Modeling Score Distributions for Information Retrieval

A dissertation presented

by

Keshi Dai

to the Faculty of the Graduate School
of the College of Computer and Information Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Northeastern University
Boston, Massachusetts
April, 2012
NORTHEASTERN UNIVERSITY
GRADUATE SCHOOL OF COMPUTER SCIENCE
Ph.D. THESIS COMPLETION APPROVAL FORM

THESIS TITLE: Modeling Score Distributions for Information Retrieval

AUTHOR: Keshi Dai

Ph.D. Thesis Approved to complete all degree requirements for the Ph.D. Degree in Computer Science.

April 26, 2012
Date

April 23, 2012
Date

April 27, 2012
Date

April 24, 2012
Date

GRADUATE SCHOOL APPROVAL:

April 26, 2012
Date

DIRECTOR, GRADUATE SCHOOL

ORIGINAL COPY DEPOSITED IN LIBRARY:

April 26, 2012
Date

REFERENCE LIBRARIAN

COPY RECEIVED IN GRADUATE SCHOOL OFFICE:

April 26, 2012
Date

Recipient’s Signature

Distribution: Original to Reference Library (boxed- with signature approval sheet)
One Copy to Thesis Advisor
One Copy to Each Member of Thesis Committee
One Copy to Director of Graduate School
One Copy to Graduate School Office (with signature approval sheet, including all signatures)

Copies submitted to those other than Reference Library should be bound in a black hardcover binder with a squeeze back. (Available in N.U. Bookstore).
Acknowledgments

I am very grateful to have many people helping me during my PhD study. I would like to express my sincere appreciation and gratitude to them for their encouragement and advice along these years.

First of all, I would like to thank my advisor Javed Aslam, who helped and always encouraged me to study and research in IR. He taught me how to research a problem, achieve goals, and present my work to other people. Most importantly, his broad knowledge and critical thinking opened a new world to me, which inspired me to look at a problem from different perspectives, and showed me how to look beyond the obvious and have a deeper understanding of the problem. Having him as my advisor was my biggest luck and a great honor in my life.

I would also like to thank my senior lab mates Virgil Pavlu and Evangelos Kanoulas. They are both great researchers, and helped me with all the hands on experience of conducting research projects, analyzing experiment results, and writing technical articles. The nights we worked together for the paper deadline at 12PM Samoa Time are among the best memories that I will never forget. Virgil deserves a special thank you for introducing me to the IR lab and helped me to start my PhD research in this field. I am very grateful about this.

I also want to give special thanks to Harriet Fell, who was my advisor when I was doing speech processing research during my first three years in the graduate school. It was such a pleasure working with her. She helped me to start my research and taught me how to write my first research paper. I am also grateful for her help with reading my thesis and providing useful comments.

Special thanks also go to Avi Arampatzis and Rajmohan Rajaraman for being my PhD committee members and giving me many valuable feedbacks along with suggestions on my thesis.

I also want to thank my lab mates for their help and friendship: Pavel Metrikov, Shahzad Rajput, Stefan Savev, Peter Golbus, Matthew Ekstrand-Abueg, and Maryam Bashir.

Finally, I would like to express my gratitude to my parents for their unconditional love and being supportive all the time. I also owe many thanks to my fiancé Bing Sun. I would have never finished my PhD without her love and companionship.
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2</td>
<td>Methodology</td>
<td>33</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Motivation</td>
<td>33</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Variational Bayesian inference</td>
<td>37</td>
</tr>
<tr>
<td>3.3</td>
<td>Experiments</td>
<td>40</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Results and analysis</td>
<td>41</td>
</tr>
<tr>
<td>3.4</td>
<td>Summary</td>
<td>61</td>
</tr>
<tr>
<td>4</td>
<td>Score Distribution Model</td>
<td>63</td>
</tr>
<tr>
<td>4.1</td>
<td>Related Work</td>
<td>64</td>
</tr>
<tr>
<td>4.2</td>
<td>Motivation</td>
<td>66</td>
</tr>
<tr>
<td>4.3</td>
<td>Score Distribution Derivation</td>
<td>66</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Deriving the term distribution from raw statistics</td>
<td>66</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Deriving the score distribution from scoring functions</td>
<td>70</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Inferring the score distribution for relevant documents</td>
<td>76</td>
</tr>
<tr>
<td>4.4</td>
<td>Summary</td>
<td>82</td>
</tr>
<tr>
<td>5</td>
<td>Inferring Score Distributions without Relevance</td>
<td>85</td>
</tr>
<tr>
<td>5.1</td>
<td>Related Work</td>
<td>86</td>
</tr>
<tr>
<td>5.2</td>
<td>Extended Expectation Maximization</td>
<td>87</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Methodology</td>
<td>87</td>
</tr>
<tr>
<td>5.2.2</td>
<td>EM and the multi-system extended EM</td>
<td>89</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Experiments</td>
<td>92</td>
</tr>
<tr>
<td>5.2.4</td>
<td>Precision-recall curve</td>
<td>93</td>
</tr>
<tr>
<td>5.2.5</td>
<td>Expected average precision</td>
<td>95</td>
</tr>
<tr>
<td>5.3</td>
<td>Application to Metasearch</td>
<td>96</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Methodology</td>
<td>98</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Experiments and results</td>
<td>99</td>
</tr>
<tr>
<td>5.4</td>
<td>Summary</td>
<td>103</td>
</tr>
<tr>
<td>6</td>
<td>Conclusions</td>
<td>105</td>
</tr>
<tr>
<td>6</td>
<td>Bibliography</td>
<td>107</td>
</tr>
</tbody>
</table>
# List of Tables

1. Four possible categories for a document in the search results ........................................... 23
2. An example of a ranked list with the corresponding precision and recall values. .......................................................... 24
3. The average number relevant documents retrieved by BM25 for each query when the different numbers of additional query terms are applied to the original title query. .......................................................... 51
4. Root Mean Square Error (RMSE) between the actual and the inferred precision-recall curves, averaged over about 200 queries from TREC collections. .......................................................... 56
5. Mean Absolute Error (MAE) between actual and inferred precision-recall curves, averaged over about 200 queries from TREC collections. .......................................................... 56
6. RMSE and MAE between the actual and the inferred precision-recall curves reported separately on TREC6-12 ad-hoc runs. .......................................................... 58
7. Mean and standard deviation of the RMSError and ABSError of the estimated PR curves for 116 automatic systems over 50 queries submitted to TREC 8 Ad Hoc Track. .......................................................... 94
8. Mean average precision achieved by meta-search using regEM, extEM, judSD, combMNZ based on randomly selected 10, 20, 30, 40, 50 systems from all automatic systems submitted to TREC 6, 7, 8, 9, 10, 12 averaged over 20 times .......................................................... 100
9. Mean average precision achieved by meta-search using regEM, extEM, judSD, combMNZ based on randomly selected 5, 10, 15, 20, 25 systems from top ranked 30 systems submitted to TREC 6, 7, 8, 9, 10, 12 averaged over 20 times .......................................................... 102
Mean average precision achieved by meta-search using regEM-interp, extEM-interp, judSD-interp, combMNZ based on randomly selected 5, 10, 15, 20, 25 systems from top ranked 30 systems submitted to TREC 6, 7, 8, 9, 10, 12 averaged over 20 times
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The snapshot of searching “today’s weather” on Google.com</td>
</tr>
<tr>
<td>2</td>
<td>The snapshot of a ranked list of documents with scores and judgements</td>
</tr>
<tr>
<td>3</td>
<td>An example of inferred score distributions compared to the actual normalized frequency distributions of the scores (green: relevant, red: non-relevant).</td>
</tr>
<tr>
<td>4</td>
<td>The diagram shows how to infer the probability of document relevance through score distributions</td>
</tr>
<tr>
<td>5</td>
<td>Actual PR curve and its interpolated version for the previous ranked list example</td>
</tr>
<tr>
<td>6</td>
<td>The distribution of BM25 scores for all 113,947 documents (containing at least one query term) on query “Ireland peace talks”. Note the different slopes at the left and at the right of the mean. Truncating the list at rank 1,000 would cause the scores’ distribution to look like an exponential one. Histograms of BM25 computation are shown step-by-step starting with TF for individual query terms(top) and ending the BM25 final score(bottom).</td>
</tr>
<tr>
<td>7</td>
<td>The histogram over the scores of non-relevant and relevant documents along with the best fit exponential and Gaussian distributions (top plot) the Gamma and $k$-Gaussians distribution (bottom plot).</td>
</tr>
<tr>
<td>8</td>
<td>The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution, over all IR models, topics and topic formulations for TREC 6, 7, 8.</td>
</tr>
<tr>
<td>9</td>
<td>The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution, over all IR models, topics and topic formulations for TREC 9, 10, 12.</td>
</tr>
</tbody>
</table>
10 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 6. .......................... 45

11 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 7. .......................... 45

12 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 8. .......................... 45

13 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 9. .......................... 46

14 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 10. .......................... 46

15 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 12. .......................... 46

16 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 6. .......................... 47

17 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 7. .......................... 47

18 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 8. .......................... 48

19 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 9. .......................... 48

20 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 10. .......................... 49

21 The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 12. .......................... 49
22 The score distribution of the relevant document for TREC8 topic 401: foreign minorities, Germany. Three topic formulations are used: title, title + description, title + description + narrative ........................ 50

23 The average values of K and M against the number of query terms for TREC 6, 7, 8. ................................................................. 52

24 The number of Gaussian components, K, against number of relevant documents retrieved over all ranked lists for each TREC data set. . . 53

25 Precision-Recall curve (blue) for query 434 and the BM25 retrieval function implemented by Terrier. It is easy to see that the PR curve estimated from the GkG model (magenta) is much better than the PR estimated from the EG model (brown). Yellow bars indicate the number of non-relevant documents in each recall interval. ........... 57

26 The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution for all automatic runs in TREC 6, 7, 8. ................................................................. 58

27 The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution for all automatic runs in TREC 9, 10, 12. ................................................................. 59

28 The number of Gaussian components, K, against number of relevant documents retrieved over all ranked lists for each TREC data set. . . 60

29 The empirical histograms of term frequencies resulting to the final scores for the query (Ireland Peace Talks) over the TREC8 Ad Hoc Track collection for BM25 (top) and JM Language Model (bottom). Each column corresponds to a single stemmed query term while the rows correspond to progressively more complex transformations of the term frequency (TF) up to the final score for the two ranking functions. DL is the document length, ADL is the average document length, CTF is the collection term frequency, and TN is the number of terms in the collection. ........................................ 67

30 The empirical histogram and the Gamma density function fit over the $\frac{DL}{TF}$ scores for term system in TREC8. ............................ 71

31 The mixture of Gamma distributions before and after the inversion, for different quality classes ....................................................... 72

32 The empirical histograms, analytically numerical plot, and MLE Gamma fitting of the distribution of approximated BM25 scores and JM language model scores for term system in TREC8. ..................... 73
33 Roberston’s TF and empirical histograms of BM25 scores with different $k_1$ for term system in TREC8 ........................................... 76
34 MLE Gamma fitting over scores of all retrieved documents for all query terms and query “Ireland Peace Talks” ............................. 77
35 A family of precision-recall curves fit through the points \{(0, 1), (rp, rp), (1, 0)\} for \(rp = 0.1, 0.2, \ldots, 0.9\). .................................................... 79
36 Inferred relevant document score distribution and empirically histogram for the TREC8 query “Estonia, economy” .......................... 80
37 Typical form of the relevant document score distribution in TREC8. . 80
38 The score distribution of the relevant documents for TREC8 query “Estonia, economy”. The red dash lines are two Gaussian components, and the blue line is the mixture of two Gaussians. The scores are produced by BM25, and the Gaussian components are inferred by the same process used in Chapter 3. ................................. 83
39 A document $d$ with scores from two different systems and the conflation rate in each system. $\phi$ and $\psi$ shown are the Gaussian-Exponential model. .......................................................... 90
40 The average RMSError and ABSError of inferred PR curve through regEM and extEM for different numbers of TREC8 systems. ...... 95
41 The scatter plot of mean EAP estimated by regEM(blue square dots)/extEM(red star dots) comparing with the one estimated by judSD for all automatic systems submitted to TREC6, 7, 8, 9, 10, 12 ....................... 97
42 The performance improvement of regEM, extEM, and judSD using the interpolation method over TREC 6, 7, 8, 9, 10, 12 ..................... 103
Abstract

When a user submits a query to a search engine, the search engine computes a score for each document according to its relevance to the query, and ranks the documents based on their scores. Due to the complexity of the modern search engine, the score itself is not sufficient for the information retrieval application requiring combining different ranked lists. Inferring the score distributions for relevant and non-relevant documents and estimating the probability of relevance become imperative. In this thesis, we address two major research questions: (1) How to model score distributions in a more accurate manner for relevant and non-relevant documents? (2) How can score distributions be better inferred in practice when the relevance information is absent?

In the first part of the thesis, we show the existing problems of today’s most widely used score distribution model, and propose to model the relevant document scores by a mixture of Gaussian distributions and the non-relevant scores by a Gamma distribution. Score distributions are further modeled in a more systematic manner. With a basic assumption of the distribution of terms in a document, the distribution of the produced scores for retrieved documents can be derived through the transformations applied on the term frequency. Meanwhile, the score distribution of relevant documents can also be derived through a general mathematical framework given the score distribution for all retrieved documents.

The second part of the thesis presents a new framework for inferring score distributions when the relevance information is unavailable. The new inference process extends the expectation maximization algorithm by simultaneously considering the ranked lists of documents returned by multiple retrieval systems, and encodes the constraint that the same document retrieved by multiple systems should have the same, global, probability of relevance. Combined, we demonstrate that it is more effective when it is applied on the task of metasearch.
Chapter 1

Introduction

With dramatically growing usage of online storage and Internet service, an unimaginably vast amount of information is digitized, and can be accessed through World Wide Web (WWW). How to efficiently and effectively find the information we want becomes an imperative challenge in our everyday life. As a result, searching has become the primary online activity [62]. People use the search engine to work, to learn, and to discover. It has already changed the way we access information, and keeps reshaping our life.

Meanwhile, the information we can receive from a search engine has been significantly diversified. In the late 1990s, the page displaying the search results only contained a simple list of document links, but today it encompasses all sorts of relevant information (e.g. news, images, videos, and even geographic locations). For instance, if you search “today’s weather” on Google, Google not only recognizes your location and presents the specific local weather information, but also returns relevant web pages from popular weather site Weather.com, along with various news stories that discuss the topic “today’s weather”. (see Figure 1).

This small example demonstrates the complexity of a modern search engine. The capability of processing various unstructured information units and combining them in a meaningful way for the end user becomes the fundamental requirement for a today’s search engine. The information can be collected from different sources, servers, and locations, or by a set of underlying sub-engines. A naive document merging method is no longer guarantee to give us the desired result. Instead, we need a reasonable and effective score normalization approach to convert a score to a meaningful quantity such as the probability of the document relevance, so the information from those sources which may have totally different characteristics can be easily blended.
Figure 1: The snapshot of searching “today’s weather” on Google.com

For instance, when a document is retrieved by a search engine, it is assigned a score to indicate how relevant it is to the given query. However, this score only serves as a relative indicator of the relevance for comparing it with other documents. For example, a document with a higher score is more likely to be relevant, but the score does not give us any clue about how much relevant information a document exactly contains. Consequently, having an accurate transformation framework that converts the score to the estimation of certain absolute measurement of the relevance (e.g. probability of document relevance) is highly beneficial to the modern search engine.

In this thesis, we investigate how scores are distributed within two categories of documents: relevant and non-relevant documents, and study the model parameter inference algorithm for the practical usage. With a more accurate statistical distribution model inferred for relevant and non-relevant document scores, the proba-
bility of document relevance can be estimated more precisely, which can be of use for many information retrieval applications.

1.1 Information Retrieval

Information retrieval is the foundation for modern search engine development. It studies the methodologies for finding information within documents that matches a particular user’s interest [76]. These days, the definition of a document has been widely broadened. Documents are not only text related objects, such as text files, web pages, blog entries, and news stories, but also include all categories of multimedia contents like photos, speeches, music, and videos. The user’s request that expresses people’s information need, and typically consists of a sequence of meaningful key words. In the latest Siri application on Apple’s iOS [1], a request can also be a real question asked by the user.

A typical information retrieval system (such as a search engine) is designed to deal with the problems of representing, storing, organizing, and accessing the information [21]. Its functionality usually consists of four major components: crawling, indexing, retrieval, and evaluation. In this section, we will give a brief overview of each component and its responsibility in the system.

Crawling

Obtaining documents and making information accessible is the first step when building a search engine. Crawling is an automatic process that identifies and acquires documents for the search engine. For example, a web crawler browses WWW in a particular order via the links between the documents on the web, and creates a copy of all visited pages for later indexing process to provide fast searches. It is also important to keep crawled documents fresh, so after a certain period of time, the archived documents need to be updated.

Indexing

After crawling, a series of data preprocessing steps is often conducted on the crawled documents before indexing, which may consist of parsing, stopping, stemming, and link analysis. Parsing process tokenizes the text document, and converts it into a structured data type with several predefined document fields. Each field contains a sequence of text tokens. In order to reduce the size of the vocabulary of the index, the stopping process removes some of the most common, short function words, which do not have any semantic meanings, such as a, the, is, and which. Besides, due to the complexity of the natural language, a word usually appears in different forms based on the context to express the same meaning. Stemming re-
duces derived words to their stem or root form to maintain a minimum number of vocabularies. For instance, talk is the stem form of talks, talked, and talking. Popular stemming algorithms often used in the indexing process include Porter [65], Snowball [64], and Krovetz [57]. Link analysis is another important preprocessing task for web documents. It is conducted on the web graph to identify the importance of a web page. There are also many other text preprocessing techniques that can be applied before the indexing such as part-of-speech tagging, name entity finding, co-reference resolution, and synonym identification. Many preprocessing techniques are language dependent. For example, tokenization is very difficult in Chinese, but stemming is very trivial. Arabic is opposite, and has very complicated stemming rules.

After crawled documents are preprocessed, the index creation can start. It reorganizes the documents in a way that they can be easily retrieved by just looking up the vocabulary. The whole process first builds a lookup table for storing various document statistics (e.g., number of documents, document length, number of terms) to facilitate later computation. For each word in the vocabulary, it computes its weight for the document that it occurs. This weight to some extent reflects the relative importance of a word in the document. There can be multiple weights like different relevance scores associated with a term in a document. Storing the term weight in the index helps the score computation to achieve a faster ranking. In the end, the index, in fact an inverted list, has been built to enable the fast full text search. It transforms the original document-term information into the term-document information. In an inverted list, each term points to a list of documents (along with corresponding term frequency, positions, weightings, and other information) where it appears.

**Retrieval**

Retrieval is a complex process. At this stage, the information retrieval system takes a user’s request, processes the query, computes scores for underlying documents in the index, and returns a ranked list of documents matching the query.

Query processing aims to interpret the actual user’s information need. The information need is usually expressed by a sequence of query words. Most of the time, they are unclear and ambiguous. It is the search engine’s responsibility to help users better define their interests. Many techniques have been developed to address this problem such as query suggestion that help users reformulate their queries by using historical query logs. Moreover, with further interaction with users, the search engine can also rely on relevance feedback to have a better understanding of the information need behind the query. Relevance feedback expands
queries by adding more informative key words appearing in the documents that are (or are more likely to be) relevant to user’s request.

With an interpreted user’s search intention, an information retrieval system retrieves a list of documents that matches the request, and ranks those documents based on their scores computed based on certain underlying retrieval model. Retrieval models (e.g., vector space model, probabilistic model, language model) in information retrieval use statistical properties of the query terms in the index to estimate the document relevance [39]. The idea of using the statistics like word frequency to represent text first started by H.P. Luhn in 1950s, and becomes popular in many fields, such as natural language processing from the 1990s [39]. Apart from that, linguistic structure of text is also used to have a better estimation of the relevance in some advanced models.

Evaluation

Another critical component during developing an information retrieval system is evaluation, which measures how well a document ranking matches a user’s expectation. Evaluation also compares the performance of different retrieval algorithms. There are two most widely used quantities in the evaluation: precision and recall. Precision is the proportion of retrieved documents that are relevant, and recall is the proportion of relevant documents that are retrieved. Many evaluation metrics are developed from these two quantities.

To compute different evaluation metrics, we also need the relevance assessment. In this process, documents are labeled as relevant or non-relevant, and the labeling task is usually completed by professional human assessors. Besides, when people evaluate a practical search engine, user interaction data such as the click-through rate and the dwell time on a document are also used to infer the document relevance, which can be directly applied to an online controlled experiment (A-B test).

1.2 Score Distributions

Given a user request, a search engine assigns a score to each document in the underlying collection according to some definition of relevance of the document to the user’s request and returns a ranked list of documents based on their scores. A document with higher score is considered to be more relevant than the one with a lower score. However, this classical scoring mechanism does not reflect the amount of relevant information contained in a document. For instance, a document may be assigned with varying scores by different search engines for the same input query, although the document contains exactly the same amount of relevant information.
This is problematic for some retrieval applications such as data fusion and recall-oriented retrieval. In data fusion, documents are retrieved by different search engines, or from different domains, servers. Their scores need to be normalized in a proper way in order to merger those documents together and construct a final ranked list. In recall-oriented retrieval (e.g. legal search, patent search), we need to estimate the number of relevant documents under certain score threshold to determine whether more documents need to be judged. Both applications would benefit from the estimation of document relevance from score distributions.

Assuming that every retrieved document can be classified as relevant and non-relevant according to how much it meets user’s information need, the ranked list of documents is in fact a mixture of both relevant and non-relevant documents (see Figure 2).

Two distinctive distributions can be used to model relevant document scores and non-relevant document scores separately. For example, in Figure 3, the green bars show how scores are actually distributed within relevant documents which can be modeled by a Gaussian distribution (green dash-dot line). The red bars show how non-relevant scores are distributed, and they can be modeled by an exponential distribution (red dash line). From those two inferred distributions, the probability of the relevance for a document given a particular score can be calculated through Bayes’ law (see Figure 4). We will discuss it in more detail in Chapter 5.

Therefore, modeling and inferring the distribution of relevant and non-relevant documents over scores in a reasonable way could be highly beneficial for many information retrieval problems. In information filtering, topic detection, and recall-oriented retrieval, score distributions can be utilized to find the appropriate threshold between relevant and non-relevant documents [82, 83, 14, 91, 37, 80]. In distributed IR and metasearch it can be used to normalize document scores and combine different collections or the outputs of several search engines [25, 58].

Under the assumption of binary relevance, numerous combinations of statistical distributions have been proposed in the literature to model the score distributions of relevant and non-relevant documents, such as two Gaussians with equal variance [82], two Gaussians with unequal variance [83], two Poisson distributions [30], and two Gamma distributions [25]. The most popular model has been a Gaussian distribution for relevant documents and an exponential distribution for non-relevant documents [58], and this model has been widely used in practical applications [58, 91, 10, 11]. The most recently proposed model is a truncated version of Gaussian-exponential model [11]. The strong argument in all studies for
choosing any particular combination of distributions is the goodness-of-fit to a set of empirical data. However, there is no real consensus on the choice of the distributions due to the complexity of the underlying process in the retrieval model that generates document scores. More detailed analysis of those models will be discussed in Chapter 3 and 4.

On the other hand, even with the assumption of the “correct” model choice, we are still faced with the problem of how to infer the model parameters. Inferring the score distributions associated with relevant and non-relevant documents can only be accomplished when many relevance judgments are available. However, we are
most often faced with the situation of estimating model parameters in the absence of relevance information. *Expectation maximization* (EM) [28] has been proposed to solve the problem in most score distribution applications [14, 11, 10, 58]. This algorithm optimizes the parameters of two distributions, and maximizes the log-likelihood associated with the data. Although with some success, it has been noted that EM suffers from issues such as treating all data equally, being very sensitive to initialization, and converging to a local optimum instead of the global one [10, 11, 12]. This makes applying score distributions to solving real IR problems an extremely difficult task. To address those issues, we will propose a new inference framework in Chapter 5.

### 1.3 Research Themes

When modeling and inferring score distributions for a list of document scores from a real information retrieval system, we are faced with two problems: (1) What score distribution model should be used? (2) How can model parameters be inferred
when the document relevance information is unknown?

**What score distribution model should be used?**

Modeling score distribution has been studied for a long time. Numerous combinations of statistical distributions have been proposed to model relevant and non-relevant document scores. The choice of the model is mostly determined by the evaluation on the empirical data. Following the same fashion, in this thesis we also observe some problems existing in the current most widely used score distribution model. We analyze why the current model fails under certain circumstances on the empirical data, and propose an improved empirical model to address the issue.

On the other hand, we also approach the same question in a systematic manner. For most retrieval models, the score assigned to a document is a consequence of a sequence of mathematical transformations applied on the term frequency. With a basic assumption of the distribution of terms in a document, the distribution of the document scores can be derived. Meanwhile, score distributions are closely connected to two evaluation metrics: precision and recall. Those two popular evaluation measures can be derived using score distributions, and vice versa. Therefore, given a general IR precision-recall model and a score distribution for non-relevant documents derived from score transformation, the score distribution for the relevant documents can be determined as well. This framework offers a new way to model score distributions from a theoretical perspective.

**How can model parameters be inferred when the document relevance information is unknown?**

Even the choice of score distribution model is known, it is still extremely difficult to infer the model parameters for a mixture of two score distributions in practice. Expectation maximization is a widely used algorithm to find maximum likelihood estimates of parameters for a mixture model. The optimization process purely maximizes the likelihood function, but does not take account into any well-known information retrieval constraints, such as the common document retrieved by different search engines for the same query would have the same probability of relevance. If the optimization iteration can encode this, the inference process would benefit from it. We extend the transitional expectation maximization algorithm to incorporate this constraint, and demonstrate its utility under the metasearch scenario.

**1.4 Thesis Outline**

This thesis is organized in the following way. In Chapter 2, we give a brief background overview of information retrieval such as scoring functions, statistical dis-
tributions, precision recall curve, and other technique concepts that will be used in the later chapters.

In Chapter 3, several score distribution models and related work are reviewed. We focus on the intuition behind those models and show their limitations. Then a new score distribution model is proposed based on an analysis of the empirical data. The new model uses a mixture of Gaussian distributions to model relevant document scores, and a Gamma distribution to model non-relevant ones. Variational Bayesian inference is used to find the best number of Gaussians for the relevant score distribution.

In Chapter 4 we start with a basic assumption, and derive the score distribution for non-relevant documents based on the transformations applied on the scoring function. Further, by using a simple model for precision-recall curves, we present a general mathematical framework which, given any score distribution for non-relevant documents, produces an analytical formula for the distribution of relevant scores.

In Chapter 5, a new framework is proposed to better infer score distributions and the probability of the document relevance. We extend the expectation maximization algorithm to estimate model parameters for multiple systems simultaneously, and take advantage of the information and constraints present in multiple retrieved lists. Furthermore, we apply proposed extended expectation maximization algorithm to metasearch and demonstrate its utility.

The thesis concludes with Chapter 6, which includes a summary of the contributions of the thesis and suggestions for the future research.
Chapter 2

Background

In this chapter, we introduce the background knowledge required to understand the following chapters. We first will give an overview of the concept of document relevance and probability ranking principle, as well as several popular retrieval models. Later, experiment setup and basic IR evaluation metrics will be presented. In the end, we will briefly cover statistical distributions used for modeling document scores and the corresponding model parameter estimation methods. More details of related work on score distributions will be covered in later chapters.

2.1 Relevance and Retrieval Models

2.1.1 Relevance

An information retrieval system ranks documents according to their predicted relevance to user’s query based on certain retrieval model. There are several key steps in a typical search activity. A user first expresses his or her information need by composing a query, which is usually ambiguous, and consists of a sequence of key words. After receiving the query, the retrieval system interprets user’s information need, and tries to match it with most relevant documents. The relevance used in the system is defined by the underlying relevance model or retrieval model that predicts a relevance score for each document for the given query. The score is often calculated based on the similarity between the interpreted user information need and indexed documents. User intent interpretation is critical to a successful retrieval task, and can be achieved through many approaches, such as query expansion or suggestion [75, 79, 27]. However, this is not the focus of this thesis. Instead, we assume that the received query perfectly reflects the user information need, and we investigate how to infer the probability of document relevance from the predicted relevance score produced by a retrieval model.
In fact, *relevance* is a complex and multi-faceted concept in information retrieval. It is very difficult to define relevance for a query even by the human, especially when the query cannot clearly reflect user’s intent. Many studies have shown that when different people are asked to judge the relevance of documents for a given query, they can often disagree with one another [89, 88, 4, 84, 77].

In information retrieval, relevance usually can be classified into two categories: topical relevance and user relevance [39]. If a document is judged as topical relevant to a given query, it means that the document and the query is about the same subject. For example, a document discussing the seasonal weather in New York is topical relevant to the query “New York”. On the other hand, user relevance is more “subjective” and can be affected by many factors. Factors themselves and their importance may also change over the time or under the different circumstances. For instance, the relevance of a news story is often event driven, and typically has a life cycle. It is only relevant to a query within a particular time window or with the occurrence of certain event. Another example is that user sometimes considers a document less relevant or even non-relevant if he or she has already seen a similar document before or has already known the facts the document talks about. Other document characteristics that may also influence relevance include the language of the document, the intended target audience, and so on. For instance, a relevant document written in Chinese is always considered to be non-relevant for non-Chinese speakers. Therefore, user relevance measures whether the information offered by a document meets the user’s particular interest. In some applications such as document filter, topic relevance is also affected by the user’s interest. Topic is viewed as a set of all documents containing subjects that satisfy a user’s interest, which also shifts with the user’s interest over the time [13, 9].

### 2.1.2 Retrieval models

Retrieval models are fundamental in an information retrieval system. Over the past decades retrieval models have evolved from the earliest simple boolean retrieval to today’s sophisticated learning-to-rank algorithm. Here we give a brief introduction to a range of common retrieval models.

**Boolean retrieval model**

The boolean retrieval model has been used in the text retrieval engines since the earliest time. It retrieves documents that exactly match the given query terms. The query is specified by a sequence of Boolean operators. Take a query example as follows:
**Subaru AND Outback NOT (Restaurant OR Steakhouse)**

This query will retrieve a set of documents about car model Outback of the brand Subaru where both query words Subaru and Outback occur but not restaurant or steakhouse. The property of the relevant documents is defined through a set of query terms connected by boolean operators. This retrieval model is very efficient, because it only performs the sample text matching, and does not require computing the relevance score for each document. However, it lacks the ranking algorithm, and it treats all retrieved documents equally relevant to the query. Meanwhile, the performance of the model heavily relies on the quality of the query. A broad query may be possible to retrieve all relevant documents, but they are difficult to find due to the large size of the retrieval set. Well selected query terms can effectively reduce the size of the retrieval set, but they require intensive domain knowledge. Usually an expert, known as intermediary, is responsible for constructing a good query for the search in some specialized domains.

**Vector space model**

The vector space model was very popular in the IR research community a couple of decades ago. It is an algebraic model and represents text documents and queries using two term vectors. Although the vector space model provides a simple and easy-to-use framework for implementing term weighting, ranking, and relevance feedback, it gives little explanation on how the term weighting scheme and ranking algorithm are correlated to the relevance [39].

In the vector space model, a document or query is represented by a \( t \) dimensional term vector, where \( t \) is the number of indexed terms. For example, we can write the vector for document \( i \) as:

\[
d_i = (f_{i1}, f_{i2}, \cdots, f_{ij}, \cdots, f_{it})
\]

where \( f_{ij} \) is the weight of \( j^{th} \) term for document \( i \). The term weight can be simply the term frequency, how many times a term occurs in a document, or any other related statistics. Hence, the whole collection of documents is now considered as a matrix of term weights. Each row stands for a document in the collection, and each column corresponds to a unique indexed term in the collection. Similarly, a query \( q \) can also be written using the same term dimensions:

\[
q = (g_1, g_2, \cdots, g_j, \cdots, g_t)
\]

this time \( g_j \) is the weight for the \( j^{th} \) term in the query, i.e. the count of the term
occurrences in the query.

Once we have the document vector and the query vector, a similarity measure could be established. The most common similarity measure used in the vector space model is the *cosine* similarity. It essentially measures the cosine of the angle between two vectors:

\[
cosine(d_i, q) = \frac{d_i \cdot q}{\|d_i\| \|q\|} = \frac{\sum_{j=1}^{t} f_{ij} \cdot g_j}{\sqrt{\sum_{j=1}^{t} f_{ij}^2 \cdot \sum_{j=1}^{t} g_j^2}}
\]

(3)

After computing the cosine similarity between the query and each retrieved document, documents are ranked by the descending cosine values. A larger cosine similarity value means the document is more relevant to the query based on the term vector representation.

A good term weighting scheme is the key to the performance of the vector space model. Over many years, various term weighting schemes have been studied in the IR research community. The most popular one is the \( tf \cdot idf \) weighting scheme. It has two components \( tf \), term frequency, and \( idf \), inverse document frequency. \( tf \) is usually computed as the normalized term frequency over the total number of terms in the document. For example, \( tf \) for term \( k \) in document \( i \) is

\[
tf_{ik} = \frac{tf_{ik}}{\sum_{j=1}^{t} tf_{ij}}
\]

(4)

This quantity reflects the relative importance of the term in a document. If a meaningful term appears more often in a document than other documents, it is more likely to be related to the topic of that document. On the other hand, the normalization denominator takes the verboseness of the author into account, and diminishes the effect that the occurrence of a term in a longer document is in nature more than in a shorter document. The other component inverse document frequency \( idf \) for term \( k \) is computed as

\[
idf_k = \log \frac{N}{n_k}
\]

(5)

where \( N \) is the total number of documents in the collection, and \( n_k \) is the number of documents that term \( k \) appears. \( idf \) to some extent measures the global importance of a term in the collection. If term A appears in fewer documents than term B, term A is considered to have more distinctive value than term B. As an extreme example, stop words such as *a* and *the* occurs in every document, but they are useless in terms of ranking documents. In \( tf \cdot idf \) weighting scheme, those stop words although have high \( tf \) values but they will be assigned with extremely low
*idf* values. As a result, they have little impact on distinguishing the relevance of the documents.

**Probability ranking principle and probabilistic model**

When user submits a query to a retrieval system, the system should retrieve and rank the documents based on their estimated probability of relevance to the user’s query. This is the central idea of the probability ranking principle and the cornerstone of the probabilistic model. The formal statement of *probability ranking principle* in [72, 70] is as follows:

The probability ranking principle (PRP): If a reference retrieval system’s response to each request is a ranking of the documents in the collections in order of decreasing probability of usefulness to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data has been made available to the system for this purpose, then the overall effectiveness of the system to its users will be the best that is obtainable on the basis of that data.

With the reasonable assumption that the relevance of a document to a query is independent of other documents, the probability ranking principle yields the maximum expected number of relevant documents, and thus maximizes the *precision* metric, the proportion of relevant documents at a given rank [72, 70]. In practice, an accurate estimation of the probability of relevance for a document is the key to applying probability ranking principle to solving IR problems. Various probabilistic models have been proposed to address this issue.

**Probabilistic model** was first introduced by Robertson and Spärck Jones [71], and attempted to solve information retrieval problem under the probabilistic framework. Following the notation in [39], given a query *q*, and a document *d*, a document is either relevant (*rel*) or not (*nrel*). The probabilistic model computes the probability of a given document *d* being relevant is *P(rel|d)*, and the probability of the document being non-relevant is *P(nrel|d)*, which is equivalent to 1 − *P(rel|d)*. Based on Bayesian law, we have

\[
P(\text{rel}|d) = \frac{P(\text{rel})P(d|\text{rel})}{P(d)} \quad (6)
\]

\[
P(\text{nrel}|d) = \frac{P(\text{nrel})P(d|\text{nrel})}{P(d)} \quad (7)
\]
Considering the task of ranking documents as a Bayesian classification problem, document $d$ can be classified as relevant if $P(\text{rel}|d) > P(\text{nrel}|d)$, otherwise non-relevant. Because $P(d)$ is the normalization denominator for document $d$, the previous inequality is equivalent to $P(\text{rel})P(d|\text{rel}) > P(\text{nrel})P(d|\text{nrel})$, essentially, $\frac{P(d|\text{rel})}{P(d|\text{nrel})} > \frac{P(\text{nrel})}{P(\text{rel})}$. In order to estimate $P(d|\text{rel})$, we assume that documents are represented by a sequence of mutually independent features $(f_1, f_2, \ldots, f_t)$, and $f_i = 1$ indicates term $i$ occurs in the document, otherwise 0. Hence,

$$P(d|\text{rel}) = \prod_{i: f_i = 1} P(f_i|\text{rel})$$  \hspace{1cm} (8)$$

$$P(d|\text{nrel}) = \prod_{i: f_i = 0} P(f_i|\text{nrel})$$  \hspace{1cm} (9)$$

This assumption leads to the binary independence model (BIM), which further assumes a document is represented by a vector of binary features indicating the term occurrence. $P(f_i|\text{rel})$ is the probability that term $i$ occurs in relevant documents. Similarly, $P(f_i|\text{nrel})$ is the probability of the occurrence of term $i$ in non-relevant documents. Let $p_i = P(f_i|\text{rel})$ and $s_i = P(f_i|\text{nrel})$, and the probability of a document occurring in the relevant (or non-relevant) set equals to the product of the probability of each term occurring in a document from the relevant (or non-relevant) set. Hence, we have:

$$\frac{P(d|\text{rel})}{P(d|\text{nrel})} = \prod_{i: f_i = 1} \frac{p_i}{s_i} \prod_{i: f_i = 0} \frac{1 - p_i}{1 - s_i}$$  \hspace{1cm} (10)$$

$$= \prod_{i: f_i = 1} \frac{p_i}{s_i} \prod_{i: f_i = 1} \frac{1 - s_i}{1 - p_i} \prod_{i: f_i = 0} \frac{1 - p_i}{1 - s_i}$$  \hspace{1cm} (11)$$

$$= \prod_{i: f_i = 1} \frac{p_i(1 - s_i)}{s_i(1 - p_i)} \prod_{i} \frac{1 - p_i}{1 - s_i}$$  \hspace{1cm} (12)$$

In Equation 12, the second product is same for all retrieved documents, so we can ignore it. After applying the logarithm to eliminate the potential problem of multiplying many small numbers, we can define the scoring function for the binary independence model as

$$\sum_{i: f_i = 1} \log \frac{p_i(1 - s_i)}{s_i(1 - p_i)}$$  \hspace{1cm} (13)$$
For a term $i$ not appearing in the query, we assume $p_i = s_i$. Therefore, the score for a document is in fact the summation of the weights of all query terms that occur in the document. If we further assume that the probability of the occurrence of a term in relevant documents $p_i$ is constant, say 0.5, and $s_i$ is approximated by using the term occurrences in the entire collection, an idf-like weighting scheme is derived:

$$\frac{P(d|rel)}{P(d|nrel)} = \sum_{i:f_i=1} \log \frac{0.5(1 - \frac{n_i}{N})}{\frac{n_i}{N}(1 - 0.5)} = \log \frac{N - n_i}{n_i}$$

(14)

where $N$ is the total number of documents in the collection, and $n_i$ is the number of documents that term $i$ occurs, as known as document frequency.

**Okapi BM25 (BM25)** is one of the most popular and effective ranking algorithms based on the binary independence model with additional document and query term weights [73]. BM25 is based on probabilistic arguments and empirical validation, but it is not a fully theoretically justified model [39]. The scoring function of BM25 has many variations, the following is the most common one:

$$BM25(q,d) = \sum_{i \in Q} \log \frac{N - n_i}{n_i + 0.5} \cdot \frac{(k_1 + 1)f_i}{K + f_i} \cdot \frac{(k_2 + 1)qf_i}{k_2 + qf_i}$$

(15)

where $f_i$ is the frequency of term $i$ occurring in the document, $qf_i$ is the frequency of term $i$ in the query, $k_1$, $k_2$ and $K$ are parameters whose values should be tuned based on empirical data. $K$ is the parameter defining how the term frequency should be normalized by the document length, and equals to $k_1((1 - b) + b \frac{dl}{avdl})$. $dl$ is the document length, and $avdl$ is the average document length of the collection. For typical TREC experiments, $k_1 = 1.2$, $k_2$ varies from 0 to 1000, $b = 0.75$ [39].

**Language model**

Instead of modeling the probability of document relevance, the language model attempts to tackle the problem from a different angle. It simulates the process of generating a document by sampling words according to the underlying probability distribution of the vocabulary. The simplest language model is the unigram model, where each word in the collection is associated with a probability of the occurrence. If a document is treated as a sequence of words, then the language model predicts what the next word in the sequence should be with a probability. Moreover, to handle more complex text, there is $n$-gram language model that defines the probability of the next word in the sequence given the previous $n - 1$ words.

The language model has been successfully used in many natural language processing applications such as speech recognition and machine translation, but was
first introduced to information retrieval and developed into the query likelihood retrieval model by Ponte and Croft in 1998 [63]. The central idea behind the ranking algorithm is that the document whose language model representing the topical content is more likely to generate the given query is considered to be more relevant.

Assume that query terms clearly address the information need, the information need can be characterized by a language model that is considered to generate the whole relevant set. Given a query \( q \), we would like to rank a retrieved document \( d \) based on the probability that query language model generates this document \( d \). Through Bayes’ law, we have

\[
P(d|q) = P(q|d)P(d)
\]

where \( P(d) \) is the document prior, and can be affected by different document properties, i.e. document length and document category. However, in our general framework we consider it is uniform, so we can rank documents simply by \( P(q|d) \).

For a multiple-term query and a unigram language model,

\[
P(q|d) = \prod_{i=1}^{n} P(q_i|d)
\]

\[
= \prod_{i=1}^{n} \frac{f_i}{|d|}
\]

where \( q_i \) is the \( i^{th} \) query term, and \( f_i \) is the number of its occurrences in document \( d \). \( n \) is the number of terms in the query. \( |d| \) is the document length. For a multinomial distribution, this is the maximum likelihood estimate, which is the estimate making the observed value of \( f_i \) most likely. The product over different query terms raises a problem that if a query term does not appear in the document then the product would yield zero. This is not appropriate since a language model for a document topical content should contain the probability of any word associated with the topic not only those words mentioned in the document.

To address the zero probability issue, a smoothing technique is used to lower the probability estimates for words that occur in the document and to give the remaining probability to the words that do not appear in the document. For those unseen words, we use a background probability \( P(q_i|C) \) computed based on the occurrence frequency of the word in the entire document collection \( C \). Hence, in order that summation of all probabilities still equals one, the probability estimate for a word seen in the document is \((1 - \lambda)P(q_i|d) + \lambda P(q_i|C)\). \( \lambda \) is the smoothing parameter, and determines how much probability is assigned to unseen words.
This is also known as the Jelinek-Mercer smoothing method. After applying the logarithm, we have the scoring function for a document \( d \) as

\[
\log P(q|d) = \sum_{i=1}^{n} \log P(q_i|d) = \sum_{i=1}^{n} \log \left( (1 - \lambda) \frac{f_i}{|d|} + \lambda \frac{c_i}{|C|} \right)
\]  

Based on TREC experiments, \( \lambda \) around 0.1 works well for short queries, whereas around 0.7 is better for much longer queries. Experiments also show it is at least as effective as BM25 ranking algorithm [39].

Another popular smoothing technique is Dirichlet smoothing whose smoothing parameter is dependent on the document length, where \( \lambda = \frac{\mu}{|D| + \mu} \). Now, the scoring function becomes

\[
\log P(q|d) = \sum_{i=1}^{n} \log \frac{f_i + \mu \frac{c_i}{|C|}}{|D| + \mu}
\]  

\( \mu \) determines the importance of relative weighting for words. Smaller \( \mu \) favors the number of matching query terms. TREC experiments show Dirichlet yields better performance than Jelinek-Mercer especially for the short queries when \( \mu \) is between 1000 and 2000 [39].

**Learning-to-rank model**

There are two categories of retrieval models. One is the *generative* model such as probabilistic model and language model, which is based on the assumption that documents are generated by certain underlying model, and training data is used to find appropriate empirical model parameters. The other category is the *discriminative* model. It directly learns a prediction model from the training data. Learning-to-rank model belongs to the later category.

Many machine learning algorithms have been adapted for the ranking purpose from simple regression to sophisticated neural networks, support vector machine (SVM) and boosting [32, 54, 48]. Training data for learning-to-rank algorithms consist of a set of labeled query-document feature vectors. The following is a document feature vector example for document \( d \) given a query \( q \): \( (f_1, f_2, \cdots, f_k) \). Features can be various relevance scores, query term statistics, PageRank or other document and query related properties. Training data also includes a judged label for each query-document feature vector indicating the relevance. Learning-to-rank algorithm learns a ranking model from the training data according to a predefined
objective function, and uses it to predict the relevance and to rank the documents for the new query.

The work in this thesis is based on the generative model, so we only give a very brief introduction to learning-to-rank models here.

2.2 Evaluation

Evaluation is another critical step in the search engine development cycle. To measure the actual performance of a search engine, researchers have developed various testbeds and metrics to evaluate the effectiveness of the retrieval model and the efficiency of the retrieval system. Effectiveness measure focuses on the system’s ability of identifying a relevant document and retrieving all of them. On the other hand, efficiency is about how quickly the system can find relevant documents and return them to the user. The evaluation metrics in this thesis are related to the effectiveness of the retrieve model, since we are investigating the theoretical properties of the retrieval model.

2.2.1 Test collection

In order to fairly and scientifically compare the performances of difference retrieval algorithms, a standard test collection is necessary to ensure that all experiments can be repeatable. Cranfield experiment [35] is the most popular evaluation paradigm in information retrieval research. It makes three basic assumptions [85]: (1) the relevance of a document is independent on the relevance of other documents, and the user information need is static during a search activity. (2) a single judge can represent the entire user population; (3) the relevance judgments are complete. Even though these assumptions are not true in general, they define a laboratory type of re-producible experiments. This evaluation paradigm has been used in annual Text Retrieval Conference (TREC) organized by the National Institute of Standards and Technology. The test collection is designed to include underlying document corpus, query topics, and corresponding relevance judgements.

As this thesis uses TREC collections as the experiment data, we give a brief view of three main components in a test collection.

Document corpus

The document corpus is a set of documents that a retrieval system is built on. For different retrieval tasks, various types of documents are collected to construct a corpus. A document is not necessarily a text file. It can be an image, a webpage, or even a tweet. Document corpora also need to keep updated to reflect the changes of data in the real world.
In this thesis our experiments are based on document corpora for TREC 6, 7, 8 ad-hoc tracks, TREC 9, 10 web tracks (ad-hoc tasks) and TREC 12 robust track. The corpus for TREC 6, 7, 8 and 12 consist of documents collected from the following collections [2]:

- Congressional Record of the 103rd Congress (1993): approximately 210000 documents, about 565 MB;
- Foreign Broadcast Information Service (1996): approximately 130000 documents, about 470 MB;

TREC 9 and 10 web tracks use the WT10g document collection. This collection is about 10GB in size, and contains 1.69 million webpages crawled from 11680 servers on the internet [22]. There are an average of 144 documents, a minimum of 5 documents per sever, and a total of 171740 inter-server links, 1.30 million documents with out-links, and 1.53 million documents with in-links [22].

**Query topics**

The query topic is the search task designed to represent a real-world search activity. It has title, desc, and narr three components. Title is similar to the query key words, and desc and narr describe the details of the information need.

In our experiment we use TREC topics 301550 for TREC 6-10 tasks, and topics 601650 TREC 12 robust track tasks. The following is an example of the TREC query topic.

```
<top>
<num> Number: 407
<title> poaching, wildlife preserves
<desc> Description: What is the impact of poaching on the world’s various wildlife preserves?
<narr> Narrative: A relevant document must discuss poaching in wildlife preserves, not in the wild itself. Also deemed relevant is evidence of preventive measures being taken by local authorities.
```
Relevance judgements

For each query, all retrieved documents need to be assessed to evaluate the retrieval performance. In practice, assessors are the users themselves who actually submit the query. They examine the ranked list from the top to the bottom one by one. Meanwhile, they determine if the document meets their interests. For a relevant document, they will click the link to read more about the article, otherwise, they will just skip it and examine the next one.

However, for those simulated topics in TREC experiments, judges are often trained professionals who have been instructed in how to determined the relevance of the documents for a search. Because relevance judgements to some extent rely on the judge’s interpretation of the information need contained in the query, they are subjective. Different judges may have different opinions about the relevance, so multiple judges are used to assure the agreement. However, TREC experiments show relative performance of systems are very stable, and the difference existing in the relevance judgements does not have a significant impact on the error rate for comparisons [39].

Meanwhile, the number of documents judged to infer the complete relevant document set and the type of relevance judgments will depend on the evaluation metric that is chosen. Relevance can be binary: relevant and non-relevant; or multi-graded, e.g. highly relevant, fairly relevant, marginally relevant, and non-relevant.

2.2.2 Evaluation metrics

To compare the search results obtained by different systems, many evaluation metrics have been developed over years. Here we will cover two of the most popular evaluation metrics in information retrieval.

Precision and recall

Precision and recall, two statistics have been widely used in pattern recognition and information retrieval for a very long time. Precision measures how well a retrieval system is doing at identifying a relevant document, which is the fraction of retrieved documents that are relevant. Recall measures how well a retrieval system is doing at finding all the relevant documents in the collection, which is the fraction of relevant documents that are retrieved.

With the assumption of binary relevance, let $R$ be the set of relevant documents for a query, and $T$ be the set of retrieved documents. $\overline{R}$ is the set of non-relevant
documents, and $\overline{T}$ is the set of non-retrieved documents. Therefore, all possible search result outcomes are categorized in Table 1.

Based on the notation used in the table, precision and recall can be defined as

\[
\text{Precision} = \frac{|R \cap T|}{|T|} \tag{22}
\]

\[
\text{Recall} = \frac{|R \cap T|}{|R|} \tag{23}
\]

Apart from precision and recall, other evaluation measurements can be also derived from Table 1. For instance, fallout is the fraction of non-relevant documents that are retrieved, and F measure is a single value for a system by trading off precision and recall.

\[
\text{Fallout} = \frac{|\overline{R} \cap T|}{|\overline{R}|} \tag{24}
\]

\[
F = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{25}
\]

**Precision-Recall curve**

When we evaluate the ranked list returned by a system, we would like to combine previous introduced two effectiveness metrics precision and recall that both range from 0 to 1. Precision-recall (PR) curve is another popular approach to visualize the overall performance of a retrieval system. It allows us to measure the precision at each recall point for a ranked list.

In a PR curve, precision usually starts at 1 and ends at 0. Recall starts at 0 and ends at 1. There is a trade-off between precision and recall. Recall increases when a relevant document is found. If we retrieve all documents in the collection, we can achieve recall 1, but precision will be almost 0. An actual precision-recall curve has a distinctive saw-tooth shape: if a non-relevant document is retrieved then recall will keep the same, but precision will drop. If a relevant one is retrieved, then both

\[
\begin{array}{|c|c|c|}
\hline
 & \text{Relevant} & \text{Non-Relevant} \\
\hline
\text{Retrieved} & R \cap T & \overline{R} \cap T \\
\text{Not Retrieved} & R \cap T & \overline{R} \cap T \\
\hline
\end{array}
\]

Table 1: Four possible categories for a document in the search results
precision and recall increase, and the curve will jump up and to the right. It is useful to have interpolated version of the precision-recall curve. For each precision $p$ at recall $r$: $p(r)$, the interpolated precision $p_{\text{interp}}$ is defined as:

$$p_{\text{interp}}(r) = \max_{r' \geq r} p(r')$$

(26)

For instance, we have the following retrieved list and the corresponding precision recall value at each rank (see Table 2). The actual PR curve (blue dot line) and its interpolated version (red line) based on this example are illustrated in Figure 5.

**Table 2:** An example of a ranked list with the corresponding precision and recall values. There are 5 relevant documents in total, and all of them are retrieved at rank 10.

<table>
<thead>
<tr>
<th>Rank</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevance</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Recall</td>
<td>1/5</td>
<td>2/5</td>
<td>3/5</td>
<td>3/5</td>
<td>3/5</td>
<td>4/5</td>
<td>4/5</td>
<td>5/5</td>
<td>5/5</td>
<td>5/5</td>
</tr>
<tr>
<td>Precision</td>
<td>1/1</td>
<td>2/2</td>
<td>3/3</td>
<td>3/4</td>
<td>3/5</td>
<td>4/6</td>
<td>4/7</td>
<td>5/8</td>
<td>5/9</td>
<td>5/10</td>
</tr>
</tbody>
</table>

**Figure 5:** Actual PR curve and its interpolated version for the previous ranked list example

**Average Precision**

The precision-Recall curve is a good way to visualize the overall retrieval performance of a system. However, a single number measurement is preferred to reflect
the quality of the retrieval performance especially for large-scale evaluations over many queries and systems. Average Precision (AP) is such a metric that averages the precisions at all relevant ranks. It is approximately the area under the precision-recall curve, and is computed as follows:

\[
AP = \frac{\sum_{i=1}^{N} p(i) \cdot rel(i)}{R}
\]  

(27)

where \(N\) is the number of retrieved documents, and \(rel(i)\) is the relevance function for the document at rank \(i\). If document is relevant, \(rel(i) = 1\), otherwise \(rel(i) = 0\). \(R\) is the number of relevant documents. For the example in Table 2, average precision for this query is computed as:

\[
AP = \frac{1 + 1 + 1 + 4/6 + 5/8}{5} = 0.86
\]

In addition to the average precision, \(F\) measure and \(R\)-precision are other two single point measures. \(F\) measure is the harmonic mean of the precision and recall. \(R\)-precision is the precision at rank \(R\). \(R\) is the number of relevant documents, and is highly correlated to average precision [17, 49, 86].

**Normalized Discounted Cumulative Gain**

Normalized Discounted Cumulative Gain (nDCG) is an evaluation metric that focuses on the top of the ranked list. It is very popular for evaluating web search quality [53], and works for multi-graded relevance as well. It is based on two assumptions:

- Highly relevant documents are more useful than marginally relevant documents.
- The relevant document at lower position is less useful for the user since it is less likely to be examined.

Discounted Cumulative Gain (DCG) at rank \(i\) is defined as

\[
DCG_i = rel(1) + \sum_{k=2}^{i} \frac{rel(k)}{\log_2 k}
\]  

(28)

where \(rel(i)\) is the relevance function for the document at rank \(i\), and the relevance can be multi-graded. With the ideal DCG (IDCG) for the query, normalized DCG at rank \(i\) can be computed as

\[
nDCG_i = \frac{DCG_i}{IDCG_i}
\]  

(29)
2.2.3 Other evaluation methods

There are also other popular evaluation experiment methods such as crowdsourcing and log analysis. Crowdsourcing is a distributed problem-solving and evaluation model. Relevance judging tasks are broadcast on public platform like Amazon Mechanical Turk [7] to an unknown group of assessors in the form of an open call for the document assessment. Users, also known as the crowd, typically form into online communities, and the crowd submits their judgements.

Log analysis attempts to infer the user’s satisfaction through their interactions with the search engine. From the query log, the clickthrough data, page dwell time, and search exit action can be used to measure the search quality.

2.3 Score Distributions

For a given query, a search engine assigns a score to each document in the collection according to some retrieval model, and ranks documents based on their scores. A document with a higher score is considered to contain more relevant information as compared to the one with a lower score.

This scoring mechanism is problematic for many retrieval applications such as data fusion tasks which need to combine several ranked lists retrieved from different search engines, domains, or severs, and the recall-oriented retrieval which requires estimating the number of relevant documents under certain score threshold. With the assumption of the binary relevance, two distinctive distributions can be used to model relevant and non-relevant document scores separately. Therefore, we would be able to infer the probability of the document relevance through two distributions, and use this probability to solve above-mentioned IR problems.

Numerous combinations of statistical distributions have been proposed in the literature to model the score distributions of relevant and non-relevant documents, such as two Gaussians of equal variance [82], two Gaussians of unequal variance [83], two Poisson distributions [30], two Gamma distributions [25], and a Gaussian distribution for relevant documents with an exponential distribution for non-relevant documents [58]. We will give a background introduction to each statistical distribution used in the previous work here.

In this section, we only discuss the mathematical background for modeling the document scores, and given an overview of several statistical. More details regarding the related works will be covered in the later chapters.
2.3.1 Gaussian distribution

The Gaussian distribution (a.k.a: normal distribution) is the most well known continuous probability distribution with a bell-shape probability density function. It naturally approximates a random variable whose values cluster around a single mean value, and arises from the summation of a large number of random variables according to the central limit theory [23]. Its probability density function is:

$$N(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$  \hspace{1cm} (30)

where $\mu$ is the mean, and $\sigma^2$ is the variance. The maximum likelihood estimation for these two parameters are as follows [59]:

$$\hat{\mu}_{MLE} = \frac{1}{n} \sum_{i=1}^{n} x_i$$  \hspace{1cm} (31)

$$\hat{\sigma}^2_{MLE} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$$  \hspace{1cm} (32)

2.3.2 Poisson distribution

The Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed time period with a known average rate of the event occurrence. The probability density function is:

$$\text{Poisson}(k) = \frac{\lambda^k}{k!} \cdot e^{-\lambda}$$  \hspace{1cm} (33)

where $\lambda$ is the average frequency of the event. The mean and variance both equal to $\lambda$, and the maximum likelihood estimation for $\lambda$ is

$$\hat{\lambda}_{MLE} = \frac{1}{n} \sum_{i=1}^{n} k_i$$  \hspace{1cm} (34)

The Poisson distribution is based on two main assumptions: (1) the probability of an event in one interval is independent of the probability of an event in other non-overlapping intervals; (2) the probability of an event within a certain interval does not change over different intervals.

2.3.3 Exponential distribution

When events are generated by a Poisson process, the time between each pair of consecutive events can be modeled by an exponential distribution. The probability
density function of exponential distribution is

\[ \text{Exp}(x) = \begin{cases} 
\lambda e^{-\lambda x}, & x \geq 0 \\
0, & x < 0 
\end{cases} \]  
(35)

where \( \lambda \) is the rate parameter, which is the average occurrence rate of the events generated by the Poisson process. The maximum likelihood estimation for \( \lambda \) is

\[ \hat{\lambda}_{\text{MLE}} = \frac{n}{\sum_{i=1}^{n} x_i} \]  
(36)

### 2.3.4 Gamma distribution

The Gamma distribution is also a continuous probability distribution with a scale parameter \( \theta \) and a shape parameter \( k \). If \( k \) is an integer, the random variable based on a Gamma distribution is the summation of \( k \) independent exponentially distributed random variables. When \( k \) is large, the Gamma distribution converges to a Gaussian distribution. The probability density function of Gamma distribution is

\[ \text{Gamma}(x) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-\frac{x}{\theta}} \]  
(37)

The maximum likelihood estimation for the scale parameter \( \theta \) is:

\[ \hat{\theta}_{\text{MLE}} = \frac{1}{kn} \sum_{i=1}^{n} x_i, \]  
(38)

and the estimation for the shape parameter \( k \) can be computed iteratively through the Newton-Raphson Method [34]. The update equation is

\[ \hat{k}_{\text{MLE}} \leftarrow \hat{k}_{\text{MLE}} - \frac{\ln k - \psi(k) - s}{\frac{1}{k} - \psi'(k)} \]  
(39)

where \( \psi' \) denotes the trigamma function, and is the derivative of the digamma function.
Chapter 3

Score Distribution Model

Part 1: An Empirical Model

When a user submits a query to an information retrieval system, the system computes a score for each document in the collection according to some definition of relevance to the user’s request and returns a ranked list of documents. In reality, this ranked list of documents is a mixture of both relevant and non-relevant documents. For a wide range of retrieval applications (e.g. information filtering, topic detection, meta-search, distributed IR), modeling and inferring the distribution of relevant and non-relevant documents over scores in a reasonable way could be highly beneficial. For instance, in information filtering, topic detection, and recall-orientated retrieval, score distributions can be utilized to find the appropriate threshold between relevant and non-relevant documents [82, 83, 14, 91, 37, 80]. In distributed IR and meta-search it can be used to normalize document scores and combine different collections or the outputs of several search engines [25, 58].

Under the assumption of binary relevance, numerous combinations of statistical distributions have been proposed in the literature to model the scores of relevant and non-relevant documents, such as two Gaussians of equal variance [82], two Gaussians of unequal variance [83], two Poisson distributions [30], and two Gamma distributions [25]. The most popular model has been a Gaussian distribution for relevant documents and an exponential distribution for non-relevant documents [58], and this model has been widely used in practical applications [58, 91, 10, 11]. The most recently proposed model is a truncated version of Gaussian-exponential model [11]. The strong argument in all studies for choosing any particular combination of distributions is the goodness-of-fit to a set of empirical data. However, there is no real consensus on the choice of the distributions due to the
complexity of the underlying process in the retrieval model that generates document scores.

This chapter is an extension of our work in [42, 56]. We use the available relevance information to study what is the appropriate choice of distributions to model scores for relevant and non-relevant documents from the empirical data. We revisit the choice of distributions used to model document scores, and propose a more flexible model that empirically fits a Gamma distribution in the scores of the non-relevant documents and a mixture of Gaussians in the scores of the relevant documents. Experiments on TREC data show that the new model outperforms the dominant Gaussian-exponential model, and demonstrates its utility in inferring precision-recall curves.

3.1 Related Work

Inferring the score distributions of relevant and non-relevant document is an essential task for many retrieval applications. Modeling score distributions in an accurate manner is often the basis of any possible inference. Due to this, numerous combinations of statistical distributions have been proposed in the literature to model scores of relevant and non-relevant documents. Most of them are based on certain assumptions and justified in terms of the goodness-of-fit to the empirical data, but few theoretical arguments are provided.

Early score distribution models

The earliest score distribution models in the literature are found in the works by Swets [82, 83]. He proposed to use two distributions to model scores of relevant documents and non-relevant documents separately. Based on that, he developed a new measure of retrieval effectiveness through statistical-decision theory. Swets argued that the separation between the areas under two probability density functions of score distributions indicated the system’s capacity to distinguish the relevant and non-relevant documents. In terms of the choice of the distributions, he experimented two Gaussians of equal variance, two Gaussians of unequal variance, and two exponentials for relevant scores and non-relevant scores respectively to derive the evaluation measure in [83]. For two Gaussians with equal variance, the evaluation measure $E$ is defined as the difference between the means of two density functions divided by the common standard deviation. For two Gaussians with unequal variance, $E$ was adopted by normalizing the difference between two means by their average standard deviation [83]. For the two-exponential model, $E$ can be described by a single parameter $k$, which is the only parameter for exponential probability density function. In fact, recent research [74] and [55] found
score distributions and information retrieval evaluation metrics such as precision-recall curve are highly correlated. Given one of them, the other can be derived through a mathematical framework. We will discuss this connection in more detail in Chapter 4.

Swets’ works proposed several combinations of statistical distributions to model document scores. However, all of them were based on certain hypothesis since the goal of his works were to design a new retrieval effectiveness metric. Therefore, the choice of score distributions lacked theoretical and empirical justifications. van Rijsbergen [67] also commented that there was no evidence that the scores produced by search engines like SMART for relevant and non-relevant documents were similarly normally distributed.

Later Bookstein pointed out in [30] that if a system follows Swets’ two-Gaussian model and uses statistical decision theory as the retrieval role, the most relevant documents are retrieved only when the standard deviations are equal. Meanwhile, the system might retrieve the least relevant documents and ignore those extremely relevant documents when the standard deviations are not equal in two Gaussian distribution, and in practice the standard deviations of two Gaussians were indeed different [83]. Instead, given that (1) most information retrieval systems (at that time) produce positive scores; (2) the random variable taking integral or discrete values should be favored; (3) the probability function of the new distribution should be similar to a normal function, Bookstein proposed to use Poisson distributions to model relevant and non-relevant scores [29, 30].

In [29] Bookstein and Swanson have developed a model for word occurrences, which estimates the document relevance based on the pattern of the query word occurrence within the document. Because a set of words relevant to the document topic has been observed to be distributed differently than those words are not relevant to the topic in a document\(^1\). Moreover, it is well known that word occurrence in a document is governed by two different Poisson processes, one for relevant words and one for the non-relevant ones [44, 81, 50]. If further assume that the relevance score is based on the number of occurrences of those related words, two Poisson distributions can be used to model scores of relevant and non-relevant documents.

Bennett in his work [26] observed that when using a two-Gaussian model for text classification, document scores outside the modes of the two Gaussians (corresponding to “extremely irrelevant” and “obviously relevant” documents) demonstrate\(^2\).

\(^1\)In the original paper, a word relevant to the document topic is defined as the word that could convey the subject meaning of the document.
strated different empirical behavior than the scores between the two models (cor-
responding to “hard to discriminate” documents). Data in the area between the 
modes corresponds to the hard example, and is very difficult for a classifier to dis-
tinguish. This motivated him to introduce several asymmetric distributions (asym-
nmetric Gaussian and asymmetric Laplace) to uncouple the scale of the outside and 
the inside of the distribution.

In [25], Baumgarten proposed a probabilistic model for optimal information 
retrieval over a distributed document collection, and two shifted gamma distributions 
were used for scores of relevant and non-relevant documents. The intuition 
was straightforward: the document score distribution was close to normally dis-
tributed if we assume the number of query features to be very large according to 
the central limit theorem [24]. However, the assumption often did not hold in re-
ality. In the high value score interval, the probability density function of the score 
distribution demonstrates some properties of a normal distribution. On the other 
end, the probability density function forms a pike when the smallest possible score 
has been reached, and becomes zero for scores smaller than that value. Based on 
those facts, Baumgarten suggested that a more flexible distribution such as Gamma 
distribution should be considered.

The Gaussian-exponential model

The current most popular score distribution model is the Gaussian-exponential 
model, which uses a Gaussian distribution to model relevant scores, and exponen-
tial distribution for non-relevant ones. This model was first proposed by Aram-
patzis et al. in [8] and followed up by a series of works [58, 91, 36]. Arampatzis and 
van Hameren in their earlier work [14] claimed that a Gaussian limit appears in 
the score distribution for the relevant documents with increasing query length, but 
for non-relevant documents, a Gaussian limit was unlikely and if it appears it does 
with a very slow query length increasing rate. Their empirical experiments also 
pointed out that the right tail of the distribution of the non-relevant documents’ 
scores could be very well approximated by an exponential distribution. The three 
explicit assumptions for using a Gaussian distribution to model relevant scores are 
进一步 discussed in Arampatzis and Robertson’s recent work [12], and we quote 
here:

1. Document terms occur independently.
2. Relevance score is computed as the linear combination of the weights of 
query terms
3. Relevant documents cluster around certain point in the document space with
some hyper-ellipsoidal density.

Authors argued that assumption 2 and 3 are reasonable and hold for many retrieval models. Meanwhile, assumption 1 is generally untrue, but independence assumption is common in probabilistic models, and ranking algorithms derived from it work very robust in reality.

In another work by Manmatha et al. [58], they extended Harter’s mixture of two Poisson distributions [50] under the assumption that the score given to a document can be reasonably approximated as being proportional to the number of matching words in the document. For the set of relevant documents, the parameter $\lambda$ in the Poisson distribution is usually large, and the Poisson distribution with a large $\lambda$ tends to a Gaussian distribution. On the other hand, $\lambda$ is small for non-relevant documents, and the Poisson distribution tends to exponential distribution since it falls rapidly in the right tail. Thus, they proposed to use a Gaussian distribution to model relevant scores and an exponential distribution to model non-relevant scores.

In a different line of work, Robertson proposed that the score distribution should follow the convexity property [74]. Based on this hypothesis, Arampatzis et al. [11] proposed a truncated version of the Gaussian-exponential model, which overcomes the theoretical problems associated with the original Gaussian-exponential model. We will discuss in further detail about the theoretical properties of score distributions in next chapter.

### 3.2 Methodology

#### 3.2.1 Motivation

In the rest of this chapter we will essentially let the data itself dictate how to model score distributions. In this section however we give an intuitive explanation for our choice of a richer set of density functions for this purpose. A theoretical analysis on the subject can be found in [55]. In their work, Kanoulas et al. derive the score distributions of non-relevant and relevant documents based on rudimentary assumptions about the distribution of terms in documents and the behavior of good retrieval systems. The Gamma distribution appears to be a good approximation of the theoretically derived distribution for the non-relevant documents. A Gaussian-like distribution with a heavy right tail, which could essentially be modeled by a mixture of two Gaussians, seems to be the appropriate model under some reasonably assumptions. Some more intuitive arguments follow.
Figure 6: The distribution of BM25 scores for all 113,947 documents (containing at least one query term) on query “Ireland peace talks”. Note the different slopes at the left and at the right of the mean. Truncating the list at rank 1,000 would cause the scores’ distribution to look like an exponential one. Histograms of BM25 computation are shown step-by-step starting with TF for individual query terms(top) and ending the BM25 final score(bottom).
Non-relevant documents score distribution

Previous work [58, 74] argues that the score distribution of non-relevant documents can be well approximated by a negative exponential density function. Often, however, a more flexible distribution is necessary. The Gamma distribution, which can range (in skewness) from an exponential to a Gaussian distribution is flexible enough. In order to explain why a Gamma distribution is a better choice, several factors should be considered.

- Truncation at a cut-off: If a list is arbitrarily truncated very early (say at rank 1,000) the distribution of the top scores may indeed look like an exponential. However looking deep down in the list (say up to rank 200,000), the shape of score distribution changes (Figure 6, bottom).

- Query complexity: Arguments for the score distribution for a single term queries have been given in the literature [58]. For a query with two or more terms, most non-trivial documents (i.e. the ones that contain at least two query terms) will have the following property: the contribution of the two or more terms to the final score of a document would often times be very different for the two or more terms, with some terms having a low contribution while others having a higher contribution. Averaging such effects is likely to produce a “hill” of score frequencies, perhaps with different slopes at the left and the right side of the mean; the Gamma distribution is known to be an average of exponential distributions.

- Inversion of term-frequency (TF): many scoring functions contain fractions with TF in the denominator (for example BM25, Robertson-TF etc). Given that the TF values are most of the time distributed zipfian-like, such inversion will likely produce “hill” histograms. Figure 6 shows the histograms of BM25 computation step by step for each query term followed by the final score; it can be observed that the “hill” appears when TF inversion takes place.

- Retrieval function: We mostly look at scoring functions that are decomposable into a sum of scores per query terms, like TF-IDF or Language Models (after taking logs); such scores also induce averaging effects(Figure 6).

Relevant documents score distribution

The Gaussian density function has been the most widely used one to model the score distribution of relevant documents. However, due to its simplicity it often places unreasonable limitations over the fitting process. An example of a single
Figure 7: The histogram over the scores of non-relevant and relevant documents along with the best fit exponential and Gaussian distributions (top plot) the Gamma and \( k \)-Gaussians distribution (bottom plot).

Gaussian density function failing to capture the distribution of relevant documents can be viewed in the upper plot of Figure 7. The figure shows the histogram over the scaled scores of relevant (thin red bars) and non-relevant (wide yellow bars) document for the TREC 8 query “Estonia economy”. In the top plot a negative exponential and a single Gaussian density functions are separately fit into the scores, while in the bottom plot shows a Gamma density function and a mixture of two Gaussians are fit into the scores. As one can observe there are two clusters of relevant documents, one centered around score 0.3 and another centered around score 0.7. A single Gaussian fails to capture these two separated masses. On the contrary, it underestimates documents from two masses and overestimates documents with scores in the middle range, which in fact are less likely to be relevant. This leads to an incorrect prediction of relevance probability for a given score, especially for the low score mass that has the largest support of relevant documents. This very same phenomenon led [26] to skew the single Gaussian distribution towards the low score relevant documents so that it does not underestimated the probability of being relevant.
An intuition behind the shape of the distribution that models the scores of relevant documents is given by [58]. Assuming that a query consists of a single term, Manmatha showed that the scores of relevant documents can be modeled as a Poisson distribution with a large $\lambda$ parameter, which approaches a Gaussian distribution. Now, let’s consider queries that consist of multiple terms and let’s revisit Figure 7. The query used in the example is: “Estonia economy”. Each relevant document in the plot corresponds either to a triangular or to a rectangular marker at the top of the plot. The triangular markers denote the relevant documents for which only one out of the two query terms occur in the document, while the rectangular ones denote the relevant documents for which both terms occur in the document. By visual inspection, the relevant documents containing a single term clearly correspond to the low-scores’ Gaussian, while the relevant documents containing both terms clearly correspond to the high-scores’ Gaussian (bottom plot). Essentially, the former documents get a low score due to the fact that only one terms appear in them but they happen to be relevant to the query, while the latter correspond to documents that are obviously relevant.

We observed the same phenomenon for many different queries independently of the IR model used for retrieval and independent of the query formulation. In the case of queries with multiple terms (e.g. queries that consists of both the title and the description), even though the possible number of query terms that may co-occur in a document is greater than 2 (e.g. for a query with 3 terms, all terms may occur in a document or only two of them or only a single one of them), we observed that there is a threshold on the number of terms occurring in the document; relevant documents containing a number of terms that is less than this threshold are clustered towards low scores (first Gaussian), while relevant documents containing a number of terms that is greater than the threshold are clustered towards high scores (second Gaussian).

3.2.2 Variational Bayesian inference

The Gamma distribution is used to model the scores of the non-relevant documents. The Gamma density function with scale $\theta$ and shape $M$ is given by,

$$P(x|M, \theta) = x^{M-1} \frac{\exp^{-x/\theta}}{\theta^M \Gamma(M)} \quad \text{for } x > 0 \text{ and } M, \theta > 0$$

(40)

where, $\Gamma(M)$ is an extension of the factorial function to real numbers, while for positive integer $M$, $\Gamma(M) = (M - 1)!$. The mean of the distribution is $M\theta$, while the variance is $M\theta^2$. The maximum likelihood (ML) estimation is used to estimate
the Gamma parameters. When $M = 1$, the Gamma distribution degrades to an exponential distribution with rate parameter $1/\theta$, and for large $M$, the Gamma distribution converges to Gaussian distribution with mean $\mu = M\theta$ and variance $\sigma^2 = M\theta^2$.

The scores of relevant documents are modeled by a mixture of $K$ Gaussians

$$P(x|\pi, \mu, \Lambda) = \sum_{i=1}^{K} \pi_i N(x|\mu_i, \Lambda_i^{-1})$$

(41)

where $\pi_i$ is the mixing coefficient for the $i^{th}$ Gaussian component, and satisfies $0 \leq \pi_i \leq 1$ and $\sum_{i=1}^{K} \pi_i = 1$. $N(x|\mu_i, \Lambda_i^{-1})$ is Gaussian probability density function with parameters $\mu_i$ and $\Lambda_i$, the mean and the precision of the Gaussian components, respectively. The mixture coefficient $\pi_i$ essentially expresses the contribution of each Gaussian to the mixture.

Fitting a mixture of Gaussians into scores could be easily done by employing the Expectation Maximization (EM) algorithm if the number of Gaussian components $K$ is known. However, we assume that we only know an upper bound on $K$. Given the fact that the larger the number of components the better the fit and that EM finds the maximum likelihood mixture of Gaussians regardless of the model complexity, the cross validation is necessary to find the best $K$. However, for most queries, there are only a limited number of relevant documents, so the EM algorithm is not appropriate for our problem. Instead, to avoid over-fitting, we employ a Bayesian treatment on the model by utilizing the Variational Bayes (VB) framework [28, 19, 20].

The VB framework takes a fully Bayesian treatment of the mixture modeling problem by introducing prior distributions over all the parameters of the model, i.e. $\pi = \{\pi_i\}$, $\mu = \{\mu_i\}$ and $\Lambda = \{\Lambda_i\}$ and thus accounting for the uncertainty of the value of these parameters. Given a fixed number of potential components (an upper bound on $K$) the variational inference approach causes the mixing coefficients of unwanted components to go to zero and essentially leads to an automatic trade-off between the goodness-of-fit and the complexity of the model.

To give some insight into how VB trades the goodness-of-fit with the complexity of the model, let’s consider the function that VB optimizes, in the general case. Given a set of variables, $X$, and a set of parameters, $\Theta$, Variational Bayes aims at optimizing the log of the marginal likelihood or evidence, $p(X)$, where the hidden
variables along with the parameters have been integrated out. That is,

$$\log p(X) = \log \int p(X, \Theta) d\Theta = \log \int q(\Theta | X) \frac{p(X, \Theta)}{q(\Theta | X)} d\Theta$$  \hspace{1cm} (42)

$$\geq \mathcal{L} \equiv \int q(\Theta | X) \log \frac{p(X, \Theta)}{q(\Theta | X)} d\Theta \text{ by Jensen’s Inequality}$$  \hspace{1cm} (43)

Inequality 43 holds for any arbitrary conditional distribution $q$. The difference between the l.h.s and the r.h.s of the inequality is in fact the KL-divergence between the arbitrary conditional distribution $q$ and the posterior distribution $p(\Theta | X)$, and thus the optimal $q$ is obtained by letting $q = p(\Theta | X)$. Given the above inequality, VB actually optimizes the lower bound $\mathcal{L}$ instead of the log marginal likelihood. If we further expand the lower bound $\mathcal{L}$ we obtain,

$$\mathcal{L} = \int q(\Theta | X) \log p(X | \Theta) d\Theta$$  \hspace{1cm} (44)

$$= \int q(\Theta | X) \log \frac{p(X | \Theta) p(\Theta)}{q(\Theta | X)} d\Theta$$  \hspace{1cm} (45)

$$= \int q(\Theta | X) \log p(X | \Theta) d\Theta - \int q(\Theta | X) \log \frac{q(\Theta | X)}{p(\Theta)} d\Theta$$  \hspace{1cm} (46)

$$= \int q(\Theta | X) \log p(X | \Theta) d\Theta - KL[q(\Theta | X) || p(\Theta)]$$  \hspace{1cm} (47)

where $p(\Theta)$ is the prior distribution over the model parameters and $KL[q(\Theta | X) || p(\Theta)]$ is the KL-divergence between the posterior distribution of the model parameters and their prior distribution. The left term of the r.h.s. of Equation 47 expresses the goodness-of-fit of the model to the data and increases with the complexity of the model, while the right term of the r.h.s. of Equation 47 is the Occam factor which penalizes over-complex models. Essentially, VB penalizes the departure of the parameters from their prior distribution. Finally, note that the Bayesian information criterion (BIC) [78] and the minimum description length criterion (MDL) [68] both emerge as a special case of a large sample expression of Equation 47 [19].

Moving back to the mixture of Gaussians, we introduced priors over the parameters $\pi$, $\mu$, and $\Lambda$. To simplify the mathematics of VB and achieve an analytic solution, we only consider conjugate prior distributions (as in [28]), such that the posterior distribution of the model parameters given the data is in the same family of distributions with the prior. Thus, we chose a Dirichlet distribution over the mixing coefficients $\pi$, i.e. $p(\pi) = \text{Dir}(\pi | \alpha_0)$, and an independent Gaussian-Wishart distribution over the mean and the precision of each Gaussian component, i.e. $p(\mu, \Lambda) = \mathcal{N}(\mu | m_0, (\beta_0 \Lambda)^{-1}) \mathcal{W}(\Lambda | W_0, v_0)$. Regarding the Dirichlet distribution,
by symmetry we chose the same parameter $\alpha_0$ for all the mixture components. Given that $\alpha_0$ can be interpreted as the effective prior number of observations associated with each component of the mixture we set $\alpha_0 = 10^{-3}$, such that the posterior distribution will be influenced primarily by the data. Regarding the Gaussian distribution, $m_0$ corresponds to the mean value of the distribution of the Gaussian means, thus we assigned the same value $m_0$ to all the mixture components and set it to the mean of the data, i.e. the mean score of the relevant documents. Regarding the Wishart distribution, $v_0$ corresponds to the dimensionality of the data which in our case is 1 and thus we set $v_0$ equal to 1. The hyperparameter $W_0$, in the general case, is a positive definite matrix, however in the case of one-dimensional data $W_0$ is simply a number, which were initialized by the precision of the data. The parameter $\beta_0$ is a scalar corresponding to the ratio between the precision of the distribution of the Gaussian mixture means and the precision of the distribution of the data. We initialized it by clustering the data into $K$ clusters and setting $\beta_0$ equal to variance of data/variance of cluster means.

To initialize the VB process we first run the k-means algorithm and obtain 10 initial clusters and then use these clusters to initialize the expectation-maximization (EM) algorithm that results in a mixture of 10 Gaussians. This mixture is then used to initialize the VB process that finds the optimal model in terms of goodness-of-fit and model complexity. The same process is run 10 times with 10 different initializations of the k-means algorithm. In the end, we select the model that leads to the highest lower bound $L$.

Remark: In this work we only consider Variational Bayes as a technique to automatically select the number of components of the Gaussian mixture by trading off the goodness-of-fit with the model complexity. However, several other criteria that can lead to achieve the same effect have been proposed in the literature. Schwarz's BIC [78], Akaike's AIC [3], Rissanen's minimum description length, the Information Complexity Criterion [31], the Normalized Entropy Criterion [33] are some of them that have been used along with an EM algorithm to fit a mixture of Gaussians with unknown number of components. Markov Chain Monte Carlo (MCMC) methods have also been used for model selection (e.g. see [66]).

3.3 Experiments

We use data from TREC 6, 7 and 8 ad-hoc tracks, TREC 9 and 10 Web tracks (ad-hoc tasks) and TREC 12 Robust track. TREC 6, 7, 8 and 12 collections consist of documents contained in the TREC Disk 4 and 5, excluding the Congressional Record sub-collection, while TREC 9 and 10 collections use the WT10g document collect-
The topics used are the TREC topics 301 – 550 and 601 – 650 [87]. The Robust track topic set in TREC 12 consists of two subsets of topics, the topics 601 – 650 and 50 old topics selected based on topic hardness from past collections. In all results regarding TREC 12 only the topics 601 – 650 are used.

To avoid the effects of arbitrary query manipulations and score transformations that systems submitted to TREC (Text REtrieval Conference) often apply, in the sections that follow we instead use scores produced by traditional IR models. Later, in Section 3.3.1, we validate our model on TREC systems.

Indexing and search was performed using the Terrier search engine [61]. Porter stemming and stop-wording was applied. The document scores obtained are the outputs of (a) Robertson’s and Spärck Jones’ TF-IDF [71], (b) BM25 [69], (c) Hiemstra’s Language Model (LM) [52], and (d) PL2 divergence from randomness [5] (with Poisson estimation for randomness, Laplace succession for first normalization, and Normalization 2 for term frequency normalization). Further, three different topic formulations were used, (a) topic titles only, (b) topic titles and descriptions, and (c) topic titles, descriptions and narratives.

Finally, note that, by convention, documents not judged by TREC assessors are considered non-relevant, since there were not retrieved by any of the submitted to TREC runs in the top-k ranks, where k is usually 100. When fitting the Gamma distribution we consider these documents as non-relevant and thus we essentially fit the Gamma distribution in both non-relevant and unjudged documents. For the rest of the article by non-relevant documents we refer to both judged non-relevant and unjudged documents.

3.3.1 Results and analysis

We separately fit the Gamma distribution and the mixture of Gaussians into the scores of the non-relevant and relevant documents, respectively, for each topic-system pair. There are 50 topics available per TREC data set and 3 query formulations (title, title and description and title, description and narrative), along with the relevance information for the top 1000 documents returned by 4 IR systems (TF-IDF, BM25, LM and PL2). Thus, there are in total 600 ranked lists of documents per TREC data set. The scores of the documents were first normalized into a 0 to 1 range by shifting and scaling to preserve the score distribution.

To summarize our results we report the parameter $M$ of the Gamma distribution, which as mentioned earlier corresponds to the number of independent exponential density functions averaged, and the number $K$ of Gaussian components in the mixture, for all four systems, all 150 topics (50 topics and 3 query formulations)
Figure 8: The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution, over all IR models, topics and topic formulations for TREC 6, 7, 8.
Figure 9: The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution, over all IR models, topics and topic formulations for TREC 9, 10, 12
for each TREC data set. Figure 8 and Figure 9 show the histograms over $M$ and $K$. Each row corresponds to each one of the TREC 6, 7, 8, 9, 10 and 12 data sets in this order. As it can be observed, $K$ is most of the times different than one, especially in the early TREC collections. Further, $M$ is spread both above and below one. This illustrates that often times, taken into account the complexity of the model, the data suggests that a Gamma distribution and a mixture of Gaussians is a better fit to relevant and non-relevant scores than a negative exponential and a single Gaussian. In particular, the mean number of Gaussian components over all TREC data sets is 1.52 while the mean number of component for each TREC data set separately is 1.75, 1.74, 1.67, 1.11, 1.37, 1.51. The mean value of the parameter $M$ over all TREC data sets is 0.98 while the mean value of the parameter $M$ for each TREC data set separately is 0.92, 0.96, 0.93, 1.10, 1.00, 0.96. Even though the mean $M$ is close to 1, as it can be viewed in Figure 8 and Figure 9, $M$ varies in a wide range below and above 1.

In this section, we attempt to analyze the parameters $K$ (number of Gaussian mixture components) and $M$ (number of independent exponential distributions averaged) to obtain a better understanding of the underlying process that generates this distribution of relevant and non-relevant documents. First, we examine whether different factors such as the IR model and the number of query terms affect the distribution of the parameters $K$ and $M$. Then, we focus on the relevant document score distribution and examine whether the total number of relevant documents retrieved affect our ability to recover complex distributions, such as a mixture of Gaussians. Further, we examine whether the different components of the mixture correspond to relevant documents of different characteristics. In particular, we explore whether (a) different Gaussians explain relevant documents uniquely identified by manual runs and relevant documents retrieved by automatic runs, and (b) different Gaussians correspond to relevant versus highly relevant documents.

**IR systems:** First we test whether and how different IR systems may affect the parameters $M$ and $K$. In Figures 10 – 15 we report the histograms over $K$ and $M$ for each system separately (50 topics with 3 topic formulations) for TREC 6, 7, 8, 9, 10, and 12 data sets. As it can be observed, both the distribution over $K$ and the distribution over $M$ appear to be independent with respect to the IR model utilized. To validate our observations we run an n-way ANOVA testing whether the mean values of $K$ per IR model are equal and we could not reject the hypothesis.

**Number of query terms:** Then, we tested how the number of query terms affect the parameters $M$ and $K$. In Figures 16 – 21 we report the histograms over $K$ and
Figure 10: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 6.

Figure 11: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 7.

Figure 12: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 8.
Figure 13: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 9.

Figure 14: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 10.

Figure 15: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and topic formulations for each IR model for TREC 12.
Figure 16: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 6.

Figure 17: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 7.
Figure 18: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 8.

Figure 19: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 9.
Figure 20: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 10.

Figure 21: The histogram over the number $K$ of Gaussian components and the parameter $M$ of Gamma distribution, over all topics and IR models for each topic formulation, for TREC 12.
$M$ for each different query formulation separately (50 topics with 3 topic formulations) for TREC 6, 7, 8, 9, 10, and 12 data sets. As it can be observed, the distribution over $K$ and $M$ appear to be independent of the number of query terms. The distribution over $M$ appears to be slightly flatter in the case of title-only queries than the rest of the formulations. However, the mean $M$ appears to remain unaffected by the number of query terms.

After we take a close look at the distribution of relevant document scores, a mixture of $K$ Gaussians, we do observe that the shape of the distribution changes when more query terms are selected. In Figure 22, we see that when the query is expanded, we still have two Gaussian components, but the score distribution for the relevant documents becomes more smooth. The second spike in the high score range for title query turns into a flat heavy tail when the narrative is also used in the query. This is very similar to Arampatzis and van Hameren’s finding in [14] that the distribution of relevant document scores converges to a Gaussian central limit when the length of the query increases. In our case, the Gaussian shape appears along with a heavier right tail.

![Figure 22](image)

**Figure 22:** The score distribution of the relevant document for TREC8 topic 401: *foreign minorities, Germany*. Three topic formulations are used: title, title + description, title + description + narrative.

To further analyze the effect of the number of query terms on the values of $K$ and $M$, we performed query expansion on the top of the title query formulation for TREC 6, 7, and 8 data sets, by varying the number of terms to expand a query with from 4 to 512 terms (increasing powers of 2). The true relevance feedback
was used in the experiment, and the additional terms were extracted from 10 relevant documents selected from TREC qrel file. The term weighting model used for expanding the queries with the most informative terms was the Bose-Einstein 1 Method [6] provided by the Terrier Toolkit [61]. We still used BM25 as our retrieval function and the same model inference process. The average values and their standard deviations (shaded area) for the parameters $K$ and $M$ against the number of expanding terms are illustrated in Figure 23. As it can be observed, both $K$ and $M$ are independent on the number of query terms. However, the average value of $K$ becomes larger as compared to Figure 16-21. In the previous experiment, the average $K$ for TREC6, TREC7, and TREC8 are 1.69, 1.81, 1.62 for BM25 on the title queries. Now the average $K$ is slightly more than 2 in Figure 23.

<table>
<thead>
<tr>
<th># of terms added</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREC6</td>
<td>55</td>
<td>58</td>
<td>59</td>
<td>61</td>
<td>63</td>
<td>63</td>
<td>64</td>
<td>63</td>
</tr>
<tr>
<td>TREC7</td>
<td>64</td>
<td>67</td>
<td>69</td>
<td>70</td>
<td>71</td>
<td>72</td>
<td>72</td>
<td>71</td>
</tr>
<tr>
<td>TREC8</td>
<td>68</td>
<td>70</td>
<td>71</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>71</td>
</tr>
</tbody>
</table>

Table 3: The average number relevant documents retrieved by BM25 for each query when the different numbers of additional query terms are applied to the original title query.

Table 3 shows the average number of relevant documents retrieved for each query when different numbers of additional query terms are applied to the original title query. In Figure 16-21, the average numbers of relevant documents retrieved for each title query are 45, 50, 59 for TREC6, 7, 8, respectively. The increment of $K$ is very likely due to the fact that more relevant documents are retrieved because of the improvement in the retrieval strategy caused by the true relevance feedback query expansion. More specifically, the score range of the relevant documents is probably enlarged to cover more cases, additional Gaussian components which are needed to increase the probability of the occurrence of the relevant documents in the corresponding score interval, especially in the high score range to create a heavy right tail in the mixture distribution (as shown in Figure 22).

Figure 23 also shows that $M$ is always around 1, and fluctuates in a very small range for the different numbers of query terms expanded. When more query terms are added, the standard deviation becomes smaller. A reasonable explanation for this observation is that when we have more query terms, more documents are actually retrieved. Since we only use top 1000 documents to model score distributions and non-relevant documents are the dominant ones, we are more likely to have the right tail of the actual score distribution in non-relevant documents after applying a top-1000 truncation. Hence, an exponential-shape distribution should appear
more often for non-relevant documents. Therefore, the standard deviation of M decreases when more terms are used.

**Number of relevant documents retrieved:** As one can observe in Figure 8 and Figure 9, the number of components of the Gaussian mixture varies across different TREC data sets. In particular, in TREC 9 and 10 the dominant model to describe the distribution of relevant documents scores is the single Gaussian. The opposite is true in TREC 6, 7, 8, and 12. One particular characteristic of TREC 9 and 10 data sets is that the number of relevant documents retrieved is particularly smaller when compared with the traditional ad-hoc tracks of TREC 6, 7, 8 and 12. Thus, we analyze $K$ with respect to the number of relevant documents in the ranked lists. The results are illustrated in Figure 24. In the scatter plot, each point corresponds to a ranked list for a given query. As it can be observed, when the number of retrieved relevant documents is particularly small, usually less than 50, the data cannot support more than one Gaussian. This clearly explains the histogram over the parameter $K$ in Figure 8 and Figure 9 for TREC 9 and 10, since as it can be viewed in Figure 24, the number of relevant documents retrieved per ranked list in these two TREC data sets is always less than 35. However, when the number of retrieved relevant documents is above 50 there is no real indication that the more the number of relevant documents retrieved the more the Gaussian across different collections. Note that, if such a phenomenon was observed, it would have raised the question of whether $K > 1$ is simply artifact.
Figure 24: The number of Gaussian components, $K$, against number of relevant documents retrieved over all ranked lists for each TREC data set.
Relevant documents retrieved by manual vs. automatic runs: In TREC 6, 7 and 8, along with the ranked lists of documents returned by different retrieval systems, a number of ranked lists of documents was retrieved with some manual effort (manual runs\(^2\)). In particular, 17 manual runs were submitted to TREC 6, 17 were submitted to TREC 7 and 13 were submitted to TREC 8. Often times, relevant documents identified with manual effort have very different characteristics from the ones that automatic system are built to find, and thus many manually identified relevant documents do not appear in the top-100 ranks of the lists returned by automatic systems. These documents may be retrieved by automatic systems as well but they are usually assigned smaller scores. One of the hypotheses we test is whether the different Gaussian components separately capture the distributions of the automatically and manually retrieved relevant documents. The question we want to answer is whether the low scores Gaussian components solely model the distribution of relevant documents uniquely identified by manual runs while the high score Gaussian components solely model the distribution of relevant documents retrieved by automatic runs. If a document is identified by both manual runs and automatic runs, we consider it retrieved by automatic runs. To answer this question, we first construct a contingency table for each ranked list of documents (i.e. for each one of the 600 system-query pairs per TREC data set). The two rows of the table correspond to manual and automatic runs, while each column corresponds to each one of the Gaussian component in the best fit mixture. To fill in the table, we first compute the responsibility of each of the Gaussian components for each relevant documents as \( \pi_i \ast P(x|\mu_i, \Lambda_i^{-1}) \), where \( x \) is the score of a relevant document; if the relevant document is uniquely identified by manual (automatic) runs then the responsibility values are appropriately added to the manual (automatic) runs row. In this manner a probabilistic table of counts is created. Each row is normalized to give the distribution of manually (automatically) identified relevant documents over the Gaussian components, i.e. \( P(i^{th} \text{ Gaussian}|\text{type of run}) \). If different components correspond to manual and automatic runs the distance between these two distributions will be high. We measure this distance by the Jensen-Shannon divergence between the two distribution. The average Jensen-Shannon divergence values for TREC 6, 7, 8 are 0.05, 0.06, and 0.10 respective, which do not reveal a correlation between manual and automatic relevant documents and the different Gaussian components. However, whether the Jensen-Shannon divergence is a good measure of the correlation between the manual and automatic with

\(^2\)A manual run may require, for instance, extensive human search in the collection to identify relevant documents
the different Gaussian components needs to be further investigated.

**Relevant vs. highly relevant documents:** Further, we want to test the hypothesis of whether different Gaussian components correspond to different grades of relevance. In TREC 9, 10 and 12 documents were judged in a three-grades scale as non-relevant, relevant and highly relevant. We repeated the same analysis as the one above, by constructing a contingency table for relevant and highly relevant documents. The Jensen-Shannon divergence values computed here also do not demonstrated any correlation between the different Gaussian components and the relevance graded. However, as mentioned before, we intent to further investigate these explanatory factors by different means other than the Jensen-Shannon divergence.

**Precision-Recall curves**

As a utility of our model for IR purposes, we estimate the precision-recall (PR) curve separately from both the Exponential-Gaussian (EG) and Gamma-k-Gaussian (GkG) model. Similarly to Robertson [74], let $f_r$ and $f_n$ denote the model densities of relevant and non-relevant scores, respectively

$$F_r(x) = \int_x^1 f_r(x) dx$$  \hspace{1cm} (48)  

$$F_n(x) = \int_x^1 f_n(x) dx$$  \hspace{1cm} (49)  

are the cumulative density functions from the right. While the density models might have support outside the range [0,1], we use integrals up to 1 because our scores are normalized. For each recall level $r$ we estimate the retrieval score at which $r$ happens, from the relevant cumulative density: \textit{score}(r) = F_r^{-1}(r)$, which we can compute numerically. Then we have $n(r) = F_n(\textit{score}(r))$ as the percentage of non-relevant documents found up to recall $r$ in the ranked list. Finally, the precision at recall $r$ can be computed as in [74], \textit{prec}(r) = \frac{r}{r + n(r)G}$, where $G$ is the ratio of non-relevant to relevant documents in the collection searched. Computing precision at all recall levels from the score distribution models $f_r$ and $f_n$ gives an estimated PR curve. In the remainder of this section we show that estimating PR curves from the GkG model clearly outperforms PR curves estimated from the dominant EG model.

To measure the quality of the estimated PR curves we report the RMS error between the actual and the predicted precisions at all recall levels for both models (see Figure 25). The results are summarized in Table 4, separately for each model. Score
Table 4: Root Mean Square Error (RMSE) between the actual and the inferred precision-recall curves, averaged over about 200 queries from TREC collections.

<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Title+Desc</th>
<th>Title+Desc+Narrative</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EG</td>
<td>GkG</td>
<td>EG</td>
</tr>
<tr>
<td>BM25</td>
<td>.196</td>
<td>.172</td>
<td>.182</td>
</tr>
<tr>
<td>LM</td>
<td>.166</td>
<td>.153</td>
<td>.151</td>
</tr>
<tr>
<td>PL2</td>
<td>.173</td>
<td>.149</td>
<td>.178</td>
</tr>
<tr>
<td>TFIDF</td>
<td>.195</td>
<td>.170</td>
<td>.185</td>
</tr>
</tbody>
</table>

Table 5: Mean Absolute Error (MAE) between actual and inferred precision-recall curves, averaged over about 200 queries from TREC collections.

<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Title+Desc</th>
<th>Title+Desc+Narrative</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EG</td>
<td>GkG</td>
<td>EG</td>
</tr>
<tr>
<td>BM25</td>
<td>.141</td>
<td>.117</td>
<td>.124</td>
</tr>
<tr>
<td>LM</td>
<td>.118</td>
<td>.102</td>
<td>.101</td>
</tr>
<tr>
<td>PL2</td>
<td>.121</td>
<td>.098</td>
<td>.120</td>
</tr>
<tr>
<td>TFIDF</td>
<td>.140</td>
<td>.115</td>
<td>.126</td>
</tr>
</tbody>
</table>

distributions fitting over scores produced by language model (LM) and divergence from randomness (PL2) seem to have slightly better PR estimates, but independent of the query formulation. The over-all RMSE of GkG vs. EG is .155 vs .176, or about 12% improvement.

Further, we report the mean absolute error between the actual and predicted precisions at all recall levels. This is the area difference between the estimated and the actual curve, which immediately gives a bound for the difference in average precision (AP) of the two curves (because the AP metric is approximated by the area under the PR curve). The results are reported in Table 5. Note that the best fit with respect to MAE are given for the full query formulation (title, description and narrative); the overall MAE for GkG is .099 vs EG with .120, or an improvement of about 21%.

**TREC search engines**

In previous sections, the scores used in our experiments are produced by traditional IR models to avoid the effects of arbitrary query manipulations and score transformations that systems submitted to TREC often applied. In this section we apply our methodology over the score distributions returned by actual search engines submitted to TREC 6, 7, 8, 9, 10 and 12. Out of all the runs submitted to TREC
Figure 25: Precision-Recall curve (blue) for query 434 and the BM25 retrieval function implemented by Terrier. It is easy to see that the PR curve estimated from the GkG model (magenta) is much better than the PR estimated from the EG model (brown). Yellow bars indicate the number of non-relevant documents in each recall interval.

A number of them was excluded from our experiments since they are manual runs and scores are computed based on the retrieved rank list. No other quality control was performed. As earlier, we report the parameter $M$ of the Gamma distribution, and the number $K$ of Gaussian components in the mixture, for all systems and all queries as histograms in Figure 26 and Figure 27. As it can be observed, similarly to the case of the traditional IR models, $K$, in most cases, is different from 1, while $M$ varies in a wide range confirming that a Gamma distribution and a mixture of Gaussians is a better fit than an exponential and a single Gaussian.

Further, we tested the same hypotheses as in the case of the traditional IR models, that is whether the number of Gaussian components is correlated with the number of relevant documents retrieved in the ranked lists, the manual versus the automatic runs and the relevant versus the highly relevant documents. Figure 28 illustrates the fact that as in the case of traditional IR models, a small number of relevant documents retrieved cannot support the reconstruction of complex models like the mixture of Gaussians. Further, the Jensen-Shannon divergence between the distributions of Gaussians for manual and automatic runs and relevant and highly relevant documents did not reveal any correlation between them and the different Gaussians.

The Precision-Recall curve estimate obtained for TREC runs is measured in Table 6. The proposed model (GkG) easily outperforms the EG model in terms of both RMSE and MAE. Overall RMSE improvement is about 18%, while overall MAE improvement is about 25%. Note that “harder” TREC (latest collections) also mean harder estimates for both GkG and EG models.
Figure 26: The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution for all automatic runs in TREC 6, 7, 8.

Table 6: RMSE and MAE between the actual and the inferred precision-recall curves reported separately on TREC6-12 ad-hoc runs.
Figure 27: The histograms over the number $K$ of Gaussian components and the parameter $M$ of the Gamma distribution for all automatic runs in TREC 9, 10, 12.
Figure 28: The number of Gaussian components, $K$, against number of relevant documents retrieved over all ranked lists for each TREC data set.
3.4 Summary

In this chapter, we proposed to model the relevant document scores by a mixture of Gaussians and the non-relevant document scores by a Gamma distribution. The intuition about the choice of the particular model from an IR perspective was given but the main approach was data-driven. We extensively studied the correlation between the number of Gaussian components K in the mixture distribution and the value of the parameter M in the Gamma distribution with a number of explanatory factors. The results of our experiments demonstrated that the distributions of K and M are independent both of the IR model used and the number of query terms. Further, we showed that a small number of relevant documents retrieved may prohibit the Variational Bayes framework to fit complex models such as the mixture of Gaussians to the relevant document scores. Finally, we demonstrated the utility of our model in inferring precision-recall curves.

The aim of this work was to revisit the problem of modeling score distributions under the observation that the Gaussian-exponential model fails to capture the underlying process that generates document scores\(^3\). However, an important question that arises here is the practical consequences of this work. Applying the proposed model to traditional applications such as information filtering, distributed retrieval or data fusion is not straightforward. In all our experiments we intentionally made use of the actual relevance judgments, while all the afore-mentioned applications require blind fit of score distributions. The fact that we proposed the use of a richer model than the traditional ones makes a blind fit harder. Given that, the applications of our model or the practical consequences of the results we obtained from the analyses we conducted is a topic of future research.

\(^3\)The recent study [12] showed that the Gamma-k-Gaussian model overall outperforms the Gaussian-exponential model, but the percentage difference between the errors of two models slightly narrows when the length of the query increases.
Chapter 4

Score Distribution Model
Part 2: Model Derivation

Inferring the score distribution for relevant and non-relevant documents in the absence of any relevance information is an extremely difficult task, if at all possible. Modeling score distributions is often the basis of any possible inference. Over the past decades, many statistical distributions have been proposed to model the relevant and non-relevant scores as we discussed in the previous chapter. However, the complexity of the underlying process that generates document scores makes it hard to theoretically argue about the actual distribution of document scores. Most of the aforementioned models were proposed on the basis of empirical fits to scores produced over different document corpora.

In this chapter, we model score distributions in a rather different, systematic manner. We start with a basic assumption on the distribution of terms in a document. Following the transformations applied on term frequencies by two basic ranking functions, BM25 and Language Models, we derive the distribution of the produced scores for all documents in an analytical form and illustrate that the derived distribution can be well approximated by a Gamma distribution.

Further, we also consider the score distribution for relevant documents. We detach our analysis from particular ranking functions. Instead, we consider a simple model for precision-recall curves proposed by Aslam and Yilmaz [15], which makes some very basic assumptions about the shapes of precision-recall curves that are produced by reasonable retrieval systems on average. Given this model, we present a general mathematical framework which, given any score distribution for all retrieved documents, produces an analytical formula for the score distribution of relevant documents that is consistent with the precision-recall curves that
follow the aforementioned model. In particular, assuming a Gamma distribution for all retrieved documents, we show that the derived distribution for the relevant documents resembles a Gaussian distribution with a heavy right tail. In the end, we show the connection between the derived score distribution to the empirical model proposed in the previous chapter.

4.1 Related Work

Numerous statistical distributions have been proposed to model relevant and non-relevant document scores as we discussed in the previous chapter. However, most of them were based on empirical fits to scores produced over different document corpora but lacking rigorous theoretical arguments due to the complexity of the underlying process that generates document scores. There have been several attempts to intuitively argue about the shape of the different distributions. The starting point for most of these attempts has been some basic assumptions about the frequency of query term occurrences in documents (e.g. in Manmatha et al. [58]). Harter [50] and Bookstein and Swanson [29] used a mixture of Poisson distributions to model the distribution of words in a document, with one Poisson (with large $\lambda$) corresponding to the distribution of words in relevant documents and the other (with small $\lambda$) to the distribution of words in non-relevant documents. Meanwhile, Arampatzis and van Hameren in their work [14] claimed that a central limit appears in the score distribution for the relevant documents with increasing query length, but for non-relevant documents, the Gaussian central limit was unlikely and if it appears it does with a very slow query length increasing rate.

In a different line of work, Robertson in [74] proposed the recall-fallout convexity hypothesis, which states:

For all good systems, the recall-fallout curve (seen from the ideal point of recall = 1, fallout = 0) is convex.

The same hypothesis can be also formulated as that the probability of the relevance for a document should be a monotonically increasing function of the score, which means the document with a higher score should have higher probability of relevance [74, 58]. Meanwhile, Robertson considered various combinations of distributions including two exponentials, two Gaussians, two Poissons, two Gammas, and Gaussian-exponential, then examined whether these combinations exhibit anomalous behavior with respect to theoretical properties of precision and recall. Among those score distribution models, the convexity condition is always satisfied in two Poisson distributions and two exponential distributions, but it is violated for some
score ranges in the case of two Gaussian distributions, two gamma distributions, and the commonly used Gaussian-exponential distributions.

Regarding Gaussian-exponential model, several works have addressed its non-convexity issue. Although convexity violation happens at both ends of the score range, the high-end scores are far more important for the practical application. Hence, most works proposed attempted to solve non-convexity for scores in the top of the rank. In [58], Manmatha et al. used a straight line from the maximum to the point \((1, 1)\) to replace the probabilities when the prediction function starts declining. This is considered as an empirical fix, but lacks of theoretical justification. The purpose is to ensure the function of the probability of relevance to be a monotonically increasing function over the scores.

In another work [11], Arampatzis et al. proposed to force the convexity hypothesis by imposing a uniform distribution over the score range where convexity violation occurs. Arampatzis et al. also argued that a more aggressive solution can be used by reversing the offending sub-rankings. However, the practical performance depends on the quality of the actual system, how much the initial ranking does satisfy the probability of ranking principle.

The other practical issue for Gaussian-exponential model is that the exponential distribution is defined at or above 0, but Gaussian distribution has a full real number support. Manmatha et al. [58, 43] normalized scores into the range between 0 and 1. Arampatzis et al. [11] proposed a truncated version of Gaussian-exponential model.

In the most recent work [40, 41], Cummins proposed a mixture of two log-normal distributions to model document scores, one for the relevant ones, and the other for the non-relevant ones. Cummins argued that under the strong score distribution hypothesis [12], the score distributions for both relevant and non-relevant documents should be able to approach a Dirac’s delta function. Meanwhile, there is no theoretically valid reason why distributions for relevant and non-relevant documents should be drawn from two different distribution families given that the document score for two categories of documents is generated by the same retrieval function in an information retrieval system. Therefore, he considered the combinations of two Gaussian distributions, two exponential distributions, two Gamma distributions, and two log-normal distributions in [40]. His experiments showed two log-normal model outperformed others for both goodness-of-fit and utilities of inferring the average precision and predicting the query performance. However, two log-normal model still violated Robertson’s recall-fallout convexity hypothesis in the empirical analysis.
4.2 Motivation

Traditional retrieval models score documents based on how well their language matches the language of the user’s request. Thus, the essential component of all traditional scoring functions is the number of occurrences of query terms within a document (term frequency, TF). Different retrieval models apply different transformations over the term frequencies to produce a score per query term. The final score of a document is usually an aggregate of the document scores for each individual term.

Before we consider the distribution of term frequencies and the transformation applied by ranking functions over them in an analytical manner we illustrate the evolution of the term frequency distribution for all retrieved documents (documents that contain at least one of the query terms) for a sample query from the TREC 8 ad hoc collection (Ireland Peace Talks) and for two different retrieval models, BM25 and Language Models, in Figure 29. The upper panel corresponds to the transformation of TF distribution by BM25 scoring function, while the lower panel corresponds to the transformation by the Jelinek-Mercer Language Model. Each column then, in both panels, corresponds to an individual query term and each row to progressively more complex transformations of the term frequency step by step. The bottom row plots illustrate the final score distribution by the two retrieval models.

As can be observed, for both retrieval models, there is a critical step in the term frequency transformation (from Row 2 to Row 3) after which the score distribution radically changes and appears to be closer to the final score distribution. Furthermore, the shape of the final score distribution appears to be dominated by the most frequent query term in the collection (as expected) — for the sample query this is the term talk — and thus our main goal will be to derive the score distribution for each individual query term.

4.3 Score Distribution Derivation

4.3.1 Deriving the term distribution from raw statistics

For a fixed query, consider a partition of the collection into relevance classes, such that \( D_Q \) is the class of documents that satisfy the information need to a certain degree \( Q > 0 \). Depending on several factors like the user, the information need, the collection of documents etc., \( Q \) can take a range of values from "completely

\[1\] The parameter values used for BM25 are \( k_1=1.2 \) and \( b=0.75 \), and \( \lambda = 0.2 \) for the Jelinek-Mercer Language Model.
The empirical histograms of term frequencies resulting to the final scores for the query (Ireland Peace Talks) over the TREC8 Ad Hoc Track collection for BM25 (top) and JM Language Model (bottom). Each column corresponds to a single stemmed query term while the rows correspond to progressively more complex transformations of the term frequency (TF) up to the final score for the two ranking functions. DL is the document length, ADL is the average document length, CTF is the collection term frequency, and TN is the number of terms in the collection.
irrelevant” (the lowest $Q$) to “extremely relevant” (the highest $Q$). Note that in test collections (such as TREC) for simplicity only two or three classes are considered. The discussion in this section assumes a fixed quality/relevance class $Q$, and assumes all documents in the class contain all query terms at least once.

A query term $t$ has a certain contribution to the document relevance if it occurs in the document in response to the user query. For a given document quality $Q$, we assume an approximately constant probability of seeing the term $t$ at any position in a document in class $D_Q$; hence we can model term $t$’s occurrences in documents in class $D_Q$ using a Poisson process with the rate $\lambda = f(g, Q)$, where $g$ relates to the general rarity of the term $t$ in the language. Such a document generating model is memoryless and assumes that the likelihood of the query term occurrence is same at any word interval in the document space. We do not model the dependence in $f$ among different terms, and $f$ can be any monotonic function, depending on the class model.

Counting the occurrences of a term $t$ when reading a random document $d \in D_Q$ is analogous to counting buses at a bus station: arrive at the station, wait for the first bus, for the second bus, etc., and leave at some point (when the document ends). It is well known that the waiting times $w_1, w_2, w_3, \ldots$ among Poisson generated events can be modeled by an exponentially distributed i.i.d. random variable

$$w_i \approx \lambda e^{-x\lambda}.$$  (50)

The average waiting time is $\theta = 1/\lambda$, the mean of the exponential distribution. Intuitively, $\theta$ corresponds to a notion of the expected ratio of document length to term frequency, i.e., $DL/TF$. Our purpose is to model the distribution of the random variable $DL/TF$ for documents in class $D_Q$. We will do so separately for each possible frequency value and then express the general distribution as a mixture.

Given a term frequency $k$, $k$ is an non-negative integer, and can be $1, 2, 3, \ldots$. Let

$$D_{Qk} = \{d \in D_Q \mid TF(t, d) = k\}$$  (51)

denote the set of documents in $D_Q$ that contain term $t$ exactly $k$ times. Here, we make the approximation that the document has been truncated and ends exactly after the $k$-th occurrence of $t$. Therefore, we can write the document length $DL$ as the sum of $k$ waiting times $\sum_{i=1}^{k} w_i$, which immediately implies that $DL$ is Gamma distributed (and more specifically Erlang-distributed), with shape parameter $k$ and
scale parameter $\theta = 1/\lambda$:

$$DL_{Qk} \sim Gamma(k, \theta).$$

(52)

Since $k$ is a constant for the subclass $D_{Qk}$, the waiting time $X_{Qk} = \frac{DL}{TF = k}$ is also Gamma distributed, where

$$X_{Qk} = \frac{\text{Doc Length}}{\text{Term Frequency}} = \frac{DL}{k} \sim Gamma(k, \theta/k).$$

(53)

Since the quality class $D_Q$ is partitioned into the classes $D_{Qk}$ for $k = 1, 2, 3, \ldots$, the waiting time $X$ on $D_Q$ follows a mixture of Gamma distributions with a constant mean $\theta$, while $DL$ on $D_Q$ follows a mixture of Gamma distributions with a constant scale $\theta$:

$$DL_Q \sim \sum_k P_Q(k) \cdot Gamma(k, \theta)$$

(54)

$$X_Q \sim \sum_k P_Q(k) \cdot Gamma(k, \theta/k)$$

(55)

where $P_Q(k) = \Pr[TF(d, t) = k \mid d \in D_Q]$ denotes the probability that a document in class $D_Q$ contains the term $t$ exactly $k$ times.

Assuming that the probability of document relevance that having a term $t$ in a document can be modeled by a Bernoulli distribution with the success probability $p$. Given a quality class $Q$, $P_Q(k)$ can be expressed as probability of $k - 1$ failures (term occurrences that do not imply quality $Q$) followed by one success (term occurrence when quality $Q$ is reached); therefore we model the mixture probabilities $P_Q(k)$ with a geometric distribution (equivalent to a negative binomial distribution with $\beta = 1$),

$$P_Q(k) = p(1 - p)^{k-1}$$

(56)

where $p = \theta / ADL_Q$ expresses the correlation between the term and the information need on the class $D_Q$ (the average document length, the general rarity of the term $t$, and the quality $Q$). Intuitively $p$ can be thought as a notion of inverse term frequency:

$$p = \theta / ADL_Q \approx \text{avg}(DL/TF) / ADL_Q \approx \text{avg}(1/TF).$$

(57)

Note that a number of different mixtures could be used, perhaps based on the query type. For instance, an informational query could use a negative binomial or a Poisson mixture. In the particular case of a geometric mixture however, an interesting result follows: Neuts and Zachs [60] show that under certain conditions
similar to ours, a negative binomial mixture of Gamma distributions with constant scale is actually itself a Gamma distribution. With a different notation, their result is

$$\text{Gamma}(\beta, \theta/p) = \sum_k p_k \cdot \text{Gamma}(\beta + k, \theta)$$

when

$$p_k = \text{NegBinomial}(p, \beta) = \binom{k + \beta - 1}{\beta - 1} p^\beta (1 - p)^k$$

Applying this on DL (with $\beta=1$) implies that DL is exponentially distributed on $D_Q$ with mean $\theta/p$. Of course this must hold for all query terms, not only for $t$, which requires a proportionality $\theta/p = \text{constant} = ADL_Q$. In practice, for a given quality class, the document length variable will not be exactly exponentially distributed for two reasons: (1) relevance judgments cover a range of qualities inducing an average effect, (2) our Poisson process model for query term occurrence works reasonably well for frequent terms, but can fail on rare terms. However, this model is fairly accurate in that DL can be modeled well by a Gamma distribution with a small shape parameter (the exponential distribution is Gamma with shape = 1.)

Figure 30 illustrates the empirical histogram of DL/TF for the query term system. As can be observed, a Gamma distribution appears to be a good approximation of the empirical score distribution, offering empirical evidence that the assumptions and approximations in our theory are reasonable.²

### 4.3.2 Deriving the score distribution from scoring functions

In this section, we will derive the score distribution of the retrieved documents in a systematic manner. We consider the transformation applied on the distribution of the elementary statistics described in the previous section by two scoring functions, BM25 and Jelinek-Mercer Language Model. The derivations presented here can be applied in the case of other retrieval models, such as TF-IDF and Divergence From Randomness (DFR).

#### Score Transformations

Consider a transformation of the random variable $X$ by a monotonic, differentiable function $r$, $Y = r(X)$. The probability density function (pdf) of $Y$, $f_Y(y)$, can then be computed as a function of the pdf of $X$, $f_X(x)$ [23]. Let $F_Y(y)$ and $F_X(x)$ be the cumulative density function (cdf) of $Y$ and $X$, respectively. Without loss of

²Some fits will be better than others, depending on the example. No theoretical model will fit all empirical examples, of course.
Figure 30: The empirical histogram and the Gamma density function fit over the $\frac{DL}{TF}$ scores for term system in TREC8.

generality let $r$ be a non-decreasing function. Then,

$$F_Y(y) = Pr\{Y \leq y\} = Pr\{r(X) \leq y\}$$  \hspace{1cm} (60)

$$= Pr\{X \leq r^{-1}(y)\}$$  \hspace{1cm} (61)

$$= F_X(r^{-1}(y))$$  \hspace{1cm} (62)

and

$$f_Y(y) = \frac{d}{dy} F_Y(y) = \frac{d}{dy} F_X(r^{-1}(y))$$  \hspace{1cm} (63)

$$= \frac{\partial r^{-1}(y)}{\partial y} \cdot f_X(r^{-1}(y))$$  \hspace{1cm} (64)

In the general case of a monotonic function $r$,

$$f_Y(y) = \left| \frac{\partial r^{-1}(y)}{\partial y} \right| \cdot f_X(r^{-1}(y))$$  \hspace{1cm} (65)

A rudimentary transformation of interest is just the inverse of $X = DL/TF$, which gives the normalized term frequency $TF/DL$. According to the previous section, $X = DL/TF \sim f_X = \sum_{k \geq 1} P_Q(k) \cdot Gamma(k, \frac{\theta}{k})$. It is known that a mixture of
Gamma can approximate any smooth function [90]. By approximating $P_Q(k)$ with a geometric distribution, inverting $TF/DL$ has the effect as displayed in Figure 31. A relevant class of documents (high $Q$) implies:

- the geometric rates $1 - p = 1 - 1/(\lambda \cdot ADL)$ for query terms are higher, which means the mean $1/p$ is higher, or the mixture $P_Q$ will have non-negligible coefficients for higher scale parameters $k$. This will make the mixture look more “hill”-like due to more effective components.

- for each query term, the Poisson generating process will be governed by a higher rate, $1/\theta$, which dictates a lower mean to all Gamma components of the mixture, or a “light” right-side tail. When the inverse transformation is performed (see below), the result distribution will have a heavier tail.

Conversely, a lower quality $Q$ implies a mixture with effectively significant coefficients only for the lower $k$ values, and also that the components of the mixture are less skewed towards the left-side, overall producing a more exponential-like distribution (after inversion).

![Figure 31: The mixture of Gamma distributions before and after the inversion, for different quality classes](image)

Note that in practice fitting a Gamma, an inverse Gamma or an inverse Gaussian distribution in the $TF/DL$ scores of existing collections/judgments (like TREC) are likely to differ in goodness-of-fit mostly due to random effects than other theoretical reasons - this is primarily due to complex score manipulations, and due to the sparsity and inaccuracy of the judgment process.
BM25 and Jelinek-Mercer LM

Assuming that query terms appear only once within a query, the BM25 for a single query term can be calculated as:

\[
\text{BM25 score} = \frac{(k_1 + 1) \cdot TF}{k_1((1 - b) + b \cdot \frac{DL}{ADL}) + TF} \cdot IDF
\]

(66)

where \(TF\) is the term frequency, \(IDF\) is the BM25 inverse document frequency, \(DL\) is the document length, and \(ADL\) is the average document length in the collection. By setting the parameter \(b\) equal to 1 (fixing the document length normalization) and defining the variable \(X = DL/TF\), BM25 can be approximated by,

\[
Y = r(X) = \frac{IDF(k_1 + 1)}{CX + 1}, \quad X > 0
\]

(67)

where \(C = k_1/ADL\). Given Equation 67 it can be shown that

\[
r^{-1}(Y) = \frac{IDF(k_1 + 1) - Y}{CY}
\]

(68)

Now, let \(f_X(x)\) be the probability density function (PDF) of \(X\) and \(f_Y(y)\) the PDF of \(Y\). Since function \(r\) is a monotonic and differentiable when \(X\) is positive, based on the principle of function transformations of random variables [23], we can calculate
the PDF of $Y$ as a function of the PDF of $X$,

$$f_Y(y) = \frac{-\text{IDF}(k_1 + 1)}{C_y^2} f_X\left(\frac{\text{IDF}(k_1 + 1) - y}{C_y}\right)$$  \hspace{1cm} (69)$$

when $0 < y < \text{IDF}(k_1 + 1)$ and 0 otherwise.

In other words we can model the PDF of an approximation of BM25 as a function of the density function of the reverse relative term frequency. Essentially, one can plug in the above formula any distribution for the relative term frequency and get an analytical form distribution of BM25.

Based on the previous section $DL/TF$ approximately follows a Gamma distribution. Let $\hat{k}$ and $\hat{\theta}$ are estimated parameters of the Gamma distribution from $X$ via maximum likelihood estimation (MLE) for all retrieved documents (see Figure 30). Then, the approximated pdf of BM25 score for a single term can be obtained as follows,

$$f_Y(y) = \frac{-\text{IDF}(k_1 + 1)}{C_y^2} \text{Gamma}\left(\frac{\text{IDF}(k_1 + 1) - y}{C_y}; \hat{k}, \hat{\theta}\right)$$  \hspace{1cm} (70)$$

### Jelinek-Mercer Language Model

We repeat the exact same derivation in the case of language models with Jelinek-Mercer smoothing. The score for each term is computed as,

\[ \text{JMLM score} = \log \left( \lambda \frac{TF}{DL} + C(1 - \lambda) \right) \]  \hspace{1cm} (71)$$

where $C = \frac{CTF}{TN}$. $CTF$ is collection term frequency and $TN$ is the number of unique terms in the collection. As before, we let $X = DL/TF$, then the LM score can be written as,

$$Y = r(X) = \log \left( \frac{\lambda}{X} + C(1 - \lambda) \right)$$ \hspace{1cm} (72)$$

Using the previous assumption that $DL/TF$ is modeled by a Gamma distribution and since the function $r$ is a monotonic and differentiable, after the random variable transform over $X$ we get the pdf of the LM scores as a function of the Gamma distribution that models the reverse relative term frequency.

$$f_Y(y) = \frac{-\lambda e^y}{(e^y - C(1 - \lambda))^2} \text{Gamma}\left(\frac{\lambda}{e^y - C(1 - \lambda)}; \hat{k}, \hat{\theta}\right)$$ \hspace{1cm} (73)$$

Figure 32 shows the comparison among the empirical histogram, the analytical model derived from the distribution of $DL/TF$, and the Gamma distribution obtained by MLE over BM25 and JM language model scores all retrieved docu-
ments for query system in TREC8 collection. As illustrated on the plots, the analytical model has more freedom than the Gamma distribution, but the Gamma is still a reasonable approximation to the term score distribution. Further, the mixture model presented in the previous section with the best-fit $\lambda$ is also shown on Figure 32 (black line denoted as “Model (theory)” in the legend).

**Remark on the Shape of the Distribution**

Most term frequency weighting functions are nonlinear monotonically increasing functions of the raw term frequency. In BM25 Roberston’s TF grows fast when the raw term frequency is small and gets gradually saturated. The parameter $k_1$ controls the speed of the saturation. The logarithm function in Language Models also has this saturation property but without the power of controlling the saturating speed. Therefore, the JM language model scoring function has a similar to BM25 impact on transforming the distribution of low level statistics, such as DL/TF or normalized TF to the final score distribution.

As it is illustrated in Figure 30 the typical shape of the distribution for the DL/TF tends to have a long right tail but a fast rising-up left tail. After applying a transformation function with the saturation property, the imbalance between two tails of the original distribution is alleviated, so the peak of the new distribution is right shifted, and with a shorter right tail compared to the original one. The amount of difference is dominated by the parameter controlling the saturating speed. This can be viewed in Figure 33. As $k_1$ becomes larger and the weighting function more linear the empirical histograms of BM25 looks more similar to the distribution of DL/TF in Figure 30. This implies that the term score distribution can also be approximated by a Gamma distribution by adjusting the shape and the scale parameters.

**Summation over Query Terms**

In this thesis, for a given query $q$ and document $d$ we have considered scoring functions with the following property:

$$\text{score}(d, q) = \sum_{t \in q} r(X_t), \quad (74)$$

where $X_t = DL/tf(t, d)$. This class of scoring functions includes BM25, TF-IDF, some Language Models etc, but does not include scores like PageRank. Assuming term independence, the intuition for the summation score $= \sum_t r(X_t)$ is as follows:

- For non-relevant documents (low quality $Q$) each $r(X_t)$ will be distributed approximately as a Gamma(low shape, low scale). If the scales are approx-
imately equal, their sum follows a Gamma distribution with the same scale (gamma distribution exhibits infinite divisibility).

- For relevant documents, the mixture for each term has more effective components, thus making the sum a rich mixture, usually Gaussian like (or Gamma-like with higher scale and shape).

Thus, the distribution of the summation of several term scores could also be modeled using a Gamma distribution if we use a Gamma distribution to model the term score distribution. Figure 34 shows this summation process.

4.3.3 Inferring the score distribution for relevant documents

In this section, we relate the score distributions for relevant and non-relevant documents with precision-recall curves. That the score distributions for relevant and non-relevant documents are related to precision-recall curves is well known and
unsurprising: Given the two score distributions, one can easily infer a precision-recall curve [74], and we shall do so below as part of the treatment that follows. More interestingly, we demonstrate that one can infer the score distribution for relevant documents given a score distribution for non-relevant documents and a precision-recall curve, and we use the technique described to show that the score distributions for relevant documents will tend to have a Gaussian-like form, with a heavy right tail.

Let \( f_R(s) \) and \( f_N(s) \) be the score distributions for relevant and non-relevant documents, respectively. For any score threshold \( t \), consider the set of documents whose scores are \( t \) or higher. The recall and fallout associated with this document set are easily defined in terms of \( f_R(s) \) and \( f_N(s) \) as follows:

\[
\begin{align*}
    r(t) &= \int_t^\infty f_R(s) \, ds \\
    fo(t) &= \int_t^\infty f_N(s) \, ds.
\end{align*}
\]

Now let \( C \) be the size of the collection and let \( \gamma \) be the fraction of the collection that is relevant to a given query. Then there are \( R = \gamma C \) total relevant documents and \( N = (1 - \gamma)C \) total non-relevant documents. At score \( t \) or above, there are

\[
R \cdot r(t) = \gamma C \cdot r(t)
\]
relevant documents and

\[ N \cdot fo(t) = (1 - \gamma)C \cdot fo(t) \]

non-relevant documents. Thus, the precision associated with this document set is simply

\[ p(t) = \frac{\gamma C \cdot r(t)}{\gamma C \cdot r(t) + (1 - \gamma)C \cdot fo(t)} = \frac{r(t)}{r(t) + O \cdot fo(t)} \]  

(77)

where \( O = (1 - \gamma) / \gamma \) is the odds of non-relevance in the collection. Equations 75 and 77 are parametric equations defining a precision-recall curve: Given the score distributions \( f_R(s) \) and \( f_N(s) \) (and \( \gamma \)), one can vary the score threshold \( t \) in Equations 75 and 77 to obtain the precision-recall curve. (A substantially similar treatment can be found in Robertson [74].)

Now suppose that one has a candidate score distribution for either relevant or non-relevant documents and one has a candidate form for a precision-recall curve: Can one derive a form for the other score distribution? In what follows, we show how this can be accomplished, and using the score distributions described in Section 4.3.2 and a simple form for precision-recall curves, we infer a form for the score distributions of relevant documents.

Consider the simple model for precision-recall curves described by Aslam and Yilmaz [15] and shown in Figure 35. This family of precision-recall curves is defined by the following equation, implicitly parameterized by the value of R-precision \( rp \):

\[ p(r) = \frac{1 - r}{1 + \alpha \cdot r}. \]  

(78)

(Here \( \alpha = (1/rp - 1)^2 - 1 \) governs the “shape” of the curve.) While it is certainly the case that “real” precision-recall curves are never this “clean”, this simple model captures many properties found in real precision-recall curves, such as high precisions at low recall levels, low precisions at high recall levels, and so on. Furthermore, Aslam and Yilmaz show that this simple model allows one to explicitly and accurately relate average precision, R-precision, precision-at-cutoff, and other seemingly disparate measures of retrieval performance.

Using such a model for precision-recall curves, we can relate the score distributions for relevant and non-relevant documents as follows. We first parameterize Equation 78 by the score threshold \( t \), obtaining

\[ p(t) = p(r(t)) = \frac{1 - r(t)}{1 + \alpha \cdot r(t)}. \]  

(79)
We now equate Equations 77 and 79

\[ \frac{r(t)}{r(t) + O \cdot fo(t)} = \frac{1 - r(t)}{1 + \alpha \cdot r(t)} \]  

(80)

and solve for \( r(t) \) as a function of \( fo(t) \)

\[ r(t) = \frac{-O \cdot fo(t) + \sqrt{(O \cdot fo(t))^2 + 4(1 + \alpha)O \cdot fo(t)}}{2(1 + \alpha)} \]  

(81)

Differentiating Equation 81 by \( t \) immediately establishes a closed-form relationship between the score distributions for relevant and non-relevant documents, since by Equations 75 and 76 and the Fundamental Theorem of Calculus, we have

\[ r'(t) = -f_R(t) \]
\[ fo'(t) = -f_N(t). \]

As an example of this methodology, let us assume that the score distribution for all documents follows a Gamma distribution, as we argued in Section 4.3.2.
Figure 36: Inferred relevant document score distribution and empirically histogram for the TREC8 query “Estonia, economy”.

Figure 37: Typical form of the relevant document score distribution in TREC8.
Since the overwhelming majority of documents are non-relevant, the score distribution for non-relevant documents will then tend to follow a Gamma distribution as well. Now consider the Gamma that fits the non-relevant documents for the TREC8 query “Estonia Economy”. Using this Gamma distribution for the non-relevant documents, together with a precision-recall curve\(^3\) from the family show in Equation 78, and employing the method described above, we obtain the score distribution for relevant documents shown in Figure 36.

While Figure 36 gives just one such example, the form of this curve is quite consistent across all tested input distributions from the Gamma family (which includes the negative exponential distribution) and all tested precision-recall curves from the family defined by Equation 78: The distribution is roughly Gaussian in form, but with a heavy right tail. That the score distribution is “Gaussian-like” is much assumed (as discussed in the introduction), but the heavy right tail is also necessary to avoid problems with a simple Gaussian, such as those described by Manmatha et al. [58] and others. Figure 37 shows the typical form of the relevant document score distribution we obtained in TREC 8. We here for the first time derive such a form, given reasonable forms for non-relevant score distributions and precision-recall curves.

Our results in this section are descriptive rather than prescriptive, and as such, we conclude the following:

The tendency of the score distributions for relevant documents to look Gaussian with a heavy right tail is a natural and inevitable consequence of the facts that (1) the score distributions of non-relevant documents tend to look Gamma and (2) precision-recall curves tend to have the form shown in Figure 35.

Comparison to the Empirical Model

In this section, the score distribution derived for relevant documents has a Gaussian-like shape but with a heavy right tail. In Chapter 3, we have proposed an empirical model, a mixture of Gaussians, to model the relevant document scores. Apparently, two models do not share any common characteristics. However, after taking a close look, we find that the weights for different Gaussian components in the empirical model are actually very different. There always exists an dominant component that makes the shape of the distribution close to a Gaussian shape. Meanwhile, other small components (often located on the right side of the score range) are added to

\(^3\)We set \(\gamma = \frac{1}{rp} - 1\) and \(\alpha = (1/rp - 1)^2 - 1\) to match those parameters from the BM25 run on that query.
increase the probability of the occurrence of relevant documents in the correspond-
ing score intervals. Those small Gaussians create a similar heavy right tail as in the
model derived in this chapter.

Figure 38 illustrates a typical mixture of two Gaussians for relevant document
score inferred based on the empirical model for query “Estonia, economy”. Two
red dash lines correspond to two Gaussian components, and the blue solid line is
the mixture of the two Gaussians, which is the final score distribution for relevant
documents. We can see that the Gaussian component on the right side has a smaller
weight but a larger variance. It contributes to the heavy right tail of the final score
distribution. This blue line is very similar to the derived one (as shown in Figure 36)
in this chapter, which is the score distribution of the relevant documents for the
exactly same query.

In the most recent work [40, 41] by Cummins, a log-normal distribution has
been proposed to model the relevant document scores, and achieved a better per-
formance compared to other empirical models. The log-normal distribution also
exhibits the similar property that it has a Gaussian-like shape along with a heavy
right tail. However, from the theoretical point of view, the log-normal model still
violates the recall-fallout convexity hypothesis in certain score range. Our derived
distribution does not have this problem.

4.4 Summary

In this chapter, we attempt to model score distributions in a rather systematic man-
ner. We start with a basic assumption that the occurrences of the query terms are
governed via a Poisson process and induced that the distribution the relative term
frequency in a document is a inverse Gamma distribution. Following the mathe-
matical transformations applied on the relative term frequencies by two basic rank-
ing functions, BM25 and Language Models, we derived the distribution of the pro-
duced scores, in an analytical form and illustrate that the derived distribution can
be well approximated by a Gamma distribution.

Further, we also considered the score distribution for relevant documents by
relating score distributions with precision-recall curves. In particular, we adopted
a precision-recall curve model that has previously been proposed and given this
model we presented a general mathematical framework under which given any
score distribution for all retrieved documents we can derive an analytical formula
for the score distribution of relevant documents. The framework is general enough
such that the same derivations can be repeated for different models of precision
recall curves. Finally, under the assumption that non-relevant documents follow a
**Figure 38**: The score distribution of the relevant documents for TREC8 query "Estonia, economy". The red dash lines are two Gaussian components, and the blue line is the mixture of two Gaussians. The scores are produced by BM25, and the Gaussian components are inferred by the same process used in Chapter 3.

Gamma distribution for all retrieved documents, we show that there is a tendency of the derived distribution for the relevant documents to look Gaussian with a heavy right-hand tail.
Chapter 5

Inferring Score Distributions without Relevance

In Chapters 3 and 4, we focused on how to model score distributions in an appropriate manner both empirically and theoretically. With the assumption of the “correct” score distribution model choice, inferring the score distributions associated with relevant and non-relevant documents can be easily accomplished when many relevance judgments are available. However, we are most often faced with the situation of estimating model parameters in the absence of relevance information. How to infer the score distributions accurately without relevance information becomes one of the biggest challenges of applying score distributions to solving practical IR problems. Expectation maximization (EM) [28] has often been used to estimate the parameters of the mixture model, and it is naturally considered as a good solution under score distribution scenario. Many previous works have relied on it with some success [14, 11, 10, 58]. However, it has been noticed that EM suffers from treating all data equally, being very sensitive to initialization, and converging to a local optimum instead of the global one [10, 11, 12], which make applying score distributions to solving real IR problems an extremely difficult task.

In this chapter, we propose a novel framework for inferring score distributions and document relevance probabilities through the simultaneous inference of multiple score distributions for different systems. For a given query, when a document is retrieved by multiple systems, we have multiple inferred probabilities of relevance for that document which can be calculated from the different score distribution models associated with each system. By averaging these per-system probabilities, we can have a better approximation to the “true” universal probability of relevance associated with that document. Based on this approach, we extend the EM algo-
algorithm to estimate model parameters for multiple systems simultaneously. While the method still initializes and updates model parameters for each system separately, during the *expectation* phase, it uses the average probability of relevance associated with a document as the inferred current universal probability of relevance for that document, thus naturally taking advantage of the information and constraints present in multiple retrieved lists.

5.1 Related Work

Given the choice of the score distribution model, model parameter inference can be easily accomplished when relevance information is available. However, when score distributions are applied to real IR problems, the relevance judgements are most often absent during the inference process. Therefore, how to accurately infer score distributions for relevant and non-relevant documents without relevance information is the key to successful application in practice.

*Expectation maximization* (EM) [28] has been used to find the maximum likelihood estimates of mixture model parameters in many areas. It has also been considered as the candidate for estimating score distribution model parameters. In previous work, EM has achieved some success in many practical applications [14, 11, 10, 58]. However, it has been noted that EM suffers from many problems. First, it treats all data equally, but scores in the top rank are far more important than others and there are more documents with low scores than the ones with high score. Hence, the inference process in EM often does not give enough weights to those high scores [10, 11]. Second, the algorithm itself is very sensitive to the initialization, and easily converges to a local optimum instead of the global one [28, 12]. Moreover, EM has typically been applied as in a traditional mixture model data fitting: the method estimates model parameters on a per-request and per-system basis, iteratively infers the probability of relevance of a document given its score from the current parameters, and then uses this inferred probability to update model parameters.

The deployment of EM also does not take any advantage of many natural IR constraints or assumptions, such as that for a given query, an individual document can be retrieved by multiple systems, and this document should have comparable if not identical inferred probabilities of relevance across the systems corresponding to its “true” probability of relevance. For example, Aslam and Yilmaz [16] showed that inferring document relevance from multiple retrieved lists can greatly benefit from this constraint. The inference of score distributions can also incorporate with the same constraint. In a result, the estimation of the probability of the document
relevance based on two score distributions should also be expected to be more accurate.

5.2 Extended Expectation Maximization

5.2.1 Methodology

The focus of this work is the inference process itself rather than the selection of probability density function (PDF) to be used in modeling the distribution of documents over scores. As a show case we select the widely used Gaussian-Exponential mixture to model score distributions and present a novel way to estimate the score distribution mixture parameters by inferring the probability of relevance for each document from multiple retrieval systems. The following table summarises the notation used.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>A retrieval system</td>
</tr>
<tr>
<td>x</td>
<td>The score for a retrieved document computed by a system</td>
</tr>
<tr>
<td>d</td>
<td>The document retrieved by a system</td>
</tr>
<tr>
<td>θ</td>
<td>The parameters for the score distribution mixture model</td>
</tr>
<tr>
<td>r = {rel, nrel}</td>
<td>The hidden variable indicating whether a document is relevant</td>
</tr>
<tr>
<td>φ</td>
<td>PDF for the distribution of relevant scores</td>
</tr>
<tr>
<td>ψ</td>
<td>PDF for the distribution of non-relevant scores</td>
</tr>
</tbody>
</table>

Probability of Relevance

The extension of the EM algorithm proposed in this work is based on the assumption that the probability of relevance inferred by the score distributions of relevant and non-relevant documents produced by a system should precisely estimate the actual probability of document relevance, which is independent of the system, and is an intrinsic quality of the document itself. In what follows we give a more formal description of this assumption.

Let us assume that a system conflates different documents over approximately the same score \( x \), and let relevance conflation rate be the proportion of relevant documents conflated over this score. This expresses the probability that a document with a certain score is relevant. For a retrieved document \( d \) with score \( x \),

\[
P(r = \text{rel}|x) = \frac{P(d \text{ is relevant}|x \approx x')}{|\{d|x \approx x'; d \text{ is relevant}\}|} \quad (82)
\]

\[
= \frac{|\{d|x \approx x'; d \text{ is relevant}\}|}{|\{d|x \approx x'\}|} \quad (83)
\]

87
The mean probability of relevance across all documents is equal to the generality \( G \) of the collection, i.e. the proportion of relevant documents in the collection, which is indeed independent of the scoring function of any IR system:

\[
G = \frac{\#\text{relevant docs}}{\#\text{docs}} \quad (84)
\]

For a given query, one can re-write \( G \) by iterating over all scores that a retrieval system would assign to all documents in the collection, or all documents in the collection as follows:

\[
G = \frac{1}{\#\text{docs}} \sum_d \frac{|\{d|x \approx x'; d \text{ is relevant}\}|}{|\{d|x \approx x'\}|} \quad (85)
\]

\[
= \frac{1}{\#\text{docs}} \sum_d |\{d|d \text{ is relevant}\}| \quad (86)
\]

\[
= \frac{\#\text{relevant docs}}{\#\text{docs}} \quad (87)
\]

We call an IR system consistent if the relevance conflation rate reflects the true probability of relevance as perceived by a user. This notion of consistency implies that all relevant documents conflated around the same score have on average the same value to a user, or in other words that \( P(r|x) \) estimates an intrinsic quality of the relevance for the document \( d \), \( P(r|d) \), regardless of the IR system.

**Score Distribution**

A score distribution mixture model can be written as the linear combination of two score distributions: \( \pi \phi(x) + (1 - \pi) \psi(x) \), where \( \pi \) is the mixing coefficient. For a given search engine \( s \), the proportion of relevant documents with score \( x \) is equal to

\[
P_s(r = \text{rel}|x) = \frac{P_s(r = \text{rel})P_s(x|r = \text{rel})}{P_s(r = \text{rel})P_s(x|r = \text{rel}) + P_s(r = \text{nrel})P_s(x|r = \text{nrel})} \quad (88)
\]

\[
= \frac{\pi \phi(x)}{\pi \phi(x) + (1 - \pi) \psi(x)} \quad (89)
\]

If the model fits the score output of a search engine, the above equation accurately estimates the true, global, probability of document relevance \( P(r|d) \). This is essentially the consequence of applying Bayesian law to score distributions of relevant and non-relevant documents [58].

88
Treating \(P_s()\) as an estimator, we can compute its bias:

\[
E_x[P_s(r|x)] - \text{mean}[P(r|d)] = \int P_s(r|x)P_s(x)dx - G = \pi - G
\]

This bias is very much dependent on the initial choice of the mixture coefficient \(\pi\) for each IR system modeled with score distributions. It is also determined by the ability of the fitting algorithm (EM in our case) to recover correct \(\pi\) from bad initial values, in subsequent iterations. We briefly discuss later how our extended EM algorithm helps on this.

### 5.2.2 EM and the multi-system extended EM

Estimating the parameters of the mixture in the absence of relevance judgments is an extremely hard task. The difficulty arises from the lack of knowledge about the hidden variable \(r\), which determines the relevance of a document. The probability of a document \(d\) being relevant, \(P(r|d)\), can be estimated through score distributions using Equation 89. Regular expectation maximization algorithm infers this probability through the posterior \(P(r|x, \theta)\), and uses it to maximize the conditional expectation \(E_{r|x, \theta}\{\log P(x, r|\theta)\}\) [28]. \(\theta\) is the model parameter, and \(x\) is the score for document \(d\). The whole process can be summarized as follows:

1. Initialize the parameters for the mixture model of score distributions
2. **E step:** estimate \(P(r|x, \theta)\) given the current model parameters
3. **M step:** update parameters to maximize \(E_{r|x, \theta}\{\log P(x, r|\theta)\}\); in our case they are the mixing coefficient \(\pi\), the Gaussian parameters \(\mu\) and \(\sigma\), and the Exponential parameter \(\lambda\).
4. Repeat steps 2 and 3 until the conditional expectation of the log-likelihood converges

Observing the above optimization iterations, the conflation rate for a document is estimated purely based on parameters for a single query-system run. Inspired by the work in [16], a better estimate can be obtained by combining information from other ranked lists, in which that document also appears. The central idea to our new approach is that a specific document has a global probability of relevance \(P(r|d)\) independent of the retrieval system and the score it has been assigned. For instance, in Figure 39, document \(d\) is retrieved by both systems \(A\) and \(B\) with scores \(x_{d,A}\) and \(x_{d,B}\) respectively. The probability of relevance of \(d\) inferred from the score distribution models for system \(A\) or \(B\) via Equation 89 should be same and equal to
Figure 39: A document \(d\) with scores from two different systems and the conflation rate in each system. \(\phi\) and \(\psi\) shown are the Gaussian-Exponential model.

\(P(r|d)\) if document scores computed by scoring functions can represent the probability of relevance correctly.

In the EM algorithm’s E step, for each system \(s\), \(P_s(r|x, \theta)\) is expected to infer the true relevance probability \(P(r|d)\). Since \(P(r|d)\) is de facto the intrinsic probability of relevance and should be independent of the system \(s\), and the qualities of systems may vary, a better estimate can be obtained from many systems by taking the average of probabilities of relevance calculated from the model for each system \(s\):

\[
\hat{P}(r|d) = \frac{1}{\# \text{ systems}} \sum_s P_s(r|x_s, \theta_s) \tag{90}
\]
The averaged estimator reduces both the bias and the variance and in this way it helps the EM algorithm.

- **Bias**: the bias $\pi_s - G$ for system $s$ depends on the current EM-iteration parameter $\pi_s$. Averaging, the estimator’s bias becomes $\frac{1}{\#\text{systems}} \sum_s \pi_s - G$, which on average is a smaller absolute bias, unless all $\pi_s$ are unusually high or low.

- **Variance**: averaging random variables always decreases the variance, up to a linear factor in the independent case.

- **Convergence**: by using Equation 90 to estimate the probability of the hidden variable, extended EM no longer guarantees that the whole inference process would converge in the end as in EM. However, we did not have any convergence problem in practice. In fact, the inference converges within a reasonable number of iterations. Our argument for this is that if systems are not consistent (in practice they are not), averaging helps by making EM converging faster: averaged probabilities of relevance need fewer iterations to become stable.

- **Parameter estimates** of EM output: if the score distribution model chosen does not always fit the data (and in practice it does not [42]), averaged probabilities help identify relevant documents better than the per-system probabilities do.

After estimating the probability of relevance for each document, we update each parameter by setting the derivative of the conditional expectation of the log-likelihood $\mathbb{E}_{r|x,\theta}\{\log P(x,r|\theta)\}$ with respect to each parameter to zero. For every document $d$ with score $x_d$ retrieved by multiple systems, the updating equations become,

$$
\mu = \frac{\sum_d \hat{P}(r|d)x_d}{\sum_d \hat{P}(r|d)}
$$

(91)

$$
\sigma^2 = \frac{\sum_d \hat{P}(r|d)(x_d - \mu)^2}{\sum_d \hat{P}(r|d)}
$$

(92)

$$
\lambda = \frac{\sum_d (1 - \hat{P}(r|d))}{\sum_d (1 - \hat{P}(r|d))x_d}
$$

(93)
One can observe that if relevance judgements were available, \( \hat{P}(r|d) \) equals 1 when document \( d \) is relevant, and 0 otherwise. Then \( \mu \) and \( \sigma \) are the mean and standard deviation of scores of relevant documents, and \( \lambda \) is the inverse of the mean of nonrelevant documents’ scores. The extended expectation maximization algorithm can be seen below.

**Algorithm 1** Extended Expectation Maximization Algorithm

**Require:** a list of retrieved document scores \( x_s \) for all \( s \in S \)

1: Initialize the parameters of the mixture \( \theta_s \) for each system \( s \)
2: while algorithm has not converged do
3:     for all system \( s \) in \( S \) do
4:         Compute the posterior \( P(r_d|x_s, \theta_s) \) for each document \( d \) with score \( x_s \)
5:     end for
6:     for all system \( s \) in \( S \) do
7:         for all document \( d \) in \( x_s \) do
8:             Estimate \( \hat{P}(r|d) \) according to Equation 90
9:         end for
10:     end for
11:     Update model parameters for system \( s \) through Equation 91
12: end while

5.2.3 Experiments

We first describe the experiment setup and three inference methods used to estimate the parameters of the score distributions and the documents’ probability of relevance. Then, we describe a series of experiments and show that the proposed extension on EM significantly outperforms the regular EM in terms of the precision that one can infer system precision-recall curves and average precision. Finally, we use the inferred probabilities of relevance in the task of metasearch and demonstrate that the extended EM can achieve good performance.

As mentioned earlier, in this work we use a mixture of Gaussian and Exponential density functions to model the score distributions of documents, regardless of some noticeable theoretical and practical problems this model has [74, 55, 12], since our primary goal is the parameter estimation method. Besides, our proposed algorithm can be easily adopted to other score distribution models. To evaluate our methodology we use TREC data and infer the score distributions for automatic search engines run over different queries. For each system-query run the scores of the retrieved documents are first normalized into a 0 to 1 range. We use 0.1 and 0.9 as the weights for exponential and Gaussian distributions. We initialize the mean of the exponential distribution to 0.2, and the mean and the standard deviation of the
Gaussian distribution to 0.8 and 0.4. Those initialization parameters were found in the empirical experiments, which yielded the best performance. The model parameters of the score distributions are then estimated through the following three approaches:

- **judSD**: TREC judgments are used to estimate the model parameters separately for relevant and non-relevant documents.

- **regEM**: the regular EM algorithm is used to estimate model parameters in absence of relevance judgments. It infers the model parameters for multiple systems one by one.

- **extEM**: the proposed extended EM algorithm is used to estimate model parameters in absence of relevance judgments. It infers the model parameter for multiple systems simultaneously.

Under the assumption that the score distribution can fit the data well, one cannot hope for a more accurate estimation of the model parameters than the one obtained when using relevance judgments. Hence, in our experiments, **judSD** is considered the gold standard. Results obtained by the other two approaches are compared with the ones obtained by **judSD**. Given that relevance judgments are available we could compare all three approaches with the actual gold standard but this way any results would conflate the effects not only of the inference process but also of the model’s inherent goodness of fit. Using **judSD** as a gold standard eliminates effect of the imperfect score distribution model.

### 5.2.4 Precision-recall curve

Precision recall (PR) curves can be easily inferred through score distributions [74]. Let \( \Phi(x) = \int_x^1 \phi(x)dx \) and \( \Psi(x) = \int_x^1 \psi(x)dx \) be the cumulative density functions from the right for the relevant and non-relevant documents respectively. We use integrals up to 1 because our scores are normalized into a range from 0 to 1. For each recall level \( r \) we can estimate the score at which the retrieval system achieves recall equal to \( r \) by the inverse of relevant document cumulative density function: \( \text{score}(r) = \Phi^{-1}(r) \). Then, \( n(r) = \Psi(\text{score}(r)) \) is the percentage of non-relevant documents found up to recall \( r \) in the ranked list. Hence, the precision at recall \( r \) can be computed similarly as in [74],

\[
\text{prec}(r) = \frac{r}{r + n(r) \times N}
\]  

(94)
Table 7: Mean and standard deviation of the RMSError and ABSError of the estimated PR curves for 116 automatic systems over 50 queries submitted to TREC 8 Ad Hoc Track.

<table>
<thead>
<tr>
<th></th>
<th>RMSError</th>
<th>ABSError</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean</td>
<td>stdev</td>
</tr>
<tr>
<td>regEM</td>
<td>0.374</td>
<td>0.237</td>
</tr>
<tr>
<td>extEM</td>
<td>0.142</td>
<td>0.131</td>
</tr>
</tbody>
</table>

where \( N \) is the number of non-relevant documents retrieved. Computing precision at all recall levels from the score distribution models \( \phi \) and \( \psi \) gives an estimated PR curve.

In the first experiment, we infer the precision recall curve based on the three inference approaches: regEM, extEM, and judSD for all 116 automatic systems submitted to TREC 8 ad-hoc track. We report the mean and standard deviation of the root mean square errors (RMSError) and absolute errors (ABSError) between the predicted precisions at all recall levels using judSD and the one using regEM or extEM.

The results are summarized in Table 7. The table shows that extEM produces significantly better PR estimates than regEM, both in terms of mean and standard deviation of the errors. Furthermore, among all 5800 runs (116 systems and 50 queries), there are 5164 (89.0%) runs for which extEM is better than regEM in terms of RMSError with an average absolute (relative) improvement of 0.27 (65.2%), and 5139 (88.6%) runs with an average absolute (relative) improvement of 0.25 (66.9%) in terms of ABSError.

Since extEM can utilize multiple systems to more accurately infer parameters for score distributions, we also report the RMSError and ABSError over different numbers of systems. Figure 40 shows the average RMSError and ABSError of predicted PR curve by regEM and extEM comparing to judSD over different numbers of TREC 8 systems. Systems are ordered by their mean average precision reported by TREC. The \( n \) systems on the plot represent the best \( n \) systems based on TREC evaluation. In all cases extEM outperforms regEM. We can observe that the prediction error produced by extEM increases a little in the beginning, then decreases when utilizing more systems. However, regEM does not have this trend. The prediction error increases when more low-quality systems are included in the experiment.
Figure 40: The average RMSError and ABSErrror of inferred PR curve through regEM and extEM for different numbers of TREC8 systems.

5.2.5 Expected average precision

Expected average precision (EAP) is a probabilistic version of average precision, and can be computed as [18]:

$$
\mathbb{E}\{AP\} = \frac{1}{R} \sum_{i=1}^{N} \left( \frac{p_i}{i} \left( 1 + \sum_{j=1}^{i-1} p_j \right) \right)
$$

(95)
where $N$ is the number of retrieved documents, and $p_i$ is the probability of the $i$th document in the rank list being relevant. This probability can be directly inferred through the estimated mixture of score distributions by Equation 89, where

$$p_i = \frac{\pi \phi(x_i)}{\pi \phi(x_i) + (1 - \pi) \psi(x_i)}$$

$x_i$ is the score for the document at rank $i$. When relevance information is available, this probability is either 0 or 1, and EAP reduces to AP. $R$ is the number of relevant documents. We compute $R$ as $\sum_{i=1}^{N} p_i$. Here $R$ may be underestimated as compared with the $R$ reported by TREC, since we only estimate the number of relevant documents from a single retrieved list. Hence, EAP computed by our approach is often overestimated as compared to AP evaluated by TREC.

We use different numbers of systems submitted to TREC to infer the model parameters. This time we extend our experiment data to include automatic systems submitted to TREC 6, 7 and 8 ad-hoc tracks, TREC 9 and 10 web tracks (ad-hoc tasks) and TREC 12 robust track. The topics used are the TREC topics 301-550 and 601-650. The Robust track topic set in TREC 12 consists of two subsets of topics, the topics 601-650 and 50 old topics selected based on topic hardness from past collections.

We first compute EAP for those systems using judSD, regEM, and extEM, then average EAPs over different queries to get the probabilistic version of mean average precision, mean EAP. Figure 41 shows the mean EAP estimated from regEM or extEM for all automatic systems submitted to different TREC and its correlation with the one estimated from judSD. A blue square dot indicates mean EAP estimated by regEM averaged over different queries, and a red star dot indicates mean EAP estimated by extEM. As we can see, red star dots are mostly clustered along the diagonal line, showing that numbers predicted by extEM are clearly more correlated with ones predicted by judSD.

### 5.3 Application to Metasearch

Due to the complexity of the modern search engine, we are often faced with the situation of merging various ranked lists of documents to construct a final retrieval list for the end users. Those ranked lists of documents can be retrieved from different sub search engines underneath, or from different document collections (images, blogs, news, webpages, etc.). To correctly merge documents whose scores may have completely different meanings, certain score normalization mechanism should be used to transform the local score to a universal one independent on the
Figure 41: The scatter plot of mean EAP estimated by regEM (blue square dots) / extEM (red star dots) comparing with the one estimated by judSD for all automatic systems submitted to TREC6, 7, 8, 9, 10, 12.
characteristics of search engines or underlying document collections. Score distributions can be used to infer the probability of the relevance through Equation 90, therefore they are very useful for information tasks such as information fusion and metasearch.

Another area where score distributions can play an important role is recall-oriented retrieval such as patent or legal search. Under this scenario, it is critical to identify the score threshold for document assessment. The cost of missing a relevant document is usually expensive in those search applications, so document assessment is often conducted by professionals. There is a cost involved when a document is judged by a human. Meanwhile, there is also a reward when more relevant documents are found. Because the score distribution approach is able to infer the number of relevant documents under certain score threshold, it allows people to make an appropriate trade-off decision between the reward and cost during the document assessment according to their budget.

In this section, we will show the utility of a score distribution model inferred by the extended EM in the metasearch scenario. A metasearch engine typically sends the user request to several underlying sub-engines, and aggregates the results into a single list. Score distributions are used to normalize the scores of documents returned by different sub-engines.

5.3.1 Methodology

There are many score normalization methods that have been proposed for metasearch, which convert the relevance scores into comparable ones across systems. The simple normalization strategies include 0-1 normalization that normalizes scores into a range between 0 and 1, and z-score normalization that normalizes scores to a common scale with a zero mean and a standard deviation of one. In a different line of work, a single score distribution model has been developed for score normalization. Fernández et al. mapped scores to an optimal score distribution, which is defined as the score distribution of an ideal scoring function that matches the ranking by actual relevance [45, 46]. In [38, 51], researchers attempt to find an appropriate score distribution by aggregating historical data or averaging over different historical queries. Apart from those standard normalization approaches, combMNZ is one of the most popular and effective metasearch algorithms [47]. CombMNZ not only normalizes the score but also boosts the score of a document by the number of systems retrieving it. For example, a document \( d \) is retrieved by \( k \) out of \( n \) systems,
and \( s_i \) is the relevance score given by system \( i \). Its combMNZ score is computed as

\[
\text{CombMNZ Score} = \sum_{i=1}^{k} \bar{s}_i \times k \quad (96)
\]

and normalized score \( \bar{s}_i \) is computed as

\[
\bar{s}_i = \frac{s_i - s_{\text{min},i}}{s_{\text{max},i} - s_{\text{min},i}} \quad (97)
\]

\( s_{\text{min},i} \) and \( s_{\text{min},i} \) are the maximum and minimum document scores retrieved by the system \( i \) for the given query. In the following experiments, we choose combMNZ as our non-score-distribution metasearch algorithm baseline for the comparison.

When score distributions are used in metasearch, the probability of a document relevance given the score can be estimated by Equation 89. Documents retrieved by different systems can be simply merged by their estimated probabilities of document relevance. If a document appears in multiple systems, this probability can be computed as the average of different estimations from different systems as shown in Equation 90.

5.3.2 Experiments and results

To test the practical utility of our methodology, we examine how well it improves the performance of using inferred score distributions for the task of metasearch. The testbed we use, again, is still all automatic search engines submitted to TREC 6, 7 and 8 ad-hoc tracks, TREC 9 and 10 web tracks (ad-hoc tasks) and TREC 12 robust track. Same queries are used as in the previous experiment. We randomly select 10, 20, 30, 40, and 50 systems for our experiments, and merge results from those systems using probabilities of relevance for each document estimated by judSD, regEM, and extEM. The whole process is repeated 20 times for different system numbers, and we report the mean average precision averaged over these 20 repetitions. The same query sets are used for different TRECs as in the previous experiments. The results are also compared with the most popular metasearch algorithm, combMNZ, which does not rely on score distributions.

Table 8 shows that judSD outperforms all other methods in metasearch. This is not surprising since judSD uses the relevance information. Score distributions estimated by extEM perform almost consistently better than ones estimated by regEM regardless of the number of systems (denoted by \( \dagger \)). For a few TREC 7 and TREC 10 systems, extEM performs worse than regEM, which may be due to some bad-quality systems included that undermine the inference process. How-


<table>
<thead>
<tr>
<th># of rand systems</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TREC6</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2139</td>
<td>0.2623</td>
<td>0.2695</td>
<td>0.2819</td>
<td>0.2852</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2709†</td>
<td>0.2998†</td>
<td>0.2988†</td>
<td>0.3084†</td>
<td>0.3102†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3569</td>
<td>0.3859</td>
<td>0.3936</td>
<td>0.4049</td>
<td>0.4071</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2474</td>
<td>0.2808</td>
<td>0.2812</td>
<td>0.2886</td>
<td>0.2882</td>
</tr>
<tr>
<td><strong>TREC7</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2139</td>
<td>0.2623</td>
<td>0.2695</td>
<td>0.2819</td>
<td>0.2852</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2462†</td>
<td>0.2652†</td>
<td>0.2679</td>
<td>0.2669</td>
<td>0.2773</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3107</td>
<td>0.3310</td>
<td>0.3376</td>
<td>0.3394</td>
<td>0.3452</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2584</td>
<td>0.2742</td>
<td>0.2732</td>
<td>0.2738</td>
<td>0.2809</td>
</tr>
<tr>
<td><strong>TREC8</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2641</td>
<td>0.2873</td>
<td>0.2965</td>
<td>0.3045</td>
<td>0.3036</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2914†</td>
<td>0.3061†</td>
<td>0.3111†</td>
<td>0.3184†</td>
<td>0.3162†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3496</td>
<td>0.3614</td>
<td>0.3700</td>
<td>0.3731</td>
<td>0.3730</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.3055</td>
<td>0.3145</td>
<td>0.3176</td>
<td>0.3224</td>
<td>0.3192</td>
</tr>
<tr>
<td><strong>TREC9</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.1951</td>
<td>0.2157</td>
<td>0.2298</td>
<td>0.2340</td>
<td>0.2415</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2042†</td>
<td>0.2238†</td>
<td>0.2369†</td>
<td>0.2430†</td>
<td>0.2466†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.2913</td>
<td>0.3114</td>
<td>0.3234</td>
<td>0.3328</td>
<td>0.3347</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2389</td>
<td>0.2525</td>
<td>0.2614</td>
<td>0.2675</td>
<td>0.2701</td>
</tr>
<tr>
<td><strong>TREC10</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.1894</td>
<td>0.2102</td>
<td>0.2179</td>
<td>0.2226</td>
<td>0.2288</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2050†</td>
<td>0.2123†</td>
<td>0.2155</td>
<td>0.2150</td>
<td>0.2214</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3072</td>
<td>0.3332</td>
<td>0.3445</td>
<td>0.3435</td>
<td>0.3527</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2458</td>
<td>0.2608</td>
<td>0.2627</td>
<td>0.2608</td>
<td>0.2683</td>
</tr>
<tr>
<td><strong>TREC12</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2139</td>
<td>0.2623</td>
<td>0.2695</td>
<td>0.2819</td>
<td>0.2852</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2544†</td>
<td>0.2756†</td>
<td>0.2826†</td>
<td>0.2884†</td>
<td>0.2892†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3072</td>
<td>0.3332</td>
<td>0.3445</td>
<td>0.3435</td>
<td>0.3527</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2852</td>
<td>0.2945</td>
<td>0.2977</td>
<td>0.3027</td>
<td>0.3019</td>
</tr>
</tbody>
</table>

Table 8: Mean average precision achieved by meta-search using regEM, extEM, judSD, combMNZ based on randomly selected 10, 20, 30, 40, 50 systems from all automatic systems submitted to TREC 6, 7, 8, 9, 10, 12 averaged over 20 times.

However, metasearch based on extEM does not seem to do better than combMNZ. ExtEM only outperforms combMNZ for experiments on systems submitted to TREC 6 (numbers shown in bold face). This might be caused by several reasons: (1) The imperfect score distribution choice might yield the incorrect estimation of the probability of document relevance. (2) Gaussian-exponential distribution violates the convexity hypothesis as we discussed in Chapter 4, and this has already been noticed to be problematic in the practical applications [10, 58]. (3) In the experiment, we randomly selected systems from all submissions to TREC. Those selections might include those systems with very poor quality, and since extEM uses the average estimates from all systems, the bad systems would damage the overall performance.
On the quality of the system

Since this work focuses on the inference process of the score distribution model, we will leave the first speculation as the future work. To test the other two speculations, two more experiments are conducted. In order to ensure the stability of the experiment results, the same random section and repeating strategy are applied. However, this time we only randomly select systems from top 30 ranked submissions based on the TREC evaluation to guarantee the system quality.

Table 9 shows the metasearch results performed by regEM, extEM, judSD, and combMNZ on randomly selected 5, 10, 15, 20, 25 systems from the top 30 system pool for TREC 6, 7, 8, 9, 10, 12 averaged over 20 times. Since the quality of systems are better than the previous systems, all numbers in the table are better than the corresponding ones (see Column 10 and Column 20 in Table 8). ExtEM outperforms regEM in all cases, but still is beaten by combMNZ except for TREC 6 systems.

On the recall-fallout convexity hypothesis

To address the non-convexity issue of the Gaussian-exponential model, we rely on the same solution used in [10]. A uniform distribution interpolation is applied to the score range where the recall-fallout convexity hypothesis is violated to keep the property that the probability of relevance is monotonically increasing with the score. If score $x_i$ is greater than score $x_{\text{max}}$, the score that yields maximum probability of document relevance from the score distribution model, and its estimation of probability of document relevance $P(x_i) < P(x_{\text{max}})$ then $P(x_i) = P(x_{\text{max}})$. A more aggressive approach used in [58] is abandoned due to the concern that the underlying systems may violate the probability ranking principle in practice.

Table 10 lists the metasearch results performed by regEM-interp, extEM-interp, judSD-interp, and combMNZ on randomly selected 5, 10, 15, 20, 25 systems from the top 30 system pool for TREC 6, 7, 8, 9, 10, 12 averaged over 20 times. RegEM-interp, extEM-interp, and judSD-interp are same as regEM, extEM, and judSD, but with using a interpolated probability of document relevance estimation. We see that both regEM and extEM benefit from the interpolation. ExtEM-interp still outperforms regEM-interp, and now extEM-interp is much more comparable with the combMNZ. It beats combMNZ in almost all experiments on TREC 6, 7, 8 and 12, and yields very close results for TREC 9 and TREC 10 experiments. Given that in TREC 9 and 10 experiments, we are using web track collection. This collection consists of queries and documents with very different characteristics. The Gaussian-exponential model might not be the best choice. With an improved model, we
<table>
<thead>
<tr>
<th># of rand systems</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TREC6</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2522</td>
<td>0.2784</td>
<td>0.2901</td>
<td>0.2993</td>
<td>0.3062</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2677†</td>
<td>0.2949†</td>
<td>0.3040†</td>
<td>0.3072†</td>
<td>0.3085†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3274</td>
<td>0.3582</td>
<td>0.3747</td>
<td>0.3808</td>
<td>0.3880</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2694</td>
<td>0.2807</td>
<td>0.2870</td>
<td>0.2904</td>
<td>0.2915</td>
</tr>
<tr>
<td><strong>TREC7</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2828</td>
<td>0.2950</td>
<td>0.3008</td>
<td>0.3041</td>
<td>0.3049</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2894†</td>
<td>0.3000†</td>
<td>0.3073†</td>
<td>0.3091†</td>
<td>0.3097†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3287</td>
<td>0.3446</td>
<td>0.3524</td>
<td>0.3534</td>
<td>0.3552</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.3002</td>
<td>0.3107</td>
<td>0.3127</td>
<td>0.3137</td>
<td>0.3152</td>
</tr>
<tr>
<td><strong>TREC8</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2820</td>
<td>0.3006</td>
<td>0.3012</td>
<td>0.3113</td>
<td>0.3112</td>
</tr>
<tr>
<td>extEM</td>
<td>0.3167†</td>
<td>0.3334†</td>
<td>0.3375†</td>
<td>0.3433†</td>
<td>0.3434†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3611</td>
<td>0.3765</td>
<td>0.3827</td>
<td>0.3853</td>
<td>0.3866</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.3437</td>
<td>0.3529</td>
<td>0.3563</td>
<td>0.3572</td>
<td>0.3574</td>
</tr>
<tr>
<td><strong>TREC9</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2281</td>
<td>0.2552</td>
<td>0.2651</td>
<td>0.272</td>
<td>0.2791</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2486†</td>
<td>0.2740†</td>
<td>0.2763†</td>
<td>0.2813†</td>
<td>0.2850†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.2823</td>
<td>0.3220</td>
<td>0.3264</td>
<td>0.3311</td>
<td>0.3410</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2787</td>
<td>0.3021</td>
<td>0.3036</td>
<td>0.3097</td>
<td>0.3148</td>
</tr>
<tr>
<td><strong>TREC10</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.1903</td>
<td>0.2046</td>
<td>0.2079</td>
<td>0.2141</td>
<td>0.2177</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2110†</td>
<td>0.2238†</td>
<td>0.2262†</td>
<td>0.2305†</td>
<td>0.2335†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3064</td>
<td>0.3290</td>
<td>0.3430</td>
<td>0.3483</td>
<td>0.3556</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2606</td>
<td>0.2698</td>
<td>0.2739</td>
<td>0.2785</td>
<td>0.2820</td>
</tr>
<tr>
<td><strong>TREC12</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM</td>
<td>0.2498</td>
<td>0.2635</td>
<td>0.2680</td>
<td>0.2716</td>
<td>0.2721</td>
</tr>
<tr>
<td>extEM</td>
<td>0.2502†</td>
<td>0.2742†</td>
<td>0.2768†</td>
<td>0.2820†</td>
<td>0.2835†</td>
</tr>
<tr>
<td>judSD</td>
<td>0.3135</td>
<td>0.3349</td>
<td>0.3408</td>
<td>0.3453</td>
<td>0.3474</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2958</td>
<td>0.3041</td>
<td>0.3061</td>
<td>0.3065</td>
<td>0.3069</td>
</tr>
</tbody>
</table>

**Table 9:** Mean average precision achieved by meta-search using regEM, extEM, judSD, combMNZ based on randomly selected 5, 10, 15, 20, 25 systems from top ranked 30 systems submitted to TREC 6, 7, 8, 9, 10, 12 averaged over 20 times.

Another very interesting observation is that judSD-interp does not benefit from the interpolated version of the probability of relevance estimation. In most cases, judSD-interp underperforms judSD. The recall-fallout convexity hypothesis in [74] for score distributions is based on the probabilistic ranking principle (PRP) hypothesis. However, systems do not always follow PRP in practice. In the case that a system does not rank documents according to PRP for a query, the interpolation approach results in more troubles than benefits. JudSD infers two score distributions with the help of the relevance information, so it knows when and where the violation happens, and infers a score distribution mode according to the reality to the most extent. Therefore, justSD greatly outperforms all other methods, and applying interpolation to the model does not help judSD at all. When the rele-
The performance improvement of regEM, extEM, and judSD using the interpolation method over TREC 6, 7, 8, 9, 10, 12. We can see that regEM benefits most from the interpolation, but the performance of judSD is undermined.

5.4 Summary

In this chapter we proposed a novel approach to infer the probability of document relevance through multiple sets of score distributions for different systems. We extend EM algorithm by imposing the constraint that the document appearing in multiple ranked lists returned by different systems should have the same probability of the relevance. In the experiment, the score distributions estimated by the new proposed extend EM clearly outperforms the one estimated by regular EM in terms of inferring precision-recall curves and estimating expected average precisions. We also demonstrate the use of these improved probabilities on the task of
<table>
<thead>
<tr>
<th># of rand systems</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREC6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM-interp</td>
<td>0.2673</td>
<td>0.2899</td>
<td>0.2993</td>
<td>0.3059</td>
<td>0.3111</td>
</tr>
<tr>
<td>extEM-interp</td>
<td><strong>0.2755†</strong></td>
<td><strong>0.3005†</strong></td>
<td><strong>0.3100†</strong></td>
<td><strong>0.3158†</strong></td>
<td><strong>0.3168†</strong></td>
</tr>
<tr>
<td>judSD-interp</td>
<td>0.3212</td>
<td>0.3505</td>
<td>0.3645</td>
<td>0.3690</td>
<td>0.3734</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2694</td>
<td>0.2807</td>
<td>0.2870</td>
<td>0.2904</td>
<td>0.2915</td>
</tr>
<tr>
<td>TREC7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM-interp</td>
<td>0.2962</td>
<td>0.3081</td>
<td>0.3123</td>
<td>0.3150</td>
<td>0.3163</td>
</tr>
<tr>
<td>extEM-interp</td>
<td><strong>0.3032†</strong></td>
<td><strong>0.3165†</strong></td>
<td><strong>0.3214†</strong></td>
<td><strong>0.3228†</strong></td>
<td><strong>0.3246†</strong></td>
</tr>
<tr>
<td>judSD-interp</td>
<td>0.3260</td>
<td>0.3408</td>
<td>0.3470</td>
<td>0.3479</td>
<td>0.3496</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.3002</td>
<td>0.3107</td>
<td>0.3127</td>
<td>0.3137</td>
<td>0.3152</td>
</tr>
<tr>
<td>TREC8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM-interp</td>
<td>0.3361</td>
<td>0.3472</td>
<td>0.3524</td>
<td>0.3555</td>
<td>0.3562</td>
</tr>
<tr>
<td>extEM-interp</td>
<td>0.3394†</td>
<td>0.3514†</td>
<td><strong>0.3570†</strong></td>
<td><strong>0.3594†</strong></td>
<td><strong>0.3598†</strong></td>
</tr>
<tr>
<td>judSD-interp</td>
<td>0.3615</td>
<td>0.3760</td>
<td>0.3820</td>
<td>0.3844</td>
<td>0.3856</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.3437</td>
<td>0.3529</td>
<td>0.3563</td>
<td>0.3572</td>
<td>0.3574</td>
</tr>
<tr>
<td>TREC9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM-interp</td>
<td>0.2638</td>
<td>0.2911</td>
<td>0.2970</td>
<td>0.3014</td>
<td>0.3066</td>
</tr>
<tr>
<td>extEM-interp</td>
<td>0.2704†</td>
<td>0.2934†</td>
<td>0.2933</td>
<td>0.2974</td>
<td>0.3054</td>
</tr>
<tr>
<td>judSD-interp</td>
<td>0.2737</td>
<td>0.3038</td>
<td>0.3067</td>
<td>0.3105</td>
<td>0.3158</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2787</td>
<td>0.3021</td>
<td>0.3036</td>
<td>0.3097</td>
<td>0.3148</td>
</tr>
<tr>
<td>TREC10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM-interp</td>
<td>0.2338</td>
<td>0.2524</td>
<td>0.2547</td>
<td>0.2644</td>
<td>0.2696</td>
</tr>
<tr>
<td>extEM-interp</td>
<td>0.2516†</td>
<td>0.2669†</td>
<td>0.2718†</td>
<td><strong>0.2795†</strong></td>
<td><strong>0.2798†</strong></td>
</tr>
<tr>
<td>judSD-interp</td>
<td>0.3020</td>
<td>0.3208</td>
<td>0.3320</td>
<td>0.3368</td>
<td>0.3425</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2606</td>
<td>0.2698</td>
<td>0.2739</td>
<td>0.2785</td>
<td>0.2820</td>
</tr>
<tr>
<td>TREC12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>regEM-interp</td>
<td>0.2884</td>
<td>0.3014</td>
<td>0.3053</td>
<td>0.3069</td>
<td>0.3082</td>
</tr>
<tr>
<td>extEM-interp</td>
<td><strong>0.2969†</strong></td>
<td><strong>0.3098†</strong></td>
<td><strong>0.3118†</strong></td>
<td><strong>0.3144†</strong></td>
<td><strong>0.3152†</strong></td>
</tr>
<tr>
<td>judSD-interp</td>
<td>0.3193</td>
<td>0.3363</td>
<td>0.3410</td>
<td>0.3439</td>
<td>0.3448</td>
</tr>
<tr>
<td>combMNZ</td>
<td>0.2958</td>
<td>0.3041</td>
<td>0.3061</td>
<td>0.3065</td>
<td>0.3069</td>
</tr>
</tbody>
</table>

Table 10: Mean average precision achieved by meta-search using regEM-interp, extEM-interp, judSD-interp, combMNZ based on randomly selected 5, 10, 15, 20, 25 systems from top ranked 30 systems submitted to TREC 6, 7, 8, 9, 10, 12 averaged over 20 times metasearch.

In the future work, Equation 90 can be extended to use a weighed average to better estimate the probability of document relevance. Weights can be determined by the quality of the system or the rank of that document in a system. Instead of the simple average, a more sophisticated way to combine multiple estimations from different systems should be investigated. Furthermore, extended EM can also be applied to estimate parameters for those recently proposed state-of-art score distribution models [11, 55].
Chapter 6

Conclusions

This thesis focuses on modeling score distributions in information retrieval. We have studied the topic in two aspects: (1) What distribution model should be used to model scores for relevant and non-relevant documents? (2) Given the choice of the score distribution model, how to infer the model parameters when relevance information is not available?

In Chapter 3, we proposed modeling the relevant document scores by a mixture of Gaussians and the non-relevant document scores by a Gamma distribution. An intuition about the choice of the particular model from an IR perspective was given but the main approach was data-driven. We extensively studied the correlation between the number of Gaussian components $K$ in the mixture distribution and the value of the parameter $M$ in the Gamma distribution with a number of explanatory factors. The results of our experiments demonstrated that the distributions of $K$ and $M$ are independent both of the IR model used and the number of query terms. Further, we showed that a small number of relevant documents retrieved may prohibit the Variational Bayes framework to fit complex models such as the mixture of Gaussians to the relevant document scores. Furthermore, we demonstrated the utility of our model in inferring precision-recall curves.

In Chapter 4, we attempted to model score distributions in a rather systematic manner. We started with a basic assumption that query terms are generated via a Poisson process and induced that the distribution the relative term frequency in a document is a inverse Gamma distribution. Following the mathematical transformations applied on the relative term frequencies by two basic ranking functions, BM25 and Language Models, we derived the distribution of the produced scores, in an analytical form and illustrate that the derived distribution can be well approximated by a Gamma distribution. We also considered the score distribution
for relevant documents by relating score distributions with precision-recall curves. In particular, we adopted a precision-recall curve model that has previously been proposed and given this model we presented a general mathematical framework under which given any score distribution for all retrieved documents we can derive an analytical formula for the score distribution of relevant documents. The framework is general enough that the same derivations can be repeated for different models of precision recall curves. Finally, under the assumption that non-relevant documents follow a Gamma distribution for all retrieved documents, we show that there is a tendency of the derived distribution for the relevant documents to look Gaussian with a heavy right-hand tail.

In Chapter 5, we proposed a novel approach to infer the probability of document relevance through multiple sets of score distributions for different systems. Based on the assumption that the documents retrieved by different systems should have the same estimation of the probability of document relevance through different system dependent score distributions. We extended the popular EM algorithm by imposing this constraint. In the experiment, the score distributions estimated by new proposed extend EM clearly outperforms the one estimated by regular EM in terms of inferring precision-recall curves and estimating expected average precisions. We also demonstrated the use of these improved probabilities on the task of metasearch.

As for the future work, applying the proposed model in Chapter 3 and 4 to traditional applications such as information filtering, distributed retrieval or data fusion is still not straight forward. The fact that the use of a richer model than the traditional ones makes a blind fit harder. On the other hand, the extended EM algorithm proposed in Chapter 5 can be further improved such as utilizing a better way to calibrate the estimation of probability of document relevance from multiple systems, and generalizing the algorithm to work with those new score distribution models [11, 55].
Bibliography


