\text{wa}}(O|t) = \sum_{i=1}^{|syn(t)|} P(s_i, O|t) = \sum_{i=1}^{|syn(t)|} P(s_i|t)P(O|s_i,t) \tag{5.28}$$

where $p_{\text{wa}}(O|t)$ denotes the probability of opinionatedness of term t, estimated using weighted averaging method.

We rely on the synset number to estimate the probability that the sense of the term t in a context is s_i (i.e. $P(s_i|t)$):

$$P(s_i|t) = \frac{\frac{1}{i}}{\sum_{j=1}^{|syn(t)|} \frac{1}{j}} \tag{5.29}$$

We also experimented with a faster (quadratic) decrease in the weight of less probable synsets as follows:

$$P(s_i|t) = \frac{\frac{1}{i^2}}{\sum_{j=1}^{|syn(t)|} \frac{1}{j^2}} \tag{5.30}$$

Equations (5.29) and (5.30) assign more probability to the most common synset of a term than the second most common and so on.

Finally we considered a thresholding mechanism and assigned $P(O|t) = 1$ if the positive or negative score of t was higher than 0.6 in at least one sense. This method, used by Zhang and Ye [117], ignores the actual difference between the opinion weights of terms, except during the initial filtering process.

Table 5.1 reports the performance of the non-proximity opinion model using different lexicons. As we can see, the KLE lexicon is the best compared to SentiWordNet with different ways of estimating $P(O|w)$.

Lexicon	Description	Size	MAP
None	-		0.3403
KLE		8449	0.3606
SW_{max}	Eq.(5.27)	144308	0.3502
SW_{wa}	Eq.(5.28),(5.29)	39066	0.3509
SW_{wa2}	Eq. (5.28),(5.30)	39066	0.3499
$SW_{threshold}$	$P(O\|w) = 1$ if pos or neg score > 0.6 in at least one sense	7570	0.3476

Table 5.1. Opinion finding MAP of the non-proximity opinion retrieval model using different opinion lexicons.

5.6 Experimental Results

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

We start our experiments by investigating different methods for normalizing the relevance scores before combining them with the opinion scores in Section 5.6.1. Next, in Section 5.6.2 we analyze the effect of different proximity kernels on the opinion retrieval performance. We discuss the effect of the parameters of different aggregation models in Section 5.6.3. We evaluate our model on the Blog06 collection and the TREC 2008 query set. We compare our model to the relevance based retrieval and non-proximity opinion retrieval in Section 5.6.4 to see if the proximity information is effective in capturing the relevance of opinion terms to the query. Finally we compare the aggregation models in more details in Section 5.7.

5.6.1 Normalizing Relevance Scores

According to the Equation (5.1), the final ranking of documents should be based on the product of the relevance and opinion scores. The relevance scoring in Equation (5.1) is the query likelihood model. However, we do not want to be limited to the query likelihood model as the relevance scoring method. Our aim is to make our model general such that it can be applied on top of every relevance retrieval baseline and improve its performance in terms of opinion finding. In order to use relevance scores produced by different methods, we should first normalize the relevance scores to make them comparable with the opinion scores produced by the opinion scoring component of our model. Here we consider different models of normalizing the relevance scores. We then discuss the effectiveness of different normalization methods in Section 5.6.1 and choose the one that performs the best for each relevance retrieval method. Depending on the score distribution and the characteristics of the relevance retrieval method, the best normalization method for that can be different.

Table 5.2 shows the opinion retrieval performance of our proposed system, using different probability estimation methods on TREC baseline 4 (the best performing of the TREC baselines). We report the results for exponentially increasing values of sigma. In this table LRS, LRLS, LRMinMax and LRZ-score, LRR and LRLR denote logistic regression using the score, log of the score, MinMax and Z-score normalized scores, using the rank of documents instead of the score and the log of the rank as the explanatory variable respectively. As can be seen form Table 5.2, MinMax, Z-score and LRLR have the highest MAP over all sigma values on TREC baseline 4 and the improvement over the score is statistically significant, but there is no statistically significant difference between these three methods. We chose Z-score for TREC baseline 4 as it had the highest MAP over the training topics. For the other baselines, we found that LRLR performed best on Baseline1, 3 and 5, and Z-score on Baseline 2.

5.6.2 Parameter selection for the proximity-based opinion density estimation

In section 5.3 we proposed a proximity-based opinion propagation method in which a proximity kernel is considered around each opinion term occurrence position in the document. The opinion density at a query term position is then calculated by estimating the accumulated opinion density from different opinion terms at that position. In this way, a query term which occurs at a position close to many opinionated terms will receive high opinion density. The proposed

σ	8	16	32	64	128
Score	0.3262	0.3355	0.3364	0.3323	0.3304
MinMax	0.3641↑	0.3701↑	0.3708↑	0.3685↑	0.3662↑
Z-score	**0.3713↑**	**0.3732↑**	**0.3740↑**	**0.3720↑**	**0.3709↑**
LRS	0.3205↓	0.33↓	0.3307↓	0.3272↓	0.3254↓
LRLS	0.3216↓	0.3289↓	0.3285↓	0.3248↓	0.3228↓
LRMinMax	0.331↑	0.3402↑	0.3421↑	0.3395↑	0.3371↑
LRZ-score	0.2966↓	0.3048↓	0.3055↓	0.3012↓	0.2969↓
LRR	0.3324	0.33	0.3291	0.3297	0.3302
LRLR	0.3613↑	0.3672↑	0.3688↑	0.3682↑	0.3673↑

Table 5.2. MAP over TREC baseline4 using a Laplace kernel with different sigma values. Rows show MAP using different relevant probability estimation methods. An uparrow (↑) and a downarrow(↓) indicate statistically significant increase and decrease over using the score directly.

relevant opinion density estimation model has two parameters: the type of kernel function and its bandwidth parameter σ which adjusts the scope of opinion propagation (referencing) over the document. In this section we investigate the difference between different kernel types and the effect of the σ parameter on the training queries. The findings of this section will help us in choosing the best parameters to test the effectiveness of our models on test queries in section 5.6.4.

Table 5.3 reports the performance of the *Max* model using different kernels on training topics with the best parameter for each kernel. Results show that all proximity kernels improve significantly over the uniform kernel (i.e. non-proximity baseline). However, when using the best parameters for each kernel, there is no statistically significant difference between them.

Figure 5.3 reports the sensitivity (in terms of MAP) of different kernels to different values of the σ parameters ranging from 2 to 128. Although there was no statistically significant difference between kernels using the optimal σ value, the Laplace kernel has the most effective and is more robust in terms of MAP over a wide parameter range.

The smoothed probability model explained in section 5.4.6 introduces another parameter λ which controls the effect of the non-proximity model. In order to find the best parameters, we tested different λ values for each σ value in the range of [2 , 128]. Table 5.4 reports the performance of different kernels using the best σ and λ parameter pairs for each kernel. It shows that, smoothing

kernel	σ	MAP	R-prec	bPref	P@10
Uniform	∞	0.3606	0.4011	0.4231	0.6190
Laplace	22	**0.3744**$*$	**0.4113**	**0.4305**	**0.6200**
Gaussian	26	0.3730$*$	0.4099	0.4305$*$	0.6160
Cosine	24	0.3729$*$	0.4095	0.4305	0.6170
Triangle	24	0.3728$*$	0.4086	0.4302	0.6180
Square	28	0.3728$*$	0.4100	0.4300	0.6130
Circle	16	0.3723$*$	0.4080	0.4298	0.6120

Table 5.3. The performance of the *Max* proximity-based opinion retrieval model for the best σ for each kernel over the baseline4. Symbol $*$ indicates statistically significant improvement over uniform kernel.

the proximity scores with the non-proximity opinion scores improves the performance. It also shows that there is no statistically significant difference between kernels when the proximity score is interpolated with the general opinion score of the document.

kernel	λ	σ	MAP	R-prec	bPref	P@10
Laplace	0.4	12	0.3775	0.4166	0.4325	0.6400
Gaussian	0.6	4	0.3772	0.4147	0.4317	0.6360
Triangle	0.5	4	0.3764	0.4121	0.4317	0.6420
Cosine	0.6	4	0.3762	0.4142	0.4318	0.6390
Circle	0.7	4	0.3764	0.4167	0.4333	0.6360
Square	0.4	18	0.3757	0.4092	0.4326	0.6230

Table 5.4. Performance of the *Max* smoothed model for the best σ and λ for each kernel.

Here we have shown that the Laplace kernel had the highest and most stable performance for the *Max* method. Although the results are reported for *Max* method, we believe they generalize across all proposed aggregation models. Therefore, we use Laplace kernel throughout the rest of the experiments for all aggregation models.

5.6.3 Parameter selection for the aggregation models

The *Max*, *Ave* and *AveMax* aggregation models do not introduce any extra parameters. The *RWA* model needs another parameter which is the bandwidth

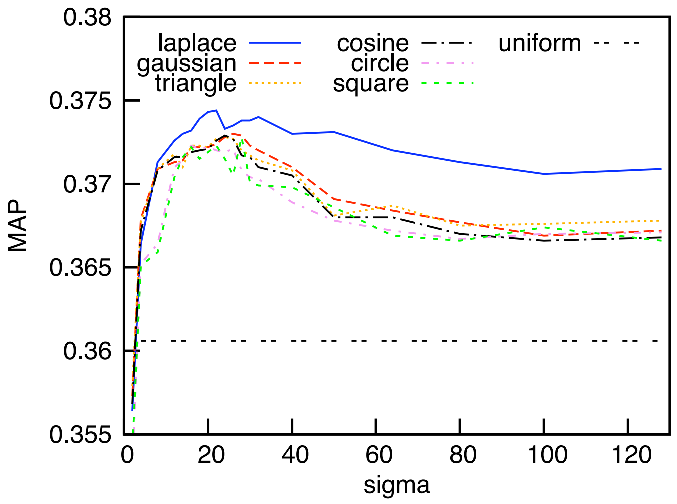

Figure 5.3. Parameter sweep for different kernel functions used in *Max* method over the baseline4.

parameter for the propagation of query terms used in the Positional Language Model, σ_{plm}. We found the model not to be very sensitive to this parameter and fixed it to the value 5.

The *OWA* aggregation model for estimating the opinion score of a document has two extra parameters: the number of highly opinionated query points that should be taken into account, K, and the degree of $orness$. We tried K values in the range [2 , 30] and also $orness$ values between 0 and 1 with step of size 0.1 over the training set of topics. To report the result for the test set of queries, we choose the pair of parameters that optimize MAP over the training queries. To better understand the behavior of the model and compare the importance of parameter tuning for K and $orness$ we carry out the following analysis. We first select the best value of $orness$ and plot the MAP value over different values of K in Figure 5.4. Then in Figure 5.5 we choose the best value of K and plot MAP obtained by different values of the parameter $orness$.

Figures 5.4 and 5.5 show that MAP changes from 0.2757 to 0.3737 when changing $orness$ from 0 to 1 while MAP changes from 0.3737 to 0.3814 by changing the K parameter from 1 to 30.

Results also show that using more than one query position is always effective and can improve the performance over the Max model if we choose a good value for the $orness$ parameter. This result is reasonable based on the role of $orness$ in the *OWA* aggregation operator. We know that *OWA* with $orness = 1$ simulates the *Max* operator and so, no matter what the value of parameter K, only the

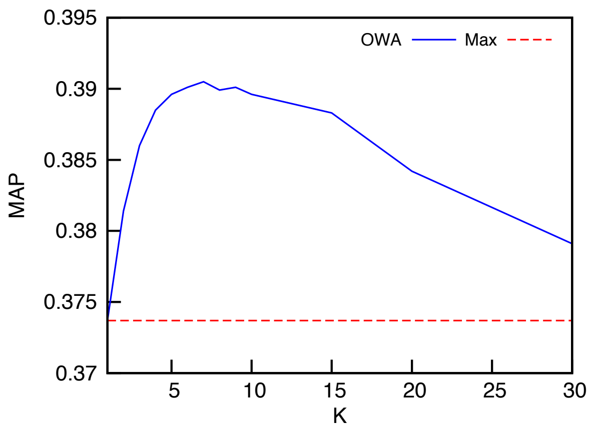

Figure 5.4. Sensitivity to the parameter *K* while *orness* is fixed to best value

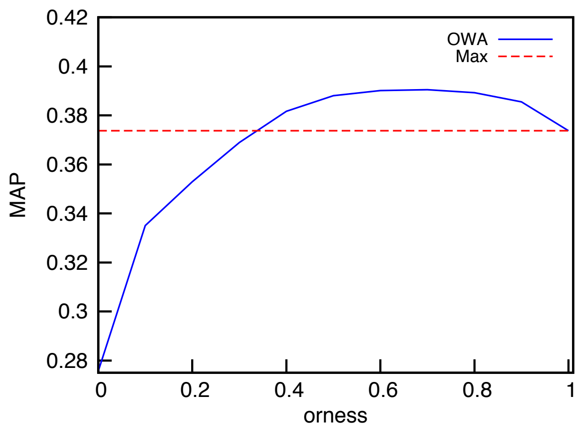

Figure 5.5. Sensitivity to the *orness* while parameter *K* is fixed to best value

most opinionated query position is taken into account. However, when we use more query positions (i.e high value of K), low value of $orness$ can simulate the *Ave* model and so it performs worst than using the highest opinionated query position. Therefore, we expect to see better or at least comparable performance to the *Max* method by having $orness$ close to 1. In fact the best value of $orness$ has been shown to be 0.7, in our setting.

From these experiments we conclude that setting the $orness$ parameter to a proper value is more important than finding the best value for parameter K. In fact, even if the value of K is set to a larger value than its optimum value, the $orness$ parameter can control the amount of weight that is given to the low opinionated positions.

5.6.4 Results on 2008 Topics

Table 5.5 presents the results of evaluating our approaches on the TREC 2008 query topics in terms of opinion retrieval performance. The results are categorized in five groups, each related to one of the standard TREC baselines. For each group, we report the opinion retrieval performance in two categories:

1. Baselines and upperbounds:
   (a) *baseline$_i$*: raw relevance retrieval ranking without any added opinion finding feature; where i is from 1 to 5 according to the naming by TREC.
   (b) *upper bound*: reranking the initial relevance ranking based on relevance judgement such that all relevant posts appear at the top of the rank list. Since we rerank the top 1000 relevant posts, it is useful to see what is the upper bound of every measure since we may not have had enough relevant posts retrieved in the top 1000 to achieve the highest performance possible.
   (c) *no-prox*: an opinion retrieval model which does not use proximity information over the corresponding relevance retrieval baseline. We use this as a stronger baseline than the relevance ranking since it is using opinion terms in scoring documents.
2. The proximity-based opinion retrieval model with different aggregation methods *Max, Ave, RWA, OWA* and *AveMax*.

Note that for every method, we select the parameters that optimize *MAP* on the training set of queries (i.e. TREC 2006 and TREC 2007 topic sets). We report the corresponding *MAP* as well as R-precision, bPref and P@10 on test topics in Table 5.5. The results for measures other than *MAP* are reported for comparison and are not necessarily optimized.

We first compare the performance of opinion retrieval models with the initial relevance-based ranking in terms of opinion retrieval. We can see that the proximity-based model is able to improve the initial relevance-based ranking of all baselines using any of the proposed aggregation methods. The improvement is statistically significant for almost all aggregation methods across all baselines except for baseline 5 where we see statistically significant improvement only in the case of the *AveMax* model. Comparing the performance of the non-proximity based opinion retrieval with the relevance ranking performance shows that the non-proximity model is able to improve the initial relevance ranking in 4 out

	MAP	R-prec	bPref	P@10
baseline1	0.3239	0.3682	0.3514	0.5800
upperbound	0.7578	0.7578	0.7578	0.9860
noprox	0.3751∗	0.4154	0.4082∗	0.6720∗
Max	0.3961∗	0.4313∗	0.4286∗	0.6840∗
Ave	0.3952∗†	0.4347∗†	0.4329∗†	0.6980∗
RWA	0.3917∗	0.4289∗	0.4296∗	0.7000∗
OWA	0.4020∗†	0.4389∗	0.4253∗	0.6940∗
AveMax	**0.4062**∗†	**0.4425**∗†	**0.4372**∗†	**0.7140**∗
baseline2	0.2639	0.3145	0.2902	0.5500
upperbound	0.7199	0.7199	0.7199	1.0000
noprox	0.2791	0.3299	0.3066	0.5740
Max	0.2849∗	0.3379	0.3141∗	0.5780
Ave	0.2972∗†	0.3484∗†	0.3293∗†	0.6020
RWA	0.2887∗†	0.3432∗	0.3186∗†	0.6060∗†
OWA	0.2872∗†	0.3366†	0.3149∗†	0.5980
AveMax	**0.2955**∗†	**0.3532**∗†	**0.3248**∗†	**0.6080**∗
baseline3	0.3564	0.3887	0.3677	0.5540
upperbound	0.8395	0.8395	0.8395	1.0000
noprox	0.3819	0.4188	0.4075∗	0.6400∗
Max	0.3976∗	0.4347∗	0.4209∗	0.6460∗
Ave	0.4075∗†	0.4436∗†	0.4395∗†	0.6400
RWA	0.4040∗†	0.4404∗†	0.4348∗†	0.6360
OWA	0.3984	0.4380	0.4217∗	0.6560∗
AveMax	**0.4218**∗†	**0.4589**∗†	**0.4445**∗†	**0.6660**∗
baseline4	0.3822	0.4284	0.4112	0.6160
upperbound	0.8405	0.8405	0.8405	0.9960
noprox	0.4129∗	0.4460	0.4368∗	0.6880∗
Max	0.4267∗	0.4545∗	0.4472∗	0.7080∗
Ave	0.4217∗	0.4634∗†	0.4509∗†	0.6980∗
RWA	0.4191∗	0.4548∗	0.4465∗	0.6840∗
OWA	0.4297∗	0.4619∗†	0.447∗	0.7060∗†
AveMax	**0.4376**∗†	**0.4695**∗†	**0.4601**∗†	**0.7100**∗
baseline5	0.2988	0.3524	0.3395	0.5300
upperbound	0.7234	0.7234	0.7234	1.0000
noprox	0.2918	0.3455	0.3497	0.5980
Max	0.3170†	0.3688†	0.3648†	0.6020
Ave	0.3091†	0.3620†	0.3642†	0.5900
RWA	0.3078†	0.3569	0.3628†	0.5880
OWA	0.3289†	0.3835†	0.3753†	0.6240∗
AveMax	**0.3566**∗†	**0.4117**∗†	**0.3978**∗†	**0.6540**∗†

Table 5.5. Opinion finding MAP results over five standard TREC baselines using different proximity methods for TREC 2008 topics. A star(∗) and dagger(†) indicate statistically significant improvement over the relevance and non-proximity opinion retrieval respectively.

of 5 baselines in terms of MAP and the improvement is statistically significant for just two baselines: baseline1 and baseline4. This indicates that improving the initial relevance-based ranking is not always easy and requires a more sophisticated method than counting all opinion expressions in a document. As a stronger baseline, we consider the non-proximity opinion retrieval model and compare its performance to the performance of our models. We see that all our proximity-based models improve over the non-proximity opinion retrieval model across all baselines. The improvement is statistically significant in most cases for baselines 1, 2, 3 and 5. The exception is baseline4 where the only method that results in statistically significant improvement over the non-proximity model is the *AveMax* method.

By comparing the performance of different aggregation methods in our proximity based model we realized that the *AveMax* method is the most effective method across all the baselines and is able to improve significantly both relevance and non-proximity opinion retrieval rankings for all baselines and for almost all performance measures.

Next we investigate the effect of smoothing our proposed proximity-based models with the non proximity-based opinion retrieval score. The results are reported in Table 5.6. Results indicate that smoothing can help to improve the performance in most cases, especially in the case of re-ranking baseline1. However the improvement is marginal and not statistically significant.

Relevance Ranking	Opinion reranking	Max	Ave	RWA	OWA	AveMax
Baseline1	proximity model	0.3961	0.3952	0.3917	0.4020	0.4062
	smoothed proximity	**0.4016**	**0.3959**	**0.3923**	**0.4050**	**0.4117**
Baseline2	proximity model	0.2849	0.2972	0.2887	0.2872	0.2955
	smoothed proximity	0.2849	0.2972	0.2887	0.2872	0.2955
Baseline3	proximity model	0.3976	0.4075	0.4040	0.3984	0.4218
	smoothed proximity	**0.4039**	0.4075	0.4040	**0.4020**	**0.4244**
Baseline4	proximity model	0.4267	0.4217	0.4191	0.4297	0.4376
	smoothed proximity	**0.4292**	0.4217	0.4190	0.4275	0.4364
Baseline5	proximity model	0.3170	0.3091	0.3078	0.3289	0.3566
	smoothed proximity	**0.3215**	0.3091	0.3078	**0.3306**	0.3565

Table 5.6. Comparing the MAP value of the ranking obtained by smoothed and non-smoothed proximity models on TREC 2008 topics.

A summary of the results reported in Tables 5.5 and 5.6 is as follows: the proposed proximity-based opinion retrieval models are consistently effective across all five standard TREC baselines and can improve not only the purely relevance retrieval baselines but also the non-proximity opinion retrieval system.

The most effective aggregation method for the proximity-based opinion retrieval was found to be the *AveMax* method. However, we did not notice statistically significant difference between the aggregation methods. Smoothing with the non-proximity model was shown to be helpful but the improvements were not significant.

Finally we compare our proposed models with the best runs at TREC 2008 blog track and report the comparison result in Table 5.7 and Table 5.8. Table 5.7 shows the performance of our proposed methods on the standard TREC baseline4, comparing to the best TREC run, named B4PsgOpinAZN, and a later proposed method in [106], named KLD+dist-FD-FV-subj-b4, on the same baseline. Interestingly, all our proximity-based models outperform B4PsgOpinAZN and KLD+dist-FD-FV-subj-b4. B4PsgOpinAZN is based on a query specific lexicon which is built via feedback-style learning. KLD+dist-FD-FV-subj-b4 uses Wikipedia for finding different facets of the query and query expansion. It then uses Kullback-Leibler divergence to weight subjective units occurring near query terms. The distance of the query term to the subjective units is also considered in this model.

Run	Map	Δ MAP
AveMax	0.4376	14.49%
OWA	0.4322	13.08%
Max	0.4267	11.64%
Ave	0.4217	10.33%
RWA	0.4191	9.65%
KLD+dist-FD-FV-subj-b4	0.4229	10.65%
B4PsgOpinAZN	0.4189	9.60%

Table 5.7. Opinion finding results for best runs on standard baseline 4, ranked by Mean ΔMAP using the TREC 2008 (test) topics.

Table 5.8 reports the mean MAP and the mean relative improvement over the five standard baselines (ΔMAP). We observe that the proposed methods have the highest mean of MAP and mean of ΔMAP across the five standard baselines. This indicates that the proposed methods are effective and stable across different relevance retrieval baselines.

Run	Map		Δ MAP	
	Mean	stdev	Mean	stdev
AveMax	0.3848	0.06	18.35%	4.98%
OWA	0.3707	0.06	13.90%	5.87%
Ave	0.3661	0.06	12.55%	6.72%
Max	0.3657	0.06	12.34%	5.94%
RWA	0.3622	0.06	11.27%	6.55%
uicop1bl1r	0.3614	0.04	11.76%	6.93%
B1PsgOpinAZN	0.3565	0.05	9.67%	0.77%

Table 5.8. Opinion finding results for best runs using all five standard baselines, ranked by Mean ΔMAP using the TREC 2008 (test) topics.

5.7 Analysis and Discussion

In the previous section we reported the average performance of our proposed proximity-based models on 50 query topics. Results showed that all the proximity methods were effective and could improve both relevance and non-proximity opinion retrieval models. Also, all our proposed models performed better than the state of the art models. To better understand the difference between the different aggregation methods, in Section 5.7.1 we perform a detailed analysis and look into the performance of every topic separately. Our aim is to find which method is the most effective for improving the performance of the largest number of query topics. We would also like to see which topics are "helped" or "hurt" more by each aggregation method. To further understand the differences between aggregation models, in Section 5.7.2 we analyze the performance of different methods versus topic length (i.e. the number of terms in the query). Finally we perform a post-level analysis and look into the posts related to the topics that are hurt using proximity-based opinion retrieval.

5.7.1 Topic-based Analysis

In this section we perform a per topic analysis and compare the proposed aggregation techniques for estimating a document opinion score.

Figure 5.6 shows the increase or decrease in *Average Precision (AP)* and *Precision at 10* when comparing different aggregation models in the proximity-based opinion retrieval model to the baseline4. The plot shows that all proposed aggregation techniques have topics for which the proximity model can improve over the simple relevance baseline as well as topics for which the proximity-based

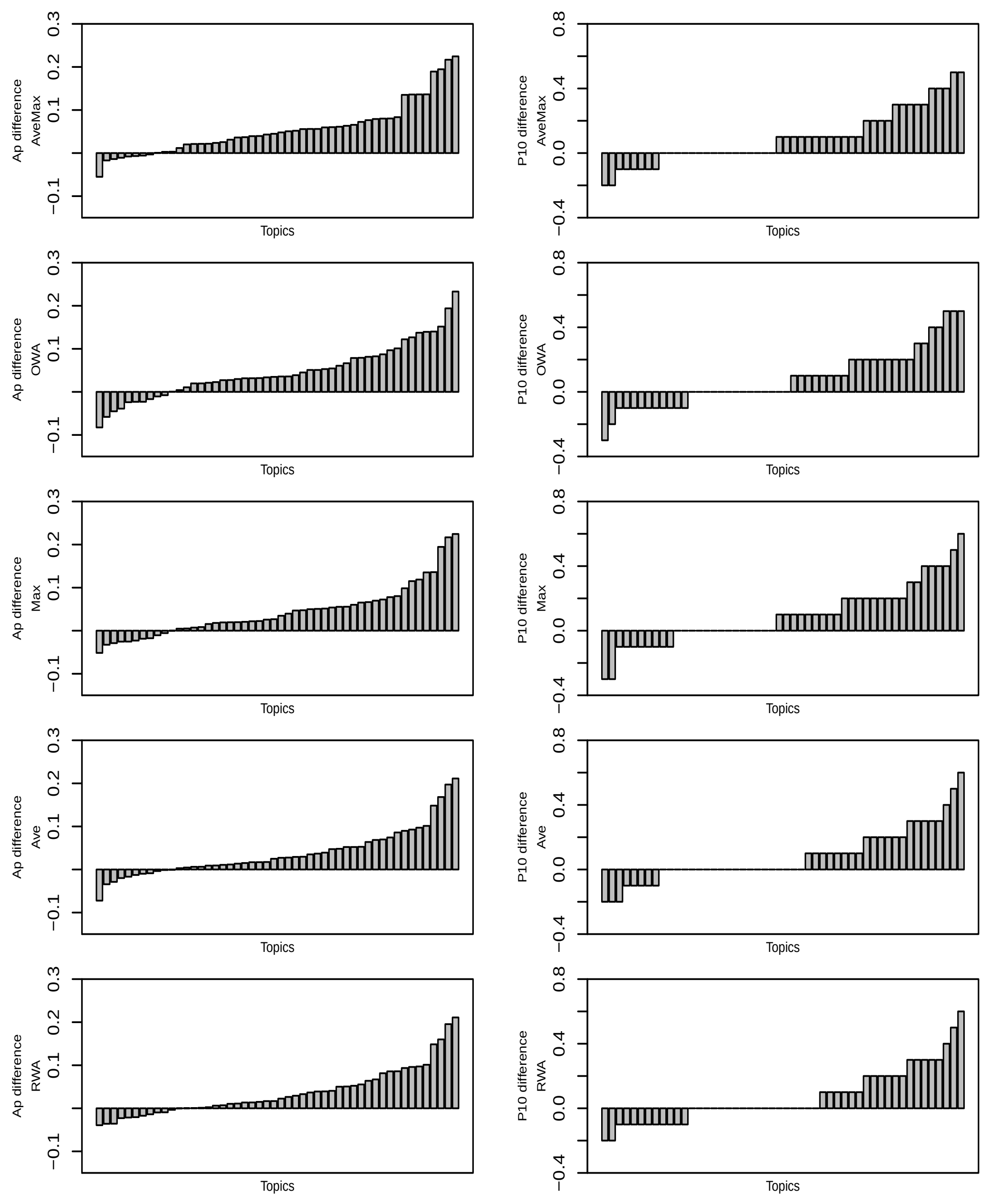

Figure 5.6. Comparing the Proximity-based model with different aggregation methods *AveMax, OWA, Max, Ave* and *RWA* against relevance baseline4 for TREC 2008 queries. Positive/negative bars indicate improvement/decline over the baseline in the reported measure.

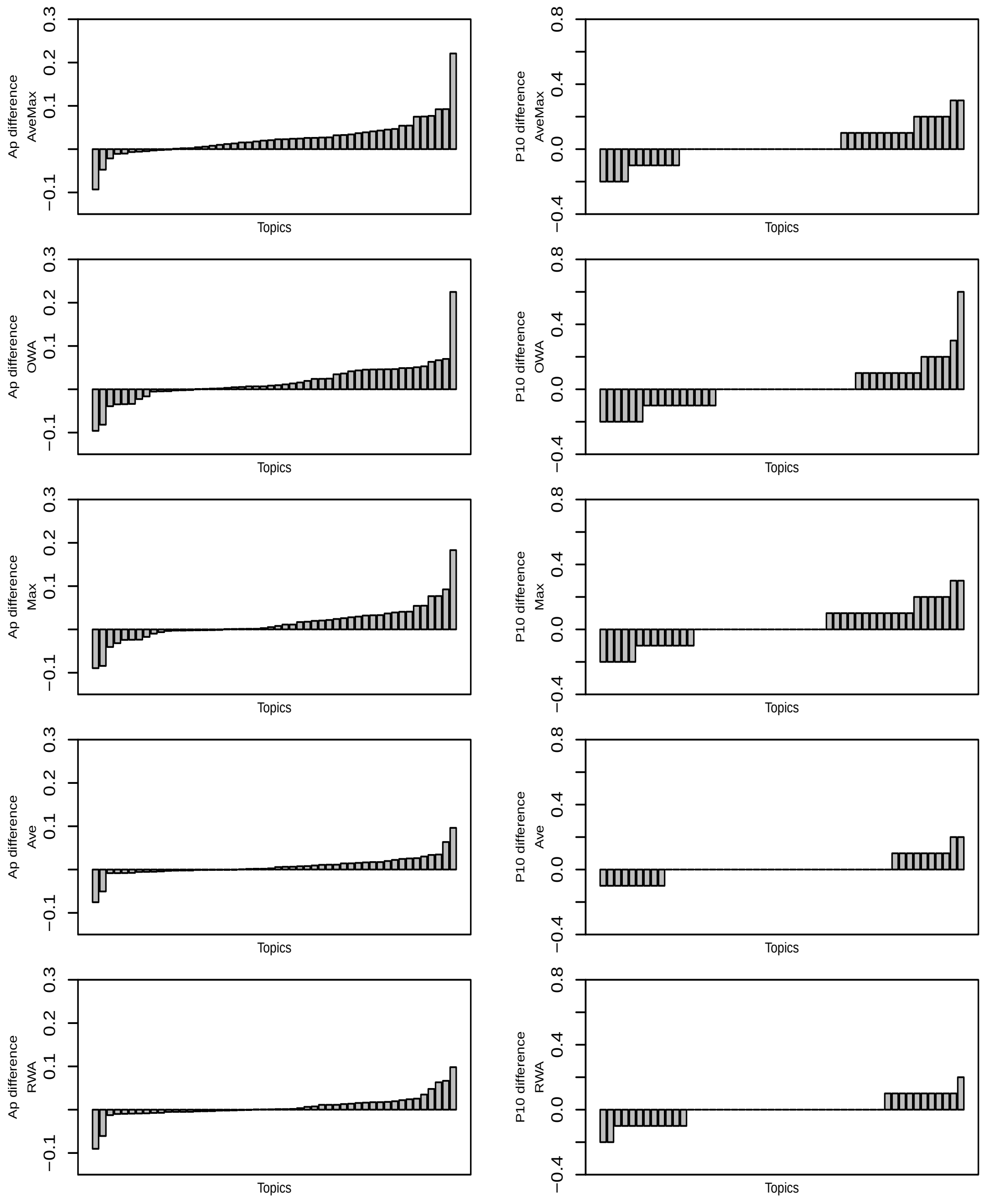

Figure 5.7. Comparing the Proximity-based model with different aggregation methods *AveMax, OWA, Max, Ave* and *RWA* against the non-Proximity opinion retrieval model for TREC 2008 queries. Positive/Negative bars indicate improvement/decline over the baseline in the reported measure.

model is not helping and even hurts the performance. However, all aggregation models have more topics for which they improve performance compared to the number of topics that are hurt. This shows that in general the proximity model is helpful and this does not depend on the choice of aggregation model. Comparing different aggregation methods, it seems that the *AveMax* method is the most effective, improving more and hurting fewer topics compared to other methods. The *RWA* method seems to be the least effective. The number of topics that were helped or hurt by these methods are reported in Table 5.9.

Method	MAP		Rprec		bPref		P10	
	helped	hurt	helped	hurt	helped	hurt	helped	hurt
AveMax	42	8	37	7	45	4	26	8
OWA	40	10	36	12	37	13	24	12
Max	39	11	33	14	37	13	26	10
Ave	39	11	34	7	39	11	22	8
RWA	32	12	31	9	37	12	20	12

Table 5.9. Number of TREC 2008 topics that were helped or hurt by the opinion retrieval methods compared to relevance baseline4 with respect to different performance measures.

To better understand the difference of the aggregation methods, we consider a stronger baseline to be the non-proximity opinion retrieval model. Figure 5.7 shows the increase and decrease in *Average Precision* and *Precision at 10* when comparing different aggregation methods in the proximity based opinion retrieval to the non-proximity based model. Table 5.10 reports the number of topics that were helped or hurt by the methods compared to the non-proximity opinion retrieval.

Method	MAP		Rprec		bPref		P10	
	helped	hurt	helped	hurt	helped	hurt	helped	hurt
AveMax	39	11	35	8	38	12	17	11
OWA	36	14	28	14	34	16	15	16
Max	32	18	30	12	30	19	19	13
Ave	30	20	27	10	33	16	10	9
RWA	28	22	21	12	33	17	11	12

Table 5.10. Number of TREC 2008 topics that were helped or hurt by the opinion retrieval methods compared to the non-proximity based model with respect to different performance measures.

From the plot and the table we can see that the number of topics that are helped by the proximity models is mostly higher than the number of topics that are hurt by them, over most performance measures. Among the aggregation methods, *AveMax* seems to be the most effective followed by *OWA* and *Max*.

Tables 5.11, 5.12, 5.13, 5.14 and 5.15 show topics that were helped or hurt the most using *Max*, *AveMax*, *OWA*, *Ave* and *RWA* in the proximity-based opinion retrieval model compared to the non-proximity model respectively. We chose threshold 0.03 and 0.01 of ΔAP for reporting the helped and hurt topics respectively.

Num	Title	ΔAP	Num	Title	ΔAP
1003	Jiffy Lube	0.1831	1013	Iceland European Union	−0.0895
1001	Carmax	0.0925	1031	Sew Fast Sew Easy	−0.0843
1002	Wikipedia primary source	0.0771	1045	Women on Numb3rs	−0.0406
1023	Yojimbo	0.0768	1021	Sheep and Wool Festival	−0.0318
1037	New York Philharmonic Orchestra	0.0548	1008	UN Commission on Human Rights	−0.0242
1010	Picasa	0.0544	1015	Whole Foods wind energy	−0.0239
1049	YouTube	0.0410	1016	Papa John's Pizza	−0.0236
1039	The Geek Squad	0.0407	1032	I Walk the Line	−0.0173
1014	tax break for hybrid automobiles	0.0391			
1040	TomTom	0.0368			
1028	Oregon Death with Dignity Act	0.0330			
1027	NAFTA	0.0326			
1048	Sopranos	0.0320			

(a) helped *(b)* hurt

Table 5.11. Topics that are helped or hurt the most in Max compared to the non-proximity opinion retrieval model.

Looking at the topics that are helped by different methods compared to the non-proximity model shows that, there is no topic that is helped just by the *Max* or *RWA* methods. Topic 1019, *China one child law*, is only helped with method AveMax and it is hurt using any other method. Topic 1016, *Papa John's Pizza*, is only helped with method *Ave*. Topics 1043, *A Million Little Pieces*, and 1045, *Women on Numb3rs*, are only helped using method *OWA*. Looking at the topics that are hurt by different methods compared to non-proximity model shows that there is no topic hurt only by *Max* or *AveMax*. On the other hand, four topics 1014, *tax break for hybrid automobiles*, 1041, *federal shield law*, 1036, *Project Runway* and 1028, *Oregon Death with Dignity Act*, are just hurt by *RWA* method. Also two topics 1009, *Frank Gehry architecture*, and 1044, *talk show hosts* are just hurt by the *Ave* method. The *OWA* method also hurt topic 1012, *Ed Norton* which is not hurt using any other method.

Table 5.16 shows topics that are helped or hurt by all aggregation models compared to the non-proximity based model. MeanΔAP in Table 5.16 indicates

Num	Title	ΔAP
1003	Jiffy Lube	0.2209
1001	Carmax	0.0925
1002	Wikipedia primary source	0.0921
1023	Yojimbo	0.0768
1014	tax break for hybrid automobiles	0.0753
1037	New York Philharmonic Orchestra	0.0749
1010	Picasa	0.0544
1041	federal shield law	0.0541
1015	Whole Foods wind energy	0.0466
1030	System of a Down	0.0451
1039	The Geek Squad	0.0429
1049	YouTube	0.0410
1022	Subway Sandwiches	0.0388
1040	TomTom	0.0368
1046	universal health care	0.0340
1027	NAFTA	0.0326
1048	Sopranos	0.0320

(a) helped

Num	Title	ΔAP
1013	Iceland European Union	−0.0931
1045	Women on Numb3rs	−0.0475
1011	Chipotle Restaurant	−0.0214
1016	Papa John's Pizza	−0.0109
1032	I Walk the Line	−0.0102

(b) hurt

Table 5.12. Topics that are helped or hurt the most in AveMax compared to the non-proximity opinion retrieval model.

the average of ΔAP across all aggregation methods for every topic. As we can see from the table, there are 17 topics that are helped by all proximity-based models and only 4 topics that none of the proximity-based models could improve. We will discuss the reason behind failure of all our proximity methods in handling these 4 queries in Section 5.7.3.

5.7.2 Topic Length-based Analysis

We also compare the performance of different aggregation methods on queries with different length. Table 5.17 shows the number of queries between 150 TREC queries with a specific length. Figure 5.8 compares different aggregation models in improving the non-proximity opinion retrieval model based on different query lengths.

As we can see in Figure 5.8, most of the aggregation methods improve the non-proximity opinion retrieval model across all topic lengths. The exception is the *Max* method which hurts the performance of long queries (i.e. length 4). This was expected since the *Max* method is only using the most opinionated query position to score the document and does not take into account the opinion density at other query terms positions. Thus, it is not able to differentiate between a document that just contains one of the query terms and one that has all query terms. This can be problematic in queries with more than one term.

Num	Title	ΔAP
1003	Jiffy Lube	0.2248
1010	Picasa	0.0701
1002	Wikipedia primary source	0.0671
1027	NAFTA	0.0634
1040	TomTom	0.0530
1048	Sopranos	0.0509
1014	tax break for hybrid automobiles	0.0491
1045	Women on Numb3rs	0.0489
1004	Starbucks	0.0466
1023	Yojimbo	0.0462
1049	YouTube	0.0460
1028	Oregon Death with Dignity Act	0.0457
1009	Frank Gehry architecture	0.0453
1022	Subway Sandwiches	0.0435
1039	The Geek Squad	0.0415
1037	New York Philharmonic Orchestra	0.0364
1018	MythBusters	0.0345

(a) helped

Num	Title	ΔAP
1013	Iceland European Union	−0.0960
1021	Sheep and Wool Festival	−0.0818
1016	Papa John's Pizza	−0.0391
1031	Sew Fast Sew Easy	−0.0348
1034	Ruth Rendell	−0.0344
1012	Ed Norton	−0.0338
1008	UN Commission on Human Rights	−0.0226
1025	Nancy Grace	−0.0164

(b) hurt

Table 5.13. Topics that are helped or hurt the most in OWA compared to the non-proximity opinion retrieval model.

Num	Title	ΔAP
1003	Jiffy Lube	0.0963
1023	Yojimbo	0.0636
1042	David Irving	0.0347
1002	Wikipedia primary source	0.0337
1014	tax break for hybrid automobiles	0.0301

(a) helped

Num	Title	ΔAP
1045	Women on Numb3rs	−0.0755
1013	Iceland European Union	−0.0506

(b) hurt

Table 5.14. Topics that are helped or hurt the most in Ave compared to the non-proximity opinion retrieval model.

In fact the *Max* method performs the best over the single-term queries. The performance improvement of the *Max* method diminishes as the query length increases. It even hurts the performance of queries with length 4 compared to the non-proximity method.

The *OWA* method is the most effective method in handling single-term queries compared to other methods. It performs better than *Max* for all query length categories. The behaviour of the *AveMax* is consistent across different query lengths. It performs better than all methods in handling long queries (i.e. length > 2). For queries with length 1, the performance of *AveMax* and *Max* is exactly the same and this is expected since there is only one query term and there is no averaging involved that differentiates the two methods. *OWA* performs better than *AveMax* for queries of length 2 and the reason is that this category of

Num	Title	ΔAP	Num	Title	ΔAP
1003	Jiffy Lube	0.0982	1045	Women on Numb3rs	−0.0904
1002	Wikipedia primary source	0.0667	1013	Iceland European Union	−0.0608
1023	Yojimbo	0.0632	1006	Mark Warner for President	−0.0126
1025	Nancy Grace	0.0479			

(a) helped *(b)* hurt

Table 5.15. Topics that are helped or hurt the most in RWA compared to the non-proximity opinion retrieval model.

Num	Title	MeanΔAP	Num	Title	MeanΔAP
1003	Jiffy Lube	0.1647	1013	Iceland European Union	−0.0780
1002	Wikipedia primary source	0.0673	1032	I Walk the Line	−0.0086
1023	Yojimbo	0.0653	1020	intelligent design	−0.0049
1010	Picasa	0.0436	1038	israeli government	−0.0023
1037	New York Philharmonic Orchestra	0.0402			
1049	YouTube	0.0344			
1048	Sopranos	0.0327			
1040	TomTom	0.0319			
1027	NAFTA	0.0302			
1039	The Geek Squad	0.0298			
1042	David Irving	0.0246			
1018	MythBusters	0.0193			
1035	Mayo Clinic	0.0165			
1026	flag burning	0.0162			
1024	Zillow	0.0151			
1029	Morgan Freeman	0.0098			
1047	Trader Joe's	0.0068			

(a) helped *(b)* hurt

Table 5.16. Topics that are helped or hurt with all aggregation methods compared to the non-proximity opinion retrieval model.

queries are usually entity names such as *Steve Jobs* or *Morgan Freeman*. In these cases, authors usually mention the full names at the beginning and then refer to the entity using the fist name only. Thus, it is natural that a model that perform the best over single-term queries performs the best here as well.

5.7.3 Document-based Analysis

As we have seen in Section 5.7.1, there are 4 topics that are hurt by all our proximity-based models compared to the non-proximity based opinion retrieval model. To investigate the reason behind this failure we look into the relevant blog posts for these topics in more detail.

According to Table 5.16, Topic 1013 is the most hurt on average across all ag-

Length	Number of queries
1	49
2	73
3	22
4	6

Table 5.17. Number of queries with specific length.

gregation models compared to the non-proximity model. This topic has just 11 opinionated and relevant posts of which 8 are retrieved by baseline4. Comparing the re-ranking of the non-proximity model and the proximity model, *AveMax* showed that all retrieved relevant opinionated posts are ranked at the top of the rank list (1 to 10 for non-prox model and 1 to 11 for AveMax). The difference between the two rank lists is the post *BLOG06-20060118-042-0012699367* which is judged as non relevant in TREC assessments whose position changed from 5 in non-proximity rank list to 1 in the AveMax ranking list. This causes the shift in the position of the top four relevant posts toward down the list. We looked into the content of this non relevant post and found it to contain all query terms: *Iceland, European, Union*. However, the post is mostly discussing the difference between Europe and European Union and why some people confuse these two concepts. It starts by the example that “Icelandic Europhiles preach that Iceland must join Europe as soon as possible".

The other difference between the two rank list was caused by the non relevant post *BLOG06-20051207-126-0009326437* whose position changed from 11 in no-prox ranking to 7 in the *AveMax* ranking. This again caused the shift in the position of all the relevant posts downward in the *AveMax* rank list. Looking into the content of this post we noticed that the post is not about the topic. However, in one of the post’s comments all query terms and a lot of opinion terms occur. The comment is depicted in Table 5.18. As we can see, the only way to improve the model to ignore this comment would be to interpret the semantics of the sentences.

The next topic which was hurt the most by proximity-based models is Topic 1032. We looked into the relevant posts of this topic whose positions shifted more dramatically downward in the list. One such post is depicted in Table 5.19. The references to the query are highlighted in red and the opinion terms are underlined. As we can see, the query terms which reflect the name of a movie are mentioned twice at the beginning of the post and later all references to the query use other terms such as *movie, film* and *it*. Thus, while the whole

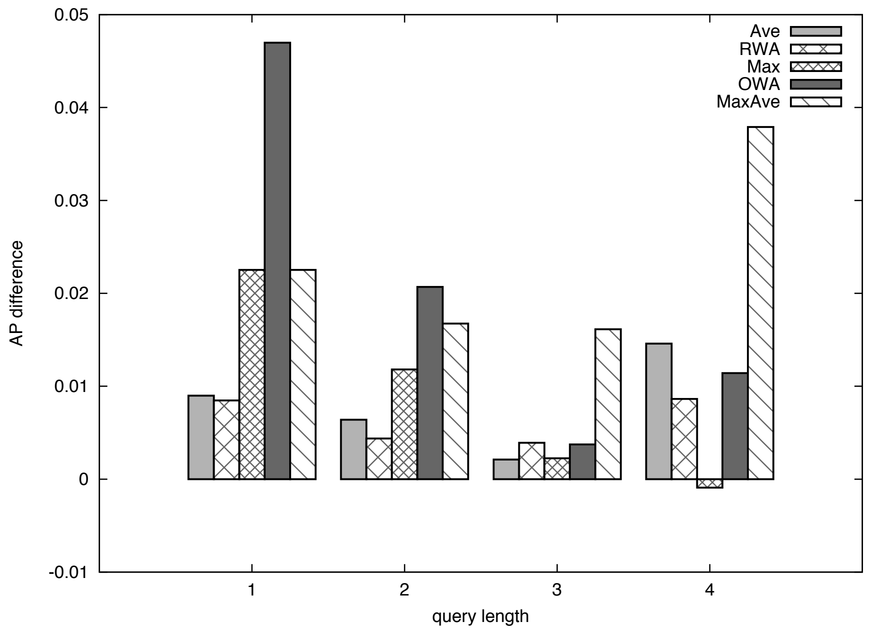

Figure 5.8. Comparing the proximity-based opinion scoring model with different aggregation methods to the non-proximity opinion scoring on topics with different length

post is relevant to the movie and is expressing an opinion about it, the proximity-based models just take into account the opinion density at query term positions. This could be improved by finding the co-refferential terms with the query using query expansion. A similar problem was noticed for topics *1020* and *1038* where terms other than query terms were used in the posts to refer to the query or related concepts. Therefore, the proximity-based models that do not take those terms into account underestimate the opinionatedness of such documents.

5.8 Conclusions

In this chapter, we proposed a novel probabilistic model for blog post opinion retrieval. We focused on the problem of opinion topic relatedness and proposed estimating the opinion density at query term positions by taking into account both the weight and proximity of opinion terms. We discussed the possibility of using different proximity kernels in the model. We also proposed the Laplace kernel which has not been used in previous studies. We discussed and experimentally showed that when kernels are compared using the best parameter,

Lennart said...
You mention that you have dual citizenship, but the US does not recognize dual citizenship with very few exceptions, Israel being the only one I know of, although all European Union countries do. So I do not think that the US would necessarily recognize your Icelandic passport. Iceland would, however, recognize your US passport. Just curious.

Table 5.18. An excerpt from Blog post BLOG06-20051207-126-0009326437: A non-relevant blog post to query topic 1013 which has been moved from position 11 in non-proximity ranking to position 7 in the MaxAve ranking. Here terms in red are query terms while underlined terms are opinion terms according to our opinion lexicon.

Walk the Line
With an incredible performance by Joaquin Phoenix Walk the Line tells the story of Johnny Cash. Phoenix doesn't so much become Cash as becomes a force of nature that must be paid attention to. Its as if Cash were a song that Phoenix was singing and making his own. Its sure to be a front runner for the Oscar with Philip Seymour Hoffman's Capote (Hoffman will probably win but Phoenix with stay in the gut of movie goers when Capote is forgotten). While the film is better than last years Ray, the film still suffers from the problem many recent bio-films have encountered and that is a reason for watching. Certainly the performance is enough , but even after watching the movie I wasn't sure why I was being told Cash's story. "I'm being told this because?" I wasn't certain. I don't know why this thought kept popping in my head but it did. Mind you I'm nit picking because I do think its a good film, with several moments that are pure electricity in a bottle (Cash singing Folsom Prison Blues for Sam Phillips for the first time is enough to make you fall out of your seat). When you can see this movie. it may not change your life but it will make you sing for a couple of hours.

Table 5.19. Blog post BLOG06-20051220-012-0010730076: A relevant blog post to query 1032 which is hurt the most on average by the proximity-based models compared to the non-proximity opinion retrieval. Here terms in red are query terms while underlined terms are opinion terms according to our opinion lexicon.

there is no statistically significant difference between them. However, we have seen better and more stable behavior from the Laplace kernel on different parameter values than other kernels. We therefore used the Laplace kernel in our model.

We proposed different models for aggregating the opinion score at different query positions in a document to estimate the document's opinion score. The proposed aggregation models were: *Maximum, Average, Relevance Weighted Average, Ordered Weighted Average* and *Average of Maximums*.We discussed the advantages and disadvantages of using each model. Based on probabilistic reasonings we showed that the proper way of combining relevance and opinion scores of documents is multiplying them together.

We evaluated our models on the standard BLOG06 collection and compared the performance of our proposed proximity-based models to the best TREC runs and state-of-the-art methods. We showed significant improvements across all previously proposed methods across all TREC baselines. We showed that all our proposed proximity-based models were able to improve the opinion retrieval performance of the TREC relevance baselines. They were also able to improve the non-proximity opinion retrieval system which used the same opinion lexicon over the same initial relevance ranking. These results were very encouraging showing that using simple proximity information in a proper way can be very helpful.

We also analyzed the effect of normalizing the relevance score before applying it in the model. Our results show that normalization can be important, and that the best normalization strategy is dependent on the underling relevance retrieval baseline.

Analysis in this chapter and Chapters 4 revealed the importance of the combination step in an opinion retrieval system. In the combination step the goal is to rank documents based on their scores in the two criteria of topical relevance and opinionatedness. In the next chapter we will look into the methods of combining the scores of multiple criteria in IR in more details and propose methods for better handling the score combination step.

Chapter 6

Score Transformation for Multi-Criteria Relevance Ranking

6.1 Introduction

Many IR tasks such as document retrieval and opinion retrieval require ranking documents based on a combination of multiple criteria; for example, in document retrieval, we may want to consider multiple relevance criteria such as topicality relevance, popularity, credibility or trustworthiness. We can also consider blog post opinion retrieval as a multi-criteria IR problem in which the goal is to rank documents according to the degree that they express opinion about a particular topic. Therefore, the two criteria involved in this ranking problem are topical relevance and the amount of opinion that a document expresses about the query. To produce such a multi-criteria ranking, we first need to score documents in each criterion and then combine these scores to generate a final ranking score for each document.

In this chapter we consider opinion retrieval as a multi-criteria IR problem and look into the linear combination which is the dominant approach for combining the scores of multiple criteria in the literature. We discuss that the linear combination of raw scores is not necessarily the optimum way of combining them due to incompatibility of scores of different criteria. We then propose a general approach to transforming scores before combining them such that the transformed scores become more compatible and their linear combination become more optimum. The proposed approach is not limited to opinion retrieval and is applicable to different multi-criteria IR tasks. However, we evaluate it for the task of opinion retrieval. In the previous chapter we followed a probabilistic model and showed that the proper way of combining relevance and opinion scores is multiplying them together. Multiplication can be considered a linear combination in the *Log* space. Therefore, *Log* seems to be the optimum transformation for the specific task of opinion retrieval. We treat *Log* as an approximation of *upper bound* and compare it with the transformation produced by our approach thoroughly.

The remainder of this chapter is structured as follows. Section 6.2 explaines the problem of incompatibility between scores of different criteria. In Section 6.3, we analyze a linear combination function and derive a general desirable constraint on the transformation functions. Based on the constraint, we propose a two stage approach to score transformation, in Section 6.4. We explain our experimental design in Section 6.5. We then present our experimental results in Section 6.6. Finally, conclusions are given in Section 6.7.

6.2 Score Incompatibility in Linear Combination Model

Linear combination of different criteria scores has so far been the dominant approach due to its simplicity and effectiveness [80, 5]. However, when the number of criterion aspects and consequently the number of criteria aspects scores increases, estimating the parameters of the linear combination is difficult. Therefore, recently, many Learning to Rank systems have been proposed to generate the final score of documents based on multiple features and aid us in estimating parameters properly. Most of the proposed and widely used Learning to Rank systems are still based on the linear combination of features [109, 113]. Linear combination model requires that the scores to be combined are "comparable" to each other, an assumption that generally does not hold due to the different ways of scoring each relevance criterion. Although it has been suggested to carry out query-based feature normalization before combination [54], it was mostly to make features comparable with respect to the range of values across different queries and features and no previous work has inspected the compatibility of scores carefully. To illustrate the complexity of the issue of score compatibility, we plot two conditional probability curves in Figure 6.1, each representing the probability of relevance given the value of a score in a different dimension of relevance criteria. Suppose S_i and S_j are two criterion scores. Comparing the influence of individual criterion score on the probability of relevance reveals that the two scores are not comparable in terms of the rate of change in the probability of relevance. It is obvious from Figure 6.1 that in low-score region, S_j should have higher coefficient since for the same score, S_j has a higher probability of relevance than S_i, whereas in the high score region, it should be the opposite for

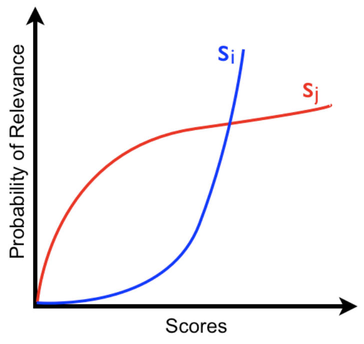

Figure 6.1. Comparison of the rate of probability of relevance change for two criteria scores.

the same reason (i.e., S_i should have a higher coefficient). This indicates that with fixed coefficients we will never be able to find the optimal combination of these scores. Intuitively, the problem is caused by the uneven growth rate of the probability of relevance with respect to different dimensions of relevance. The only way to get out of this non-optimality is to transform scores before combining them linearly.

6.3 Problem Formulation

In multi-criteria IR, the general form of scoring documents based on a linear combination model is given by $S(d) = f(\alpha_0 + \sum_{k=1}^{K} \alpha_k S_k(d))$, where $S_1, S_2, ..., S_k$ are the scoring functions related to K criteria, $\alpha_0, ..., \alpha_k \in \Re$ are the coefficients and f is a monotonically increasing function which can be linear or any non-linear function such as Logistic function. This form of combination, assumes that the scores to be combined (i.e. S_i) are compatible. However as discussed in Section 6.2, if such assumption does not hold, the combination would not be optimal and inevitably leads to non-optimal results.

The main question we study in this chapter is how to find a transformation function g_i for each score S_i such that any pair of transformed scores, would be more compatible and their linear combination be more likely optimal. Formally, let $S_i'(d) = g_i(Si(d))$ be the transformed score for document d. The linear combination of transformed scores would have the following form:

$$S(d) = f(\sum_{i=1}^{k} \alpha' S_i'(d)) \tag{6.1}$$

We suppose $S'_i(d)$'s would be more comparable than the original $S_i(d)$'s. Thus, our problem is to find the proper function g_i for every criterion score S_i. Clearly, a main challenge is how to quantify the notion of "comparability". More specifically, the question is how can we formally capture the problem of non-comparability illustrated in Figure 6.1. To answer this question, we look into desirable properties that should be satisfied by score transformations to achieve the optimal score through linear combination. One of these properties can be explained by examining the rate of change in the optimum score (i. e. probability of multi-criteria relevance) with respect to each score criterion. Thus, we look into the partial derivative of the multi-criteria probability of relevance with respect to each score criterion. Let S be the optimum multi-criteria score, if the optimum score is indeed satisfied by the linear combination of criteria scores, we should have the following partial derivative of S with respect to S'_i:

$$\frac{\partial S}{\partial S'_i} = \frac{\partial f}{\partial \sum_{k=1}^{K} \alpha'_k S'_k(d)} \frac{\partial \sum_{k=1}^{K} \alpha'_k S'_k(d)}{\partial S'_i} = \frac{\partial f}{\partial \sum_{k=1}^{K} \alpha'_k S'_k(d)} \alpha'_i \tag{6.2}$$

Although the obtained partial derivative does not show any desirable property by itself, looking into the ratio of the partial derivatives with respect to two different criteria functions leads to an interesting constraint which we call the *Compatibility Constraint*.

Compatibility Constraint:

If the optimum score can be modeled by the linear combination of criteria scores, using Equation (6.1), the ratio of derivatives of the optimum score with respect to any two score criteria functions, should be constant:

$$\frac{\frac{\partial S}{\partial S'_i}}{\frac{\partial S}{\partial S'_j}} = \frac{\alpha'_i}{\alpha'_j} = c_{i,j},\ c_{i,j} \textit{ is constant} \tag{6.3}$$

Intuitively, this constraint requires that the growth of the optimum scoring function due to the growth of a score in each dimension to be "synchronized" through a constant ratio. One way to satisfy the above constraint is to find function g_i that maximizes the correlation between the transformation of $S_i(d)$ and the optimal score of d.

$$g_i^* = \arg\max_g \rho(g_j(S_j(d)), S(d)) \tag{6.4}$$

where ρ is the *pearson correlation coefficient*. Clearly, if $\forall i \quad \rho(g_i(S_i(d)), S(d)) = 1$, then the compatibility constraint holds.

Notice that maximizing the correlation coefficient, is a stronger requirement than the compatibility. In this sense, solving this optimization problem to find a transformation is a conservative strategy for satisfying the constraint. In this conservative approach, the partial derivative of the scoring function w.r.t. each criterion score would be a constant, whereas in general, this does not have to be a constant (even though their ratios are). Equation (6.4) can only be solved if we have available a sample of both the scores in all the aspects of relevance and the “ideal" score $S(d)$ which can be defined as the probability of relevance of document d. Such a (training) sample can be constructed based on a set of past queries and their relevance judgments. In the next section, we present a two-stage approach to learning optimal transformation functions from such a training data.

6.4 A Two-Stage Approach to Score Transformation

In this section we present a general two-stage approach to solving the optimization problem defined in the previous section. In the first stage, we would use the ACE algorithm, to learn a non-parametric optimal transformation of $S_j(d)$ based on the training data; in the second, we would apply BoxCox to obtain a parametric power transformation that can approximate the non-parametric transformation learned using ACE. Below we first give a general introduction to ACE and then discuss each stage in detail.

6.4.1 Obtaining non-parametric optimal transformation using ACE

Alternating Conditional Expectation

Alternating Conditional Expectation (ACE) [8], is a nonparametric model for finding the optimal transformations of the response and predictor variables such that the transformed variables have linear relationship. One important advantage of ACE is that it does not make any parametric assumption about the transformations, thus it can potentially work for any data to uncover the desired transformations.

Assume we have the response variable Y and the set of predictor variables $X_1, ..., X_p$. ACE finds the optimal transformation of the response variable as $\theta(Y)$

and the set of predictor variables as $\phi_1(X_1), ..., \phi_p(X_p)$ such that:

$$\theta(Y) = \alpha + \sum_{i=1}^{p} \phi_i(X_i) + \epsilon \tag{6.5}$$

ACE starts with arbitrary transformation functions $\theta(Y)$, $\phi_1(X_1), ..., \phi_p(X_p)$ which have zero mean and iterates through Equations (6.6) and (6.7) until ϵ^2 no further decreases. ϵ^2 is the error variance that is not explained by a regression of the $\theta(Y)$ on $\sum_{i=1}^{p} \phi_i(X_i)$ and is estimated by : $\epsilon^2 = E[\theta(Y) - \sum_{i=1}^{p} \phi_i(X_i)])$.

$$\phi(X_i) = E[\theta(Y) - \sum_{j \neq i}^{p} \phi_i(X_j) | X_i]) \tag{6.6}$$

$$\theta(Y) = \frac{E[\sum_{i=1}^{p} \phi_i(X_i) | Y]}{\left\| E[\sum_{i=1}^{p} \phi_i(X_i) | Y] \right\|} \tag{6.7}$$

While it is possible to apply ACE to obtain a nonlinear transformation simultaneously for multiple variables, as a preliminary exploration, in this chapter, we apply ACE to each predictor variable separately. Thus in our problem setting, the number of predictor variables is one (i. e. $p = 1$) and for each criterion S_k, we have $Y = p(R|S_k)$ and $X_1 = S_k$, where $p(R|S_k)$ indicates the probability of multi-criteria relevance in score criterion S_k. We use ACE to find the optimum transformations $\theta(p(R|S_k))$ and $\phi_1(S_k)$ such that equation (6.5) holds. For more details about ACE refer to [8, 105].

Training Set Construction

To apply ACE in the context of score transformation, we need to construct a training set with the probability of relevance given the scores in each criterion dimension. To calculate this probability, theoretically we should calculate the proportion of relevant documents in the set of documents with a specific score. However, as the number of documents with exactly the same score is very small in practice, we use a bin strategy as follows: First, we use *MinMax* method to normalize the scores to the same range across queries. Then, we sort documents based on their criterion score increasingly and consider every n consequent documents to belong to the same bin. We use the median of scores in each bin as its "identifier score" S_i, and estimate the conditional probability of multi-criteria relevance, $P(R|S_i)$, by dividing the number of relevant documents by the total number of documents in bin_i.

In Figures 6.2 and 6.3 , we show some sample results of applying ACE on transforming the topical relevance and opinion scores. Comparison between

Figures 6.2(a) and 6.2(c) for topical relevance score shows that after ACE transformation, we can indeed obtain high linear correlations between the probability of relevance and score. The same happens in case of transforming the opinion scores.

6.4.2 Obtaining Power Transformation functions using BoxCox

Unfortunately, the output from ACE doesn't give us the functional form of the transformation. This does not prevent us from applying the learned transformation to new test instances, though, as we can compute the transformation of a new instance using that of its neighboring scores in the training set (we use *ACE* to refer to this single-stage method). However, it may be desirable to have a parametric transformation function to reduce the chance of overfitting. Indeed, if the training data is sparse, the transformation obtained from ACE may overfit the training data, especially when the probability distribution of relevance of training queries is different from that of a test query. By further obtaining a simpler monotonic parametric transformation function, we can control the complexity of the transformation and thus reduce the chance of overfitting. Thus, we follow the suggestions reported in [105] and propose to further estimate the parametric form of ACE transformation using the BoxCox method [7]. We use *ACE-BC* to refer to this two-stage model. We will show later, with experiments, that ACE-BC indeed performs better than ACE, though the difference is not statistically significant.

BoxCox method

BoxCox [7] is a parametric technique that can be used to estimate continuous and monotonic transformations through a family of power transformation. Power transformation can be defined as follows:

$$t_\lambda(y) = \begin{cases} y^\lambda, & \text{if } \lambda \neq 0 \\ log(y), & \text{if } \lambda = 0 \end{cases} \tag{6.8}$$

BoxCox method finds the best power transformation of the response variable Y as $t_\lambda(Y)$ in a linear regression. Box and Cox proposed maximum likelihood and Bayesian analysis for the estimation of λ parameter.

Estimating Functional Form of ACE transformations

In order to find an approximation to the ACE outputs, we consider the original data Y or X as the response and ACE output $\theta(Y)$ or $\phi(X_i)$ as the predictor. We

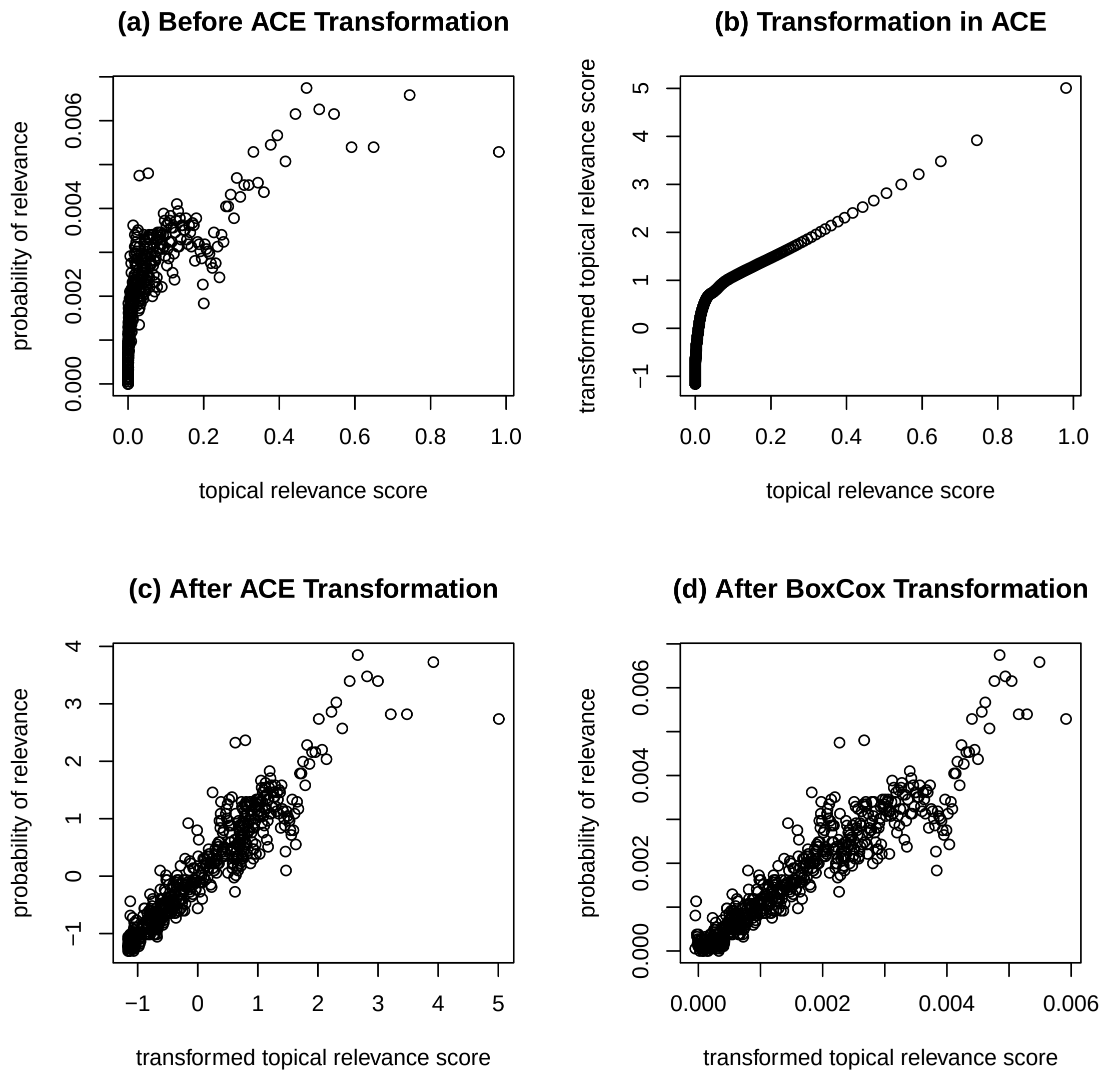

Figure 6.2. The effect of topical relevance score transformations by ACE and BoxCox methods on linearizing the relationship between probability of relevance and score criteria.

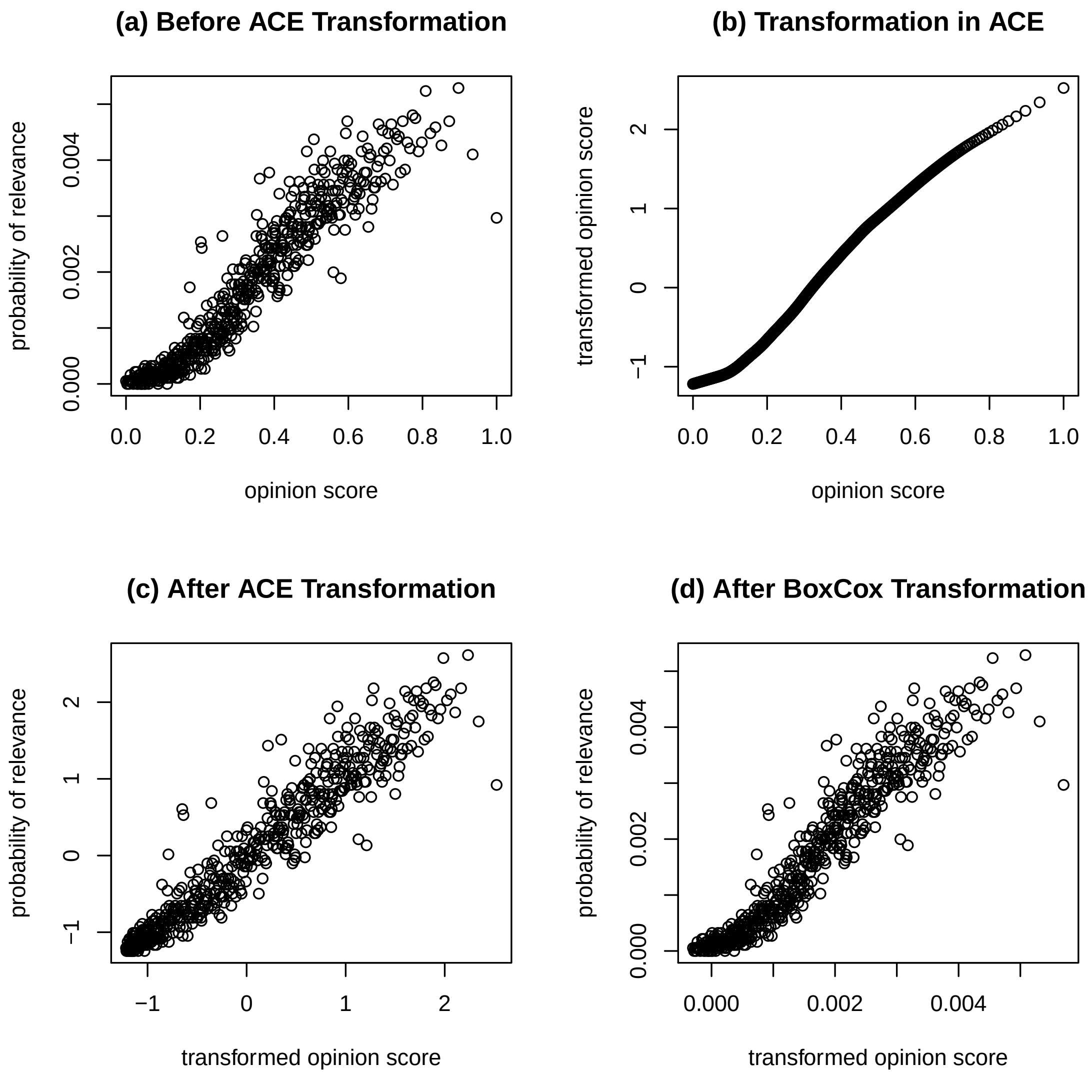

Figure 6.3. The effect of opinion score transformations by ACE and BoxCox methods on linearizing the relationship between probability of relevance and score criteria.

then use BoxCox method to find the maximum likelihood parameter λ_y and λ_x such that:

$$t_{\lambda_y}(p(R|S_i)) = \alpha_0 + \alpha_1\theta(p(R|S_i)) + \epsilon$$

$$t_{\lambda_x}(S_i) = \alpha_0 + \alpha_1\phi(S_i) + \epsilon$$

Since we are interested to find transformation for S_i, not for $p(R|S_i)$, we can force ACE in the previous step, to transform the response variable linearly. Therefore λ_y would be equal to 1. Note that, in case of a single-stage model *ACE*, we should put a constraint on ACE enforcing the monotonic transformation of scores. But in case of using the full ACE-BC model, the non-monotonicity of the ACE transformation can be fixed by the BoxCox approximations which are always monotonic. Figures 6.2(d) and 6.3(d) show the plot of probability of relevance versus BoxCox transformed relevance and opinion scores respectively. Comparison between Figures 6.2(c) and 6.2(d) and also Figures 6.3(c) and 6.3(d) show that BoxCox transformations are indeed good estimates for ACE transformations.

Finally, the parametric function $g_i(S_i(d))$ is approximated as follows: $g_i(S_i(d)) = b_0 + b_1 \times t_{\lambda_x}(S_i)$, where b_0 and b_1 can be estimated through linear regression over $t_{\lambda_x}(S_1)$ and $p(R|S_i)$.

6.4.3 Summary

We can summarize the *ACE-BC* procedure of score transformation for every score criterion S_i in the following 6 steps:

1. Create bins for the score criterion based on training data and assign the median score in each bin as its identifier, S_b.

2. Estimate $P(R|S_b)$ based on the proportion of relevant documents in bin S_b.

3. Apply ACE to learn non-parametric transformations of scores.

4. Apply BoxCox to estimate an approximate parametric function $g_i(S_i(d))$ to the ACE transformation.

5. Use $g_i(S_i(d))$ to transform every document score for criterion score S_i.

6. Finally, plug in the transformation functions g_i to any existing linear combination method.

Note that the single-stage *ACE* model stops at step 3 and uses the learnt non-parametric model to estimate the transformed value of a new score before linear combination.

6.5 Experimental Design

Opinion Retrieval Task: For topicality relevance, we use the Language Modeling approach with Dirichlet prior [115]. For the opinion aspect, we use the *proximity-based* opinion scoring method. We consider the topicality relevance score and opinionatedness score as the two criteria scores and compare different models of transformation before linear combination of them to produce final ranking.

Measures: We report the MAP as well as P@10, P@30 and P@100 in terms of opinion finding. We report the statistical test result of non-directional paired t-test at significance level 0.01.

Final ranking function and parameter tuning: The focus of our work is on learning score transformation functions in multi-criteria relevance ranking. The learned transformations are meant to be applicable to any linear combination-based final ranking function. Here, since we have only two dimensions and thus essentially only one parameter to tune, we manually searched for the optimal value of the interpolation coefficient. In case of having more criteria, any linear learning to rank method can be used for the combination of criteria's scores.

Formally, let g_t and g_o be the two transformation functions learned using our method for topical relevance and opinion scoring, respectively. Our final scoring function is:

$$Score(Q,D) = \alpha \times g_t(p(Q|D)) + (1-\alpha) \times g_o(p(O|Q,D)). \tag{6.9}$$

We tried different values of $\alpha \in (0,1)$ with step 0.01 and chose the best value based on the performance on the training set of queries. This parameter tuning was done for all results reported in this chapter.

6.6 Experimental Results

In this section we explain the experiments we conducted in order to evaluate our proposed method. We first check if the necessity of compatibility between different aspect scores in linear combination is supported by the experimental results in real application. Therefore, we check the combination coefficient, α, learnt on the training set of queries, in case of using *MinMax* normalization. Results showed that the final combination relies just on relevance score to produce the final ranking (i.e. $\alpha = 1$). This result may lead to the conclusion that the applied opinion scoring method is not effective. However, we will show later that if we use an appropriate transformation before the linear combination we are able to gain statistical significant improvements over topical relevance baseline.

Table 6.1 reports the results of our proposed methods *ACE* and *ACE-BC* along with some of the normalization methods, discussed in Section 3.3, on opinion retrieval problem.

Method	MAP	P@10	P@30	P@100
MinMax	0.2929	0.4800	0.4207	0.3448
Z-score	0.2913	0.4840	0.4373†•	0.3604†
Sum	0.3090†‡	0.5560†‡•	0.4927†‡•	0.3844†‡•
LogReg	0.2956	0.4820	0.4200	0.3522
SHIS	0.2996	0.5420	0.4660	0.3714
ACE	0.3307†‡∗•	0.5660†‡•	**0.5127**†‡•	0.4054†‡∗•⋄
ACE-BC	**0.3332**†‡∗•⋄	**0.5820**†‡•	0.5107†‡•	**0.4064**†‡∗•⋄

Table 6.1. Comparison of different score normalization for opinion retrieval. Symbols †, ‡, ∗, • and ⋄ indicate statistically significant improvements over MinMax, Z-score, Sum, Logistic Regression and SHIS method respectively.

Table 6.1 shows that our proposed ACE-based methods help us in exploiting the opinion scores of documents and improving the retrieval performance. The performance of the ACE-based methods is statistically significant higher than all normalization methods in almost all measures. The most effective normalization baseline appears to be *Sum*, however the ACE-based methods are still more effective. *LogReg* in Table 6.1 indicates using Logistic Regression[1] to transform every criterion score to the probability of relevance. It is similar to our proposed methods in the sense that it also refers to the probability of relevance. However, the parametric function of logistic regression may be too restrictive.

[1]using LingPipe: http://alias-i.com/lingpipe/

This, coupled with the sparseness of training data, may explain the poor performance of *LogReg*. This also indicates the importance of not overfitting the training data to optimize probability of relevance; instead, the goal should be to learn a relatively generalizable non-linear transformation function with good linear correlation with probability of relevance.

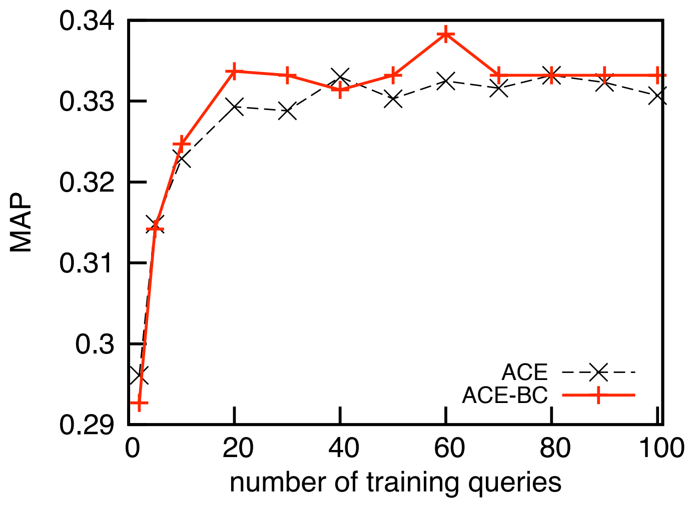

Figure 6.4. Comparing the performance of ACE and ACE-BC over different number of training queries

Comparing the performance of the two ACE-based models shows that *ACE-BC* is slightly more effective than *ACE*, but the difference is not statistically significant. To further investigate the effect of having BoxCox stage on the model, we tried different number of queries in the training stage of the ACE. Figure 6.4 shows the performance of *ACE* and *ACE-BC* in terms of MAP. Results show that, on average, estimating the parametric form of transformation using Box-Cox is beneficial, suggesting that the parametric model may be less sensitive to the training size, thus helping to avoid overfitting.

Table 6.2 shows the transformation functions learned thorough *ACE-BC* for raw scores as well as different perturbations.

We also check the sensitivity of the method on the bin size which is used to estimate probability of relevance. We tried different bin size values of 100, 300, 500, 1000 and 1500. Figure 6.5 shows the plot of MAP versus different values of bin size. As we can see from the Figure, the method is not so sensitive w.r.t. the bin size and in all cases, we can outperform the baselines.

Input Score S_i	Transformation Function $g(S_i)$
S_r	$0.0002 + 0.006S_i^{0.3}$
S_o	$-0.0003 + 0.006S_i$
S_r^5	$0.003 + 0.00005log(S_i)$
S_o^5	$-0.0004 + 0.006S_i^{0.2}$
e^{S_r}	$0.004 - 0.003S_i^{-21.5}$
e^{S_o}	$0.02 - 0.024S_i^{-0.3}$
$e^{S_r}S_r^5$	$0.004 - 4 \cdot 10^{11}S_i^{-24.8}$
$e^{S_o}S_o^5$	$0.005 - 0.006S_i^{-2.3}$

Table 6.2. Transformation functions learned through the ACE-BoxCox model.

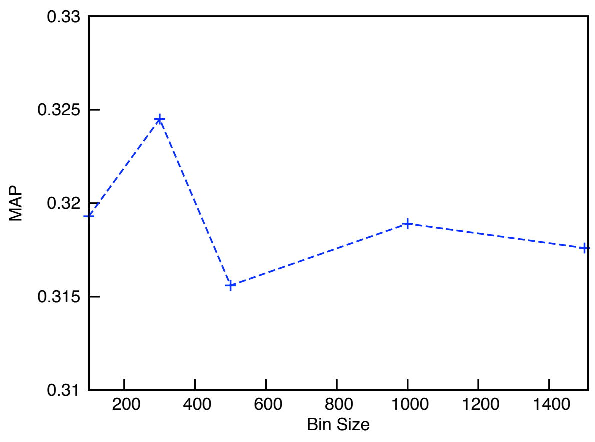

Figure 6.5. Sensitivity to the bin size.

6.6.1 Log as an Approximation to Upper Bound:

Although we are evaluating the proposed method in the context of opinion retrieval, we are not using any property of the task or scoring functions. This indicates the generality of the model. However, it is still interesting to see how close are our learnt transformations to the upper bound of the optimal transformations for the specific task of opinion retrieval. In previous chapter we proposed a probabilistic model which considers the multiplication of relevance and opinion scores as the proper way of combining them. Multiplication can be considered a linear combination in the *Log* space. Therefore, *Log* seems to be the optimum transformation for this specific task. We treat *Log* as an approximation of *upper bound* and compare *ACE-BC* with log thoroughly. Table 6.3 presents the result of using *Log* or *ACE-BC* transformations on original scores and their different perturbations. Results show that *ACE-BC* is fairly close to this carefully crafted pseudo upper bound, and *ACE-BC* is much more robust as we perturb the scores. Perturbation is common in retrieval systems. For example, when deriving a retrieval function, we often take a log transformation to preserve precision. We also often ignore any irrelevant constant that doesn't affect document ranking, etc. Thus in real applications, we often cannot assume a particular fixed distribution of scores in any dimension.

score perturbation	Metric	Log	ACE-BC
original score	MAP	**0.3384**	0.3332
	P@10	**0.5840**	0.5820
	P@30	**0.5200**	0.5107
	P@100	0.4054	**0.4064**
$\alpha S_r^5 + \beta S_o^5$	MAP	**0.3384**	0.3368
	P@10	0.5840	**0.5880**
	P@30	0.5200	**0.5227**
	P@100	0.4054	**0.4078**
$\alpha e^{S_r} + \beta e^{S_o}$	MAP	0.2929	**0.3176** $*$
	P@10	0.4800	**0.5860** $*$
	P@30	0.4207	**0.5027** $*$
	P@100	0.3448	**0.3884** $*$
$\alpha e^{S_r} S_r^5 + \beta e^{S_o} S_o^5$	MAP	0.2946	**0.3171** $*$
	P@10	0.4760	**0.5840** $*$
	P@30	0.4167	**0.5013** $*$
	P@100	0.3490	**0.3886** $*$
$\alpha e^{S_r} + \beta S_o$	MAP	0.2771	**0.3166** $*$
	P@10	0.4820	**0.5820** $*$
	P@30	0.4300	**0.5007** $*$
	P@100	0.3362	**0.3892** $*$
$\alpha S_r + \beta e^{S_o}$	MAP	0.3314	**0.3343**
	P@10	0.5820	**0.5860**
	P@30	0.5093	**0.5133**
	P@100	**0.4052**	0.4024

Table 6.3. Comparison of the robustness of ACE and Log w.r.t different score perturbations. The symbol $*$ indicates that improvement is statistically significant compare to the alternative transformation.

We also plot the *precision-recall curve* of ACE-BC and Log transformations in Figure 6.6. As we can see from the plot, ACE-BC is indeed a good approximation.

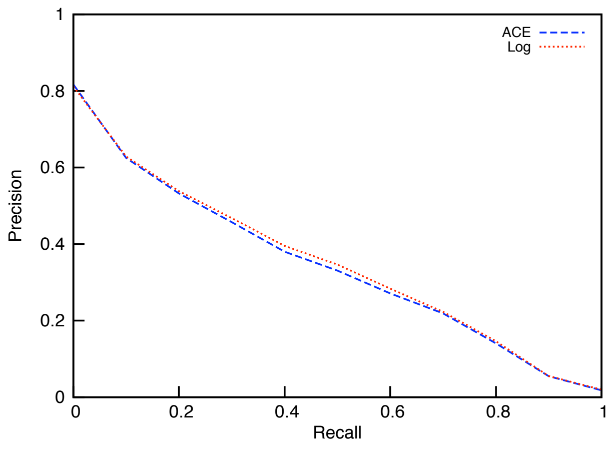

Figure 6.6. Precision-Recall curve of ACE-BC and Log transformation over original scores.

We interpret these results as very promising because our method does not use any specific knowledge about this task. This gives us confidence in generalizing the positive results of the proposed method to other domains and applications.

6.7 Conclusions

In this chapter we analyzed the incompatibility problem of scores in multi-criteria IR and proposed the necessity of scores transformation before applying the linear combination. We proposed a general and principled approach to score transformation based on the Alternating Conditional Expectation and BoxCox model. The proposed approach is general and can be used for score transformation of different criteria for all Multi-criteria IR problems. In this chapter, we evaluated the proposed method in the context of opinion retrieval and showed its effectiveness. We compared the proposed approach with the state of the art score normalization techniques and showed the superiority of our method. Although the proposed method is general, in this chapter, we used opinion retrieval task to evaluate the method.

From the previous chapter we learnt that the multiplication of relevance and opinion score, which is linear combination in the log space, is the proper way of combining the two scores. Therefore, we compared the transformation learnt using our general approach with *Log* transformation as an upper bound. We showed that the performance of the ranking produced using the ACE-based transformed scores is very close to the one produced using *Log*. We also showed that the proposed approach can still find the proper transformation after perturbing the original scores, where *Log* is not necessarily the best transformation.

Although the score incompatibility problem exists for linear combination in all multi-criteria IR problems and the proposed score transformation is a promising solution, in this chapter we only used the opinion retrieval problem to show the validity of the proposed methods. In future, we plan to apply the proposed method on other multi-criteria IR tasks and experimentally evaluate the generality of the proposed method. We would also like to use the proposed score transformation model to normalize feature values before applying Learning to Rank methods. We believe the transformation learnt through this method can help exploiting features and learn better ranking models.

Chapter 7

Conclusions

The main motivation for this thesis was to propose retrieval models that facilitate access to the highly opinionated contents of blogs. We focused on the specific task of *blog post opinion retrieval* which aims at finding blog posts that express an opinion about a topic. We considered blog post opinion retrieval task as a multi-criteria IR problem where the two criteria involved are topical relevance and opinionatedness toward the query topic. We used standard methods for relevance and proposed different methods for scoring blog posts based on their opinionatedness toward a query. We also investigated different ways of combining topical relevance and opinion scores to produce a final ranking. We showed that our proposed methods improve opinion retrieval performance over state-of-the-art methods.

In Section 7.1 we revisit our research questions and provide answers to each of them. In Section 7.2 we list the future research directions following from this thesis.

7.1 Answers to Research Questions

At the time we started our research a lot of different opinion retrieval approaches were proposed that make use of the internal (inside collection) or external information in different ways for different components of an opinion retrieval system. However, it was not clear from those studies which method in which component of the system was responsible for the performance gain demonstrated. Therefore, we started our research by limiting ourselves to use only data available in the TREC blog collection and asked:

RQ1 How can we make use of the information available in the collection to improve the opinion retrieval performance? In such settings, are learning methods more effective than non-learning methods in handling the different steps of an opinion retrieval system?

In Chapter 4 we used the relevance judgments of the documents in the Blog06 collection and investigated different models of weighting terms based on the statistics of their occurrence in relevant and opinionated versus non-relevant documents. Based on terms' weights, we selected terms as features for learning an opinion classifier. We also used weighted terms to create an opinion lexicon. We then tried simple learning and lexicon-based methods for assigning opinion scores to documents. Finally we tried different methods for combining relevance and opinion scores to produce a final ranking. We evaluated different models experimentally and compared the performance with the topical relevance retrieval baseline.

Results showed that the best term weighting methods are mutual information and weighted log likelihood ratio. We did not notice any statistical significant difference between the two methods. We also did not notice a statistical significantly difference between the learning-based and lexicon-based methods of opinion scoring. However, unlike the lexicon-based approach, the learning-based methods showed statistical significantly improvement over the topical relevance retrieval baseline. Finally, comparison between different methods of combining the relevance and opinion scores showed that SVMmap, as a learning to rank method is the only combination method that leads to improvement over the topical relevance baseline [27].

Based on the results of Chapter 4 we learnt that the combination step is very important such that, even with a good opinion scoring method, no improvement could be gained over topical relevance retrieval system without a proper combination method. We also realized the necessity of using a more precise opinion scoring method to improve the opinion retrieval performance. The opinion scoring methods in Chapter 4 were all based on the assumption that every opinion expressed in a relevant document to a topic is referring to the topic. Therefore, the opinion scoring component was only responsible for finding opinion terms in a document and scoring the document based on the frequency of opinion terms in it. In Chapter 5, we explored the validity of this assumption and asked:

RQ2 Assuming a document to be relevant to the user's information need, can we consider all opinion expressions occurring in the document as targeted to the topic of the query? If no, how can we identify the relevant opinion expressions to the topic of the query?

In chapter 5, we considered the proximity of opinion terms to the query terms as an indicator that the opinion term is referring to the query. We proposed using kernel functions to model the proximity of opinion terms to the query terms. We considered some previously proposed kernel functions and also proposed a new kernel function named Laplace kernel. We proposed a method for estimating the opinion density at query term positions. We then proposed different methods for aggregating the opinion density at different query terms positions in a document to estimate an overall opinion score for the document. The proposed aggregation models were: *Maximum, Average, Relevance Weighted Average, Ordered Weighted Average* and *Average of Maximums*.

Based on probabilistic reasonings we showed that the proper way of combining relevance and opinion scores is multiplying them together.

To evaluate the effect of using proximity information, we considered a simpler model, called the non-proximity opinion retrieval model, where the opinion score of a document is calculated without taking into account the proximity of opinion terms to the query terms. In the non-proximity opinion retrieval model, we used the same opinion lexicon, same topical relevance retrieval model and the same combination method as the proximity-based model for producing the final ranking. Note that the non-proximity based model has the same assumption as the opinion scoring methods of Chapter 4, that is all opinion terms in a relevant document to a query are about the query. The comparison between the proximity-based and non-proximity opinion retrieval models showed that the proximity information helps in improving the performance of an opinion retrieval system. The comparison between different kernel functions for modeling the proximity showed that there is no statistically significant difference between the kernels when the best bandwidth parameter is used for each. However, the Laplace kernel was the most robust kernel across all range of the bandwidth parameter.

We discussed the advantages and disadvantages and assessed the impact of each aggregation method on the performance of the proximity-based model. We found that all resulted proximity-based models were able to improve the opinion retrieval performance of the TREC topical relevance baselines and the non-proximity based opinion retrieval model.

We also argued that normalizing the relevance scores before combining with opinion scores is very important. We showed that the final performance is influenced by the normalization method and the best normalization strategy is dependent on the underling relevance retrieval baseline.

Comparison with the best TREC runs and state-of-the-art methods showed that the proximity-based model leads to significant improvements across state-of-the-art score normalization methods across all TREC baselines. These results were very encouraging showing that using simple proximity information in a proper way can be very helpful [28, 29].

Analysis in Chapters 4 and 5 revealed the importance of the combination step in an opinion retrieval system. In the combination step the goal is to rank documents based on their scores in the two criteria of topical relevance and opinionatedness. In Chapter 6 we looked into the methods of combining the scores of multiple criteria in IR and found that the linear combination is the common approach. We investigated the necessity of score transformation before linear combination and asked:

RQ3 What is an optimal way of transforming scores of different criteria before linear combination to increase their compatibility?

We analyzed the linear combination model and found that such a strategy of combination requires that the scores to be combined are “comparable” to each other, an assumption that generally does not hold due to the different ways of scoring each relevance criterion. We argued that it is necessary to transform the raw scores for different criteria appropriately to make them more comparable before combining them linearly. To this end, we proposed a new principled approach to score transformation in linear combination, called ACE-BC, in which we learn a separate non-linear transformation function for each relevance criterion based on the ACE algorithm and Box-Cox transformation.

We looked at the opinion retrieval problem as a multi-criteria IR problem with two criteria of topical relevance and opinion scores. We showed that by transforming the scores using the ACE-BC transformation, we can obtain much better results in linear combination compared to using the raw scores. We also compared the effect of ACE-BC transformation with the state-of-the-art score normalization techniques that are used to make the scores comparable in terms of range or distributions. We showed that the ACE-BC transformation is more effective and leads to statistically significant higher performance than all previously proposed normalization techniques.

We also noticed that multiplying the topical relevance and opinion scores, as suggested in Chapter 5, can be considered as a linear combination in Log space. Therefore, we compared the effect of ACE-BC transformation with the effect of Log transformation. We showed that the ACE-BC transformations leads to a ranking which is very close in terms of effectiveness to the one produced by the

Log transformation. We also showed that the ACE-BC transformation method is more robust than the Log transformation when using the perturbed scores rather than using the original raw scores [30].

7.2 Future Research Directions

In Chapter 5 of this thesis we proposed a proximity-based method for estimating the opinionatedness of a document about a query. We used a general opinion lexicon for finding opinion terms and relied on the proximity of an opinion term to a query term to estimate the probability that the opinion term is targeted at the topic of the query. We can now think of the following two directions to improve this method:

Topic-specific Opinion Lexicon: The set of terms that are usually used to express opinion about different topics are different. For example, we do not usually use the term "delicious" to express our opinion about an entity of electronic type or a device while we definitely use if to express opinion about an entity of food type. Having access to such topic-specific opinion lexicon enables the proximity-based model to ignore unrelated opinion terms even if they occur in close proximity to the query terms. The other advantage of such model is when an opinion term is used with objective sense (i.e. non-opinionated meaning) in a context. For example the term "delicious" can be used as a reference to the social bookmarking web service, delicous.com, rather than an opinionated term. If the opinion term is used in an objective sense in a context, it is usually not present in the opinion lexicon of the query. Therefore, the opinion retrieval model will ignore such an objective usage of the opinion term. To create a topic-specific opinion lexicon we should weight terms based on the probability that a term is expressing an opinion about the topic. We can then use topic-specific opinion lexicons in the proximity-based opinion retrieval model to further improve its performance.

Coreference Resolution: The proximity-based model scores documents based on the amount of opinion expressed around the exact query terms. However, writers sometimes use anaphoric expressions to refer to a query. For example, pronouns are usually used to refer to a query of type person. Also, terms such as *documentary*, *film* or *movie* are usually used to refer to a query of type movie.

More research is necessary to find such references to a query and extend the proximity-based model to consider such references in estimating the opinionatedness of a document toward the query.

In this thesis we focused on the opinion retrieval task. Although opinion retrieval is a useful task by itself, it can serve as a basic step for more complex tasks. Below, we briefly explain three examples of such tasks:

Polarity Detection: Finding opinionated content about a topic is useful, however knowing the polarity of the opinion, that is whether the opinion is positive or negative, is more interesting. A large body of research has been devoted to polarity detection in different domains such as movies or products reviews. However finding if a document's opinion is positive or negative about a topic, in blogs, is more challenging. The reason is that blogs are not usually specific to a domain or a topic. Therefore, it is necessary to first locate relevant opinions about a topic and then perform polarity detection over that. Also, using a topic-specific polarity lexicon plays a very important role here. The reason is that some terms can be used to express positive opinion about a topic while expressing negative opinion about another one. For example, the opinion term "long" is usually used as a positive term when referring to a battery life while it may reflect a negative meaning if it is referring to the distance between office and home. Ranking blog posts based on their positive or negative opinionatedness about a topic was a subtask of the opinion retrieval task in TREC 2007 to TREC 2008. The results of running this task in two years showed that polarity detection in blogs is difficult and more research is needed in this area.

Analysis of Opinion over Time: Looking at the changes in the amount and polarity of opinionated blog posts toward a topic over time is interesting for a wide variety of applications. For example, companies can monitor the public opinion about a product and analyze the reasons behind a shift in the polarity of opinion from positive to negative, if any, toward the product. Another possible application is predicting public's future opinion about a topic based on the past opinions about it.

Summarizing Opinion about a Topic: Presenting the list of all relevant and opinionated documents about a topic to a user is not very useful. The reason is that the user has to go through all documents to find the opinionated parts of document about the topic. In fact, a user for an opinion retrieval system is not looking for a single relevant document to its query but trying to find out different opinions by different people about different aspects of the topic. Therefore, it is more useful to gather all opinions about the topic of the query and present the user a summary of all the positive and negative opinions about different aspects of the topic of the query.

The above mentioned research directions are all related to the opinion retrieval. Another line of research which is motivated by Chapter 6 of this thesis is related to the score transformation before combining scores in multi-criteria IR. In Chapter 6 we proposed a general method for score transformation but we only used the opinion retrieval task to evaluate the method. One future research direction would be to apply the proposed method on other multi-criteria IR tasks and experimentally evaluate the generality of the proposed method. It is also interesting to use the proposed score transformation model to normalize feature values before applying Learning to Rank methods. We believe the transformation learnt through this method can help exploiting features and learn better ranking models.

Bibliography

[1] A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[2] G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. Automatic construction of an opinion-term vocabulary for ad hoc retrieval. In *Proceedings of the 30th European conference on IR Research*, ECIR'08, pages 89–100, 2008.

[3] A. Arampatzis and S. Robertson. Modeling score distributions in information retrieval. *Information Retrieval*, 14:26–46, 2011.

[4] R. Barquin. Sentiment analysis in government. *BeyeNETWORK*, July 18 2011.

[5] M. Bendersky, W. B. Croft, and Y. Diao. Quality-biased ranking of web documents. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 95–104, 2011.

[6] B. Bickart and R. M. Schindler. Internet forums as influential sources of consumer information. *Journal of Interactive Marketing*, 15(3):31–40, 2001.

[7] G. E. P. Box and D. R. Cox. An Analysis of Transformations. *Journal of the Royal Statistical Society*, B(26):211–252, 1964.

[8] L. Breiman and J. H. Friedman. Estimating Optimal Transformations for Multiple Regression and Correlation. *American Statistical Association*, 80(391), 1964.

[9] C. Buckley and E. M. Voorhees. Evaluating evaluation measure stability. In *Proceedings of the 23rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, pages 33–40, 2000.

[10] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 25–32, 2004.

[11] B. Carterette, J. Allan, and R. Sitaraman. Minimal test collections for retrieval evaluation. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 268–275, 2006.

[12] C. W. Cleverdon, J. Mills, and M. Keen. Factors determining the performance of indexing systems. *Volume I - Design, Volume II - Test Results, ASLIB Cranfield Project*, 1966.

[13] C. Costa Pereira, M. Dragoni, and G. Pasi. Multidimensional relevance: A new aggregation criterion. In *Proceedings of the 31th European Conference on IR Research*, ECIR '09, pages 264–275, 2009.

[14] N. Craswell, S. Robertson, H. Zaragoza, and M. Taylor. Relevance weighting for query independent evidence. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 416–423, 2005.

[15] F. Crestani, M. Lalmas, C. J. Van Rijsbergen, and I. Campbell. "is this document relevant?... probably": a survey of probabilistic models in information retrieval. *ACM Computing Survey*, 30(4):528–552, 1998.

[16] B. Croft, D. Metzler, and T. Strohman. *Search Engines: Information Retrieval in Practice*. Addison-Wesley Publishing Company, USA, 1st edition, 2009.

[17] K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 519–528. ACM Press, 2003.

[18] M. Efron. Cultural orientation: Classifying subjective documents by cociation [sic] analysis. In *Proceedings of the AAAI Fall Symposium on Style and Meaning in Language, Art, Music, and Design*, pages 41–48, 2004.

[19] K. Eguchi and V. Lavrenko. Sentiment retrieval using generative models. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 345–354, 2006.

[20] TREC tracks. http://trec.nist.gov/tracks.html, last accessed 20-April-2012.

[21] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation*, LREC ’06, pages 417–422, 2006.

[22] M. Fernández, D. Vallet, and P. Castells. Probabilistic score normalization for rank aggregation. In *Proceedings of the 28th European Conference on IR Research*, ECIR ’06, pages 553–556, 2006.

[23] M. Fernández, D. Vallet, and P. Castells. Using historical data to enhance rank aggregation. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’06, pages 643–644, 2006.

[24] E. Fox. Characteristics of two new experimental collections in computer and information science containing textual and bibliographic concepts. Technical report, Cornell University: Computing Science Department, 1983.

[25] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[26] M. Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING ’04, 2004.

[27] S. Gerani, M. J. Carman, and F. Crestani. Investigating learning approaches for blog post opinion retrieval. In *Proceedings of the 31th European Conference on IR Research*, ECIR ’09, pages 313–324, 2009.

[28] S. Gerani, M. J. Carman, and F. Crestani. Proximity-based opinion retrieval. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’10, pages 403–410, 2010.

[29] S. Gerani, M. Keikha, and F. Crestani. Aggregating multiple opinion evidence in proximity-based opinion retrieval. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’11, pages 1199–1200, 2011.

[30] S. Gerani, C. Zhai, and F. Crestani. Score transformation in linear combination for multi-criteria relevance ranking. In *Proceedings of the 34th European Conference on IR Research*, ECIR '12, pages 256–267, 2012.

[31] A. B. Goldberg, X. Zhu, and S. Wright. Dissimilarity in graph-based semi-supervised classification. *Journal of Machine Learning Research - Proceedings Track*, 2:155–162, 2007.

[32] D. Harman. Overview of the first text retrieval conference (TREC-1). In *Proceedings of the first Text Retrieval Conference*, pages 1–20, 1992.

[33] D. Harman. Overview of the first TREC conference. In *Proceedings of the 16th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '93, pages 36–47, 1993.

[34] B. He, C. Macdonald, J. He, and I. Ounis. An effective statistical approach to blog post opinion retrieval. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 1063–1072, 2008.

[35] B. He, C. Macdonald, and I. Ounis. Ranking opinionated blog posts using opinionfinder. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 727–728, 2008.

[36] T. Hoffman. Online Reputation Management is Hot — But is it Ethical, February 2008.

[37] M. Hu and B. Liu. Mining opinion features in customer reviews. In *Proceedings of the 19th International Conference on Artifical Intelligence*, AAAI'04, pages 755–760, 2004.

[38] X. Huang and W. B. Croft. A unified relevance model for opinion retrieval. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 947–956, 2009.

[39] M. Hurst and K. Nigam. Retrieving topical sentiments from online document collections. In *Document Recognition and Retrieval XI*, pages 27–34, 2004.

[40] K. Inui, S. Abe, K. Hara, H. Morita, C. Sao, M. Eguchi, A. Sumida, K. Murakami, and S. Matsuyoshi. Experience mining: Building a large-scale

database of personal experiences and opinions from web documents. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 314–321, 2008.

[41] V. Jijkoun, M. de Rijke, and W. Weerkamp. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 585–594, 2010.

[42] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 133–142, New York, NY, USA, 2002.

[43] Thorsten Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. Cambridge, MA, 1999.

[44] T. Joyce and R. M. Needham. The thesaurus approach to information retrieval. *American Documentation*, 9:192–197, 1958.

[45] K. Uehara K. Seki. Adaptive subjective trigger for opinionated document retrieval. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 25–33, 2009.

[46] T. Kurashima, T. Tezuka, and K. Tanaka. Mining and visualizing local experiences from blog entries. In *Proceedings of the 17th international conference on Database and Expert Systems Applications*, volume 4080 of *DEXA'06*, pages 213–222, 2006.

[47] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 111–119, 2001.

[48] M. Laver, K. Benoit, and J. Garry. Extracting policy positions from political texts using words as data. *American Political Science Review*, 97(2):311–331, 2003.

[49] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 120–127, 2001.

[50] J. H. Lee. Analyses of multiple evidence combination. In *Proceedings of the 20th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 267–276, 1997.

[51] S-W. Lee, J-T. Lee, Y-I. Song, and H-C. Rim. High precision opinion retrieval using sentiment-relevance flows. In *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 817–818, 2010.

[52] 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 the 17th Text REtrieval Conference*, TREC '08, 2008.

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

[54] T.-Y. Liu, J. Xu, T. Qin, W.-Y. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, 2007.

[55] Y. Lv and C. Zhai. Positional language models for information retrieval. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 299–306, 2009.

[56] C. Macdonald, B. He, I. Ounis, and I. Soboroff. Limits of opinion-finding baseline systems. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 747–748, 2008.

[57] C. Macdonald and I. Ounis. The TREC blogs06 collection : Creating and analysing a blog test collection. *DCS Technical Report Series*, 2006.

[58] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC-2007 blog track. In *Proceedings of the 16th Text REtrieval Conference*, TREC '07, 2007.

[59] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC 2009 Blog Track. In *Proceedings of 18th TExt Retrieval Conference*, TREC '09, 2009.

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

[61] R. Manmatha, T. Rath, and F. Feng. Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 267–275, 2001.

[62] R. Manmatha and H. Sever. A formal approach to score normalization for meta-search. In *Proceedings of the 2nd international conference on Human Language Technology Research*, HLT '02, pages 98–103, 2002.

[63] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[64] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, June 1999.

[65] M. E. Maron and J. L. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of Association for Computing Machinery*, 7:216–244, July 1960.

[66] G. Mishne and M. de Rijke. A study of blog search. In *In Proceedings of 28th European Conference on IR Research*, ECIR '06, pages 289–301, 2006.

[67] G. Mishne and N. Glance. Predicting movie sales from blogger sentiment. In *AAAI 2006 Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006.

[68] M. M. S. Missen, M. Boughanem, and G. Cabanac. Opinion finding in blogs: a passage-based language modeling approach. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, RIAO '10, pages 148–152, 2010.

[69] M. Montague and J. A. Aslam. Relevance score normalization for metasearch. In *Proceedings of the 10th ACM International Conference on Information and Knowledge Management*, CIKM '01, pages 427–433, 2001.

[70] M. Montague and J. A. Aslam. Condorcet fusion for improved retrieval. In *Proceedings of 11th International Conference on Information and Knowledge Management*, CIKM '02, pages 538–548, 2002.

[71] S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima. Mining product reputations on the web. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 341–349, 2002.

[72] T. Mullen and N. Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, EMNLP'04, 2004.

[73] T. Mullen and R. Malouf. A preliminary investigation into sentiment analysis of informal political discourse. In *AAAI Symposium on Computational Approaches to Analysing Weblogs*, AAAI-CAAW, pages 159–162, 2006.

[74] S-H. Na, Y. Lee, S-H. Nam, and J-H. Lee. Improving opinion retrieval based on query-specific sentiment lexicon. In *Proceedings of the 31th European Conference on IR Research*, ECIR '09, pages 734–738, 2009.

[75] V. Ng, S. Dasgupta, and S. M. N. Arifin. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 611–618, 2006.

[76] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39(2-3):103–134, 2000.

[77] M. O'Hagan. Aggregating template or rule antecedents in real-time expert systems with fuzzy set logic. In *Proceedings of Annual IEEE Conference on Signals, Systems, Computers*, pages 681–689, 1988.

[78] I. Ounis, M. de Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC-2006 blog track. In *Proceedings of the 15th Text REtrieval Conference*, TREC '06, 2006.

[79] I. Ounis, C. Macdonald, and I. Soboroff. On the TREC blog track. In *Proceeding of the 2008 International Conference on Webloging and Social Media*, ICWSM 08, 2008.

[80] I. Ounis, C. Macdonald, and I. Soboroff. Overview of the TREC-2008 blog track. In *Proceedings of the 17th Text REtrieval Conference*, TREC '08, 2008.

[81] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2:1–135, January 2008.

[82] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '02, pages 79–86, 2002.

[83] K. C. Park, Y. Jeong, and S. H. Myaeng. Detecting experiences from weblogs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1464–1472, 2010.

[84] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 275–281, 1998.

[85] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 275–281, 1998.

[86] E. Riloff and J. Wiebe. Learning extraction patterns for subjective expressions. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, pages 105–112, 2003.

[87] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of 13th Conference on Information and Knowledge Management*, CIKM '04, pages 42–49, 2004.

[88] S. E. Robertson. The probability ranking principle in ir. pages 281–286. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

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

[90] S. E. Robertson, C. J. van Rijsbergen, and M. F. Porter. Probabilistic models of indexing and searching. In *Proceedings of the 3rd International ACM Conference on Research and Development in Information Retrieval*, SIGIR '80, pages 35–56, 1981.

[91] R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? In *Proceedings of the IEEE*, volume 88, pages 1270–1278, 2000.

[92] G. Salton. *Automatic Information Organization and Retrieval.* McGraw Hill Text, 1968.

[93] G. Salton. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Inc., 1971.

[94] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. *Journal of Association for Computing Machinery*, 15:8–36, January 1968.

[95] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[96] M. Sanderson and J. Zobel. Information retrieval system evaluation: effort, sensitivity, and reliability. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 162–169, 2005.

[97] R. L. Santos, B. He, C. Macdonald, and I. Ounis. Integrating proximity to subjective sentences for blog opinion retrieval. In *Proceedings of the 31th European Conference on IR Research*, ECIR '09, pages 325–336, 2009.

[98] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Survey*, 34(1):1–47, 2002.

[99] H. Sever and M. R. Tolun. Comparison of normalization techniques for metasearch. In *Proceedings of the 2nd International Conference on Advances in Information Systems*, ADVIS '02, pages 133–143, 2002.

[100] K. Sparck Jones and C. J. van Rijsbergen. Report on the Need for and Provision of an “Ideal” Information Retrieval Test Collection. British Library Research and Development Report 5266, University of Cambridge, 1975.

[101] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. *Information Processing Management*, 36:779–808, 2000.

[102] P. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, 2002.

[103] P. Turney and M. L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4):315–346, 2003.

[104] P. K. T. Vaswani and J. B. Cameron. *The National Physical Laboratory Experiments in Statistical Word Associations and Their Use in Document Indexing and Retrieval*. National Physical Laboratory, Teddington, 1970.

[105] R. D. De Veaux. Finding Transformations for Regression Using the ACE Algorithm. *Sociological Methods and Research*, 18(327):327–359, 1989.

[106] O. Vechtomova. Facet-based opinion retrieval from blogs. *Information Processing and Management*, 46(1):71–88, 2010.

[107] E. M. Voorhees and D. K. Harman. *TREC: Experiment and Evaluation in Information Retrieval*. Digital Libraries and Electronic Publishing. MIT Press, 2005.

[108] T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. Opinionfinder: a system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, HLT-Demo '05, pages 34–35. Association for Computational Linguistics, 2005.

[109] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'07, pages 391–398, 2007.

[110] R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man, and Cybernetics*, 18(1):183–190, 1988.

[111] K. Yang, N. Yu, A. Valerio, H. Zhang, and W. Ke. Fusion approach to finding opinions in blogosphere. In *Proceedings of International Conference on Weblogs and Social Media*, ICWSM'07, 2007.

[112] J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, ICDM ’03, 2003.

[113] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’07, pages 271–278, 2007.

[114] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’07, pages 271–278, 2007.

[115] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’01, pages 334–342, 2001.

[116] 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.

[117] M. Zhang and X. Ye. A generation model to unify topic relevance and lexicon-based sentiment for opinion retrieval. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’08, pages 411–418, 2008.

[118] Q. Zhang, B. Wang, L. Wu, and X. Huang. Fdu at trec 2007: Opinion retrieval of blog track. In *Proceedings of the 16th Text REtrieval Conference*, TREC’07, 2007.

[119] W. Zhang, L. Jia, C. Yu, and W. Meng. Improve the effectiveness of the opinion retrieval and opinion polarity classification. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management*, CIKM ’08, pages 1415–1416, 2008.

[120] W. Zhang, C. Yu, and W. Meng. Opinion retrieval from blogs. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, CIKM ’07, pages 831–840, 2007.