THE SMOOTHED DIRICHLET DISTRIBUTION: UNDERSTANDING CROSS-ENTROPY RANKING IN INFORMATION RETRIEVAL

A Dissertation Presented

by

RAMESH NALLAPATI

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

July 2006

Computer Science

© Copyright by Ramesh Nallapati 2006

All Rights Reserved

THE SMOOTHED DIRICHLET DISTRIBUTION: UNDERSTANDING CROSS-ENTROPY RANKING IN INFORMATION RETRIEVAL

A Dissertation Presented

by

RAMESH NALLAPATI

Approved as to style and content by:

James Allan, Chair

W. Bruce Croft, Member

Sridhar Mahadevan, Member

John Staudenmayer, Member

Thomas P. Minka, Member

W. Bruce Croft, Department Chair
Computer Science

DEDICATION

To Sri Sri Ravishankar.

ACKNOWLEDGMENTS

This thesis in its present form would not have been possible without the advice, support and help of many individuals who I feel extremely grateful to. Although words are very limited in expressing innermost feelings such as gratitude, I will make an effort below nevertheless, for the lack of a better medium of expression.

First and foremost, I would like to thank my advisor Prof. James Allan for guiding me through these turbulent six years of graduate study. I feel very grateful to him for keeping complete faith in my abilities even when I went through a rough patch in terms of my publications record, which is the hallmark of performance in a publish-or-perish culture. I also thank him for giving me the freedom to pursue the work that interested me most, even if it is sometimes not exactly in line with the general research direction of the lab. His ability to put things in proper perspective, his attitude of constantly keeping in mind the relevance and utility of a piece of work to the research area and his articulation of complex research ideas in terms of simple words and pictures have all been an eye-opener for me. In addition, I am also thankful to him for playing the role of a dependable counselor whenever I was low in spirits, confused, or simply clueless. I am also indebted to him for pointing out from time to time my weaknesses and potential areas to improve myself, which helped me a great deal in growing and maturing as a researcher. I also fondly recall several moments when his poker faced humor often caught me in two minds.

I am also extremely grateful to Dr. Tom Minka who played a significant role in shaping my thesis work. His outstanding technical expertise in machine learning, his attention to detail, his intuition behind why things work the way they do and his quest for perfection have made working with him a remarkable learning experience. I thank him sincerely for

his guidance and support over the last two years and for his readiness to take some time out and help despite his other pressing commitments.

I would like to express my gratitude to Prof. Bruce Croft and Dr. Stephen Robertson for their help and advice. I put them together in the same sentence because they are similar in many respects. Both are senior IR researchers, both are considered luminaries in the area and both have the remarkable ability to keep the voluminous IR research of the past on their finger tips. I thank them for their advice and suggestions on my thesis work without which this thesis would not have been complete.

I would also like to thank my other committee members Prof. Sridhar Mahadevan and Prof. John Staudenmayer for patiently going though the manuscript and offering helpful suggestions and comments which helped me improve the quality of this thesis.

The Center for Intelligent Information Retrieval offered me a friendly, interactive environment to learn and share ideas. I have vivid memories of the several lively discussions I had with my senior lab mate Victor Lavrenko, each of which taught me something new about the field. I will also preserve in my memory all the interactions I had with ex-lab mate Jeremy Pickens and all my current lab mates, especially Hema Raghavan, Fernando Diaz, Don Metzler, Jiwoon Jeon, Mark Smucker, Giridhar Kumaran, Vanessa Murdock, Yun Zhou, Ao Feng, Xiaoyong Liu, Ron Bekkerman and Xing Wei. I feel especially thankful (and even slightly apologetic) to colleague and friend Hema Raghavan for patiently putting up with me despite pestering her with innumerable silly questions almost on a daily basis.

I also thank our system administrator Andre Gauthier for offering prompt assistance whenever I ran into some trouble with the system. Kate Moruzzi, our lab secretary, has made my life much easier with her prompt disposal of all the paper work and her friendly demeanor. Likewise, I thank Pauline Hollister for her trademark radiant smile and her willingness to help with almost anything in the world. Sharon Mallory, our graduate pro-

gram manager, has made me almost forget there is any beaureaucratic work to do at all throughout my stay at UMass.

This acknowledgment would be incomplete if I failed to acknowledge the indirect but invaluable contribution of my friends to the successful completion of my graduate study. I made some very close friends during my very very long sojourn at UMass (often the inspiration for many a joke) and I feel very happy and thankful for their presence in my life. I will cherish and deeply value for my lifetime the friendship of Hema Raghavan, Purushottam Kulkarni (Puru) and Sudarshan Vasudevan (Suddu), my constant buddies for the last 5-6 years. All the fun and frolic in their company will be etched in my memory forever and will be dearly missed. I especially thank my friend and lab mate Hema for paying a patient ear to all my emotional outpourings, academic and otherwise, with kindness and understanding. As the saying goes, a friend in need is a friend indeed.

I would also like to thank many other people for their friendship and I am sure I will miss out on a few of them considering the duration of my stay at UMass. The names, not in any particular order, are Bhuvan Urgaonkar, my jogging and swimming buddy, Tejal Kanitkar, my ‘argument’ buddy, Ravi Tangirala, my tennis buddy, Pallika Kanani, my ‘home-cooked-food’ buddy, Kausalya Murthy, Sharad Jaiswal, Ravi Uppala, another ‘argument’ and ‘homework’ buddy, Ashish Deshpande, Pranesh Venugopal, Subhrangshu Nandi, Vijay Sundaram, my ‘PJ’ buddy and many more.

I will also fondly remember my fellow Art-of-Living volunteers and friends Akshaye Sikand, the PJ master and the ring leader, Harshal Deshmukh, the trouble maker, Debanti Sengupta, the princess, Denitza Stancheva, the ‘baibee’, Kishore Indukuri, the never-say-die hero and Ujjwala ‘Odwalla’ Dandekar. The youthful energy of the singing and dancing along with the good work we did together added a new dimension to my life and helped me grow as a person. I am also deeply grateful to Prof. Atul Sheel and Mrs. Rashmi Sheel, at whose home I enjoyed the comforts of my own home.

I thank my parents and family for standing by me in times of crisis and for being patient with my endless years of graduate study.

Last but not the least, my deepest gratitude goes out to Sri Sri Ravishankar, my spiritual master and the founder of the Art-of-Living foundation. Dispassionate yet loving, playful yet sincere, innocent yet wise, carefree yet compassionate, in the present moment yet infinitely deep, child-like yet mysterious, He is a beautiful and rare confluence of apparent contradictions and has been a source of tremendous strength and inspiration in my life. He taught me (and I am still learning) that success is measured by smile, not by wealth or achievements, that life is lived in the present moment, not in the past or future, that happiness lies in serving and sharing, not in accumulating, that lasting transformation happens through love and acceptance, not by revolt and rebellion and that to love is our very nature, not an act of barter. I owe all my positive qualities and my 'success' to His gift of spiritual knowledge, breathing and meditation techniques. I dedicate this thesis to the master as a token of deep felt gratitude.

This work was supported in part by the Center for Intelligent Information Retrieval, in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903, and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor.

ABSTRACT

Unigram Language modeling is a successful probabilistic framework for Information Retrieval (IR) that uses the multinomial distribution to model documents and queries. An important feature in this approach is the usage of the empirically successful cross-entropy function between the query model and document models as a document ranking function. However, this function does not follow directly from the underlying models and as such there is no justification available for its usage till date.

Another related and interesting observation is that the naïve Bayes model for text classification uses the same multinomial distribution to model documents but in contrast, employs document-log-likelihood that follows directly from the model, as a scoring function. Curiously, the document-log-likelihood closely corresponds to cross entropy, but to an asymmetric counterpart of the function used in language modeling. It has been empirically demonstrated that the version of cross entropy used in IR is a better performer than document-log-likelihood, but this interesting phenomenon remains largely unexplained.

One of the main objectives of this work is to develop a theoretical understanding of the reasons for the success of the version of cross entropy function used for ranking in IR. We also aim to construct a likelihood based generative model that directly corresponds to this cross-entropy function. Such a model, if successful, would allow us to view IR essentially as a machine learning problem. A secondary objective is to bridge the gap between the generative approaches used in IR and text classification through a unified model.

In this work we show that the cross entropy ranking function corresponds to the log-likelihood of documents w.r.t. the approximate Smoothed-Dirichlet (SD) distribution, a novel variant of the Dirichlet distribution. We also empirically demonstrate that this new

distribution captures term occurrence patterns in documents much better than the multinomial, thus offering a reason behind the superior performance of the cross entropy ranking function compared to the multinomial document-likelihood.

Our experiments in text classification show that a classifier based on the Smoothed Dirichlet performs significantly better than the multinomial based naïve Bayes model and on par with the Support Vector Machines (SVM), confirming our reasoning. In addition, this classifier is as quick to train as the naïve Bayes and several times faster than the SVMs owing to its closed form maximum likelihood solution, making it ideal for many practical IR applications. We also construct a well-motivated generative classifier for IR based on SD distribution that uses the EM algorithm to learn from pseudo-feedback and show that its performance is equivalent to the Relevance model (RM), a state-of-the-art model for IR in the language modeling framework that uses the same cross-entropy as its ranking function. In addition, the SD based classifier provides more flexibility than RM in modeling documents owing to a consistent generative framework. We demonstrate that this flexibility translates into a superior performance compared to RM on the task of topic tracking, an online classification task.

TABLE OF CONTENTS

Page

ABSTRACT . . . . . **ix**

LIST OF TABLES . . . . . **xiv**

LIST OF FIGURES . . . . . **xv**

CHAPTER

1. INTRODUCTION **1**

1.1 Motivation 2
1.2 Main Contributions 3
1.3 Notation and Definitions 4

1.3.1 Machine learning 5

1.3.1.1 Generative and Discriminative models 6

1.3.2 Generative models 6

1.3.2.1 The multinomial distribution 7
1.3.2.2 Maximum Likelihood Estimation 8
1.3.2.3 Generative models in IR 9

1.3.3 Rank equivalence and Class equivalence 10

1.4 Background 11
1.5 Overview of the thesis 15

2. THE CROSS ENTROPY FUNCTION **18**

2.1 Cross entropy ranking in Language Modeling for IR 18

2.1.1 Language models: estimation 19
2.1.2 Basic ranking function: Query Likelihood 19
2.1.3 Advanced ranking function: Cross entropy 20

2.1.4 Query-likelihood as cross entropy 22

2.2 Cross entropy in Text Classification 23
2.3 Cross entropy in IR as Text Classification 24
2.4 Anomalous behavior of Cross Entropy 26

3. THE SMOOTHED DIRICHLET DISTRIBUTION 29

3.1 Motivation: Dirichlet distribution and its rank-equivalence to TCE 29
3.2 Drawbacks of the Dirichlet distribution 33
3.3 Our solution: The Smoothed Dirichlet distribution 36

3.3.1 SD normalizer 36
3.3.2 Generative Process 37
3.3.3 Approximations to SD 39
3.3.4 Maximum Likelihood Estimation of Approximate SD parameters 41

3.3.4.1 Computationally efficient estimation 43

3.3.5 Inference using approximate SD distribution 44

3.4 Data analysis 45

4. TEXT CLASSIFICATION 52

4.1 Indexing and preprocessing 52
4.2 Evaluation 54
4.3 Parameter estimation 56
4.4 Inference 58
4.5 Results 63

4.5.1 Ranking:Reuters Collection 64
4.5.2 Classification: 20 Newsgroups and Industry Sector 65
4.5.3 Comparison with previous work 68

4.6 Conclusions 69

5. INFORMATION RETRIEVAL 70

5.1 Relevance model for ad hoc retrieval 71
5.2 SD based generative classifier 72

5.2.1 Approximations 75

5.3 Experiments 80

5.3.1 Models considered 80
5.3.2 Data sets 80
5.3.3 Evaluation measures 81
5.3.4 Training the models 82

5.4 Results and Discussion 83

6. TOPIC TRACKING: ONLINE CLASSIFICATION 89

6.1 Data sets 90
6.2 Evaluation 91
6.3 Models considered 92

6.3.1 Vector Space model 93
6.3.2 Relevance model 94
6.3.3 Naïve Bayes classifier 95
6.3.4 SD based classifier 97

6.4 Results 98

7. CONCLUSIONS 102

7.1 Future work 103

BIBLIOGRAPHY 109

LIST OF TABLES

Table **Page**

2.1 Comparing Language modeling with naïve Bayes classifier 27

4.1 Performance comparison on Reuters Corpus 64

4.2 Performance comparison on the 20 News groups and Industry sector data sets: subscripts reflect significance as described at the start of section 4.5.2 (Statistical significance results are identical w.r.t. the T-test and the sign test.) 66

5.1 Comparison of RM and SD 78

5.2 Data Collections 80

5.3 Performance comparison of various models on 4 different TREC collections: the values of Mean Average Precision (MAP) and Precision at 5 documents (Pr(5)) are in percentage. Significance results are reported only between RM and SD in the table. Bold faced numbers indicate statistical significance w.r.t the Wilcoxon test as well as a paired two tailed T-test at 95% confidence level. Note that the sign test did not indicate any significant differences between the two models on any of the runs. 84

6.1 Statistics TDT data 91

6.2 Comparison of performance at various levels of pseudo feedback: N is the number of pseudo feedback documents and bold faced indicates best performing model for the corresponding run. The superscript * statistical significance compared to the nearest model w.r.t. the paired one-tailed T-Test at 90% confidence interval. Note that the sign test at 95% confidence level did not indicate any significant differences. 99

6.3 Number of topics of the 29 test topics on which SD outperforms RM at various levels of pseudo feedback 99

LIST OF FIGURES

Figure **Page**

1.1 Graphical representation of the Multinomial distribution 8

3.1 Graphical representation of the DCM distribution: A multinomial $\boldsymbol{\theta}$ is sampled from the Dirichlet distribution from which words are repeatedly sampled in an IID fashion to generate the document 31

3.2 Graphical representation of the Dirichlet and SD distributions: A multinomial $\boldsymbol{\theta}$ representing the document is sampled directly from the Dirichlet (SD) distribution. 31

3.3 Domain of smoothed proportions Δ^s for various degrees of smoothing: dots are smoothed language models $\boldsymbol{\theta}^D$ and the triangular boundary is the 3-D simplex. 35

3.4 (a) Comparison of the normalizers (b) Gamma function and its approximators 40

3.5 Comparison of predicted and empirical distributions 49

5.1 Dirichlet distribution has lower variance for higher values of precision S 79

5.2 Comparison of precision recall curves on AP training queries 86

5.3 Comparison of precision recall curves on AP test queries 87

5.4 Comparison of precision recall curves on WSJ queries 87

5.5 Comparison of precision recall curves on LAT queries 88

5.6 Comparison of precision recall curves on FT queries 88

6.1 A comparison of DET curves for various models for the case where 5 pseudo feedback documents are provided for feedback. 101

7.1 Volume of S_{MLE} in two dimensional case 107

CHAPTER 1

INTRODUCTION

Considering the vast amounts of textual information available on the internet today, retrieving information relevant to a user's specific information need from large collections of documents such as the world-wide-web, scientific literature, medical data, news repositories, internal databases of companies, etc. has become a great challenge as well as a pressing need.

Modern search engines such as *Google* and *Yahoo* have made efficient access to relevant information possible through keyword based search on the web. The basic idea is to pre-index the entire web and quickly retrieve a ranked list of documents that match the user's key words based on some weighted ranking functions.

The Information Retrieval (IR) research community has addressed the same challenge of efficient information access from large collections through the formal problem of *ad hoc retrieval* defined by the Text REtrieval Conference (TREC) [1]. Ad hoc retrieval, also sometimes called query based retrieval, is considered a core research problem in IR and can be defined as the problem of retrieving a ranked list of documents relevant to a query from a large collection of documents. The TREC research community has developed certain standardized test beds and objective evaluation metrics to measure the quality of retrieval algorithms [13] on this task. These standardized test beds and metrics have permitted repeatability of retrieval experiments and objective comparison of retrieval algorithms thereby fostering active research in this area for more than a decade now.

[1] http://trec.nist.gov

The main focus of the ad hoc retrieval task is to improve retrieval effectiveness by better modeling of document content and the information need. Search engines on the other hand leverage a variety of other features such as link structure, anchor text, manual labeling, page design, etc. to retrieve relevant documents.

1.1 Motivation

In the problem of ad hoc retrieval, a query specific ranking function is typically employed to rank documents in decreasing order of relevance. The choice of the ranking function is critical to the performance of the IR system. In this work, we will investigate *cross-entropy* ranking, an empirically successful ranking function employed in a popular probabilistic approach to ad hoc retrieval called *language modeling*. We will present the motivating reasons for this investigation in detail in the following chapter, but we summarize them briefly below.

1. Cross-entropy ranking in the language modeling approach does not follow directly from the underlying model - its choice seems to be influenced more by its empirical success than by modeling considerations. Additionally, because cross-entropy is an asymmetric function, it naturally provides two choices for ranking and it has been empirically found that the popular version performs significantly better than its asymmetric counterpart. However, no theoretical explanation has been available for this phenomenon.

2. Ad hoc retrieval is closely related to text classification[2], however the models employed on these tasks have significant differences in their inference techniques. The closest counterpart to language modeling is the naïve Bayes classifier in text classification. Although both of them use the multinomial distribution to model topics, the

[2]The problem of classifying documents into pre-defi ned categories.

classification function used by the former corresponds to the asymmetric counterpart of the cross-entropy employed for ranking in IR.

The primary aim of our investigation is to understand the reasons behind the empirical success of the particular form of cross-entropy ranking and to construct a generative model that directly explains this cross-entropy function in terms of likelihood of documents w.r.t. the model. In addition, we also aim to translate the success of cross-entropy in IR to text classification through the new model.

In the larger context, one of our aims is to construct a principled likelihood based generative model [3] for information retrieval thereby allowing IR to be viewed essentially as a machine learning problem. Our other larger objective is to bridge the gap between text classification and information retrieval through a unified machine learning model to both the problems. We hope that our work motivates researchers enough to apply more sophisticated machine learning techniques to IR problems in the future.

1.2 Main Contributions

Without going into details at this point, the following is the summary of the main contributions of this thesis.

1. The first and foremost contribution of our work is our justification of the cross entropy function, hitherto used in an ad hoc manner in information retrieval. Our justification consists of the following two important results:

    - We show that the cross entropy ranking function corresponds to the log-likelihood of documents w.r.t. the approximate Smoothed-Dirichlet (SD) distribution, a novel variant of the Dirichlet distribution.

[3] We will describe what this means in section 1.3.2.3.

- We also empirically demonstrate that this new distribution captures term occurrence patterns in documents much better than the multinomial.

2. Our approximations to the Smoothed Dirichlet distribution result in a closed form solution for maximum likelihood estimation, making the approximate distribution as efficient to estimate as the multinomial, but superior in modeling text.

3. We demonstrate that the success of the specific version of cross-entropy used in IR can also be translated to text classification through a new generative classifier based on the SD distribution as shown by its superior performance compared to all other existing generative classifiers in the classification task.

4. We show that the same SD based generative classifier is also successful in ad hoc retrieval and is on par with the state-of-the-art Relevance Model, demonstrating the applicability of likelihood based machine learning approaches for information retrieval. We also show that the consistent generative framework of the SD based model overcomes the document weighting problems of the Relevance Model in the task of topic tracking and outperforms the latter in a pseudo-feedback setting.

5. We argue that the success of the SD based generative classifier on various tasks of information retrieval shows that a unified perspective of IR as a binary classification problem is a promising framework, provided we use an appropriate distribution in the generative model.

1.3 Notation and Definitions

Before we present the background of our work, we will first present the notation used in this thesis and some useful definitions.

We will adopt the following standard notation in this thesis. We represent vectors in bold such as $\mathbf{f}$, $\boldsymbol{\theta}$, etc., and their scalar components by regular faced letters such as f_j, θ_j

etc. We will use subscripts to represent the index number of a component of a vector and superscript to denote the object name/number. For example θ_j^D represents the jth component of the vector $\boldsymbol{\theta}^D$ where the superscript ^{D} indicates that the vector represents the document D. Similarly, $\boldsymbol{\theta}^i$ denotes a vector $\boldsymbol{\theta}$ corresponding to the ith document in an indexed repository. We will use the letter j to indicate the index number of a word and i to represent the index number of a document in a repository. We use V to represent the vocabulary size of an indexed document collection and $\mathcal{C}$ to represent the number of documents in the collection. The symbols Q and D denote a query and a document respectively.

We also typically use the vector $\mathbf{f}^D$ to denote the counts representation of a document D, where each component f_j^D represents the number of times the jth word occurs in the document. We also use $|D|$ to represent the length of a document D, which is equal to $\sum_{j=1}^{V} f_j^D$, the sum of the occurrence counts of all words in the document. Likewise, $|Q| = \sum_{j=1}^{V} f_j^Q$ is used to denote the length of a query Q and $|C| = \sum_{i=1}^{\mathcal{C}} \sum_{j=1}^{V} f_j^i$ to denote total number of word occurrences in the collection. Sometimes we skip the limits of summation, as in, we use the short notation $\sum_j$ for $\sum_{j=1}^{V}$ where it is clear from context.

Note that sometimes we use the phrases ad hoc retrieval and IR interchangeably. Considering the centrality of the ad hoc retrieval task to information retrieval, we hope this abuse of notation is not a major offense. We assure the reader that the meaning will be clear from the context.

1.3.1 Machine learning

As a broad subfield of artificial intelligence, Machine learning is concerned with the development of algorithms and techniques that allow computers to learn and draw inferences on data. There are broadly two classes of machine learning approaches called generative and discriminative models as described below.

1.3.1.1 Generative and Discriminative models

Generative models are those that explicitly model data using a generative distribution $P(x|\theta)$ where x is a data point and θ is the parameter of the generative distribution.

If the task is to classify the data into one of several categories, the typical approach is to estimate a distribution θ^T for each category T and then classify the data point x into the category that maximizes the posterior probability of the category $P(T|x)$. This is estimated using the Bayes' rule as shown below.

$$T_{best} = \arg\max_T P(T|x) = \arg\max_T \frac{P(X|\theta^T)P(T)}{P(x)} \tag{1.1}$$

Note that the posterior probability is expressed as a product of the class-conditional likelihood of the data $P(X|\theta^T)$ and the class-prior $P(X)$. Such models are called *generative classifiers*. One of the best examples of generative models for text is the naïve Bayes classifier [31].

Discriminative approaches on the other hand do not model data explicitly. Instead, they build a classifier directly by estimating a decision rule $f(T, x)$ or a probability $P(T|x)$ directly from training data. Examples of discriminative models for text include SVMs [16], maximum entropy models [37], etc.

Both generative and discriminative approaches have their own advantages and disadvantages and the choice of the approach is usually governed by the school of thought one comes from. For example, Ng and Jordan have done a detailed investigation on the naïve Bayes and maximum entropy classifiers for text [36] and found that both have desirable properties under certain conditions.

1.3.2 Generative models

We choose the generative framework in our work, but our choice is not so much influenced by the analysis of the differences between these two approaches as it is by the fact that we are interested in explaining the ranking function used in language modeling, which

is a generative model. In this subsection, we discuss the details of generative models using the multinomial distribution as a running example.

1.3.2.1 The multinomial distribution

In the recent past, the multinomial has become the *de facto* distribution in any generative model for text, be it in classification [31], ad hoc retrieval [26], document clustering [63] or topic modeling [5]. We will deal with this distribution extensively throughout this thesis.

The multinomial probability distribution, whose parameters are represented by $\boldsymbol{\theta}$ is a distribution over a discrete random variable X that can take any value in the range $\{1, \cdots, V\}$. One can imagine the jth value of the random variable to correspond to the word w_j in the vocabulary. The probability that the random variable takes the value j, or in other words, the probability of generating the word w_j from the multinomial distribution is given by the jth component θ_j of the parameter vector $\boldsymbol{\theta}$ as shown below.

$$P(w_j|\boldsymbol{\theta}) = \theta_j \tag{1.2}$$

The probability distribution $\boldsymbol{\theta}$ has the following property:

$$\forall_j \theta_j \geq 0; \ \sum_j \theta_j = 1 \tag{1.3}$$

One can also generate strings of words from the multinomial distribution by sampling words one at a time, in an independent and identically distributed (IID) manner. In such a case, the probability of the particular string is simply given the product of the probabilities of each of the words in the string. Mathematically, the probability of generating a particular string Q that has a counts representation $\mathbf{f}^Q$ from this distribution is given by:

$$P(Q|\boldsymbol{\theta}) = \prod_{j=1}^{V} (\theta_j)^{f_j^Q} \tag{1.4}$$

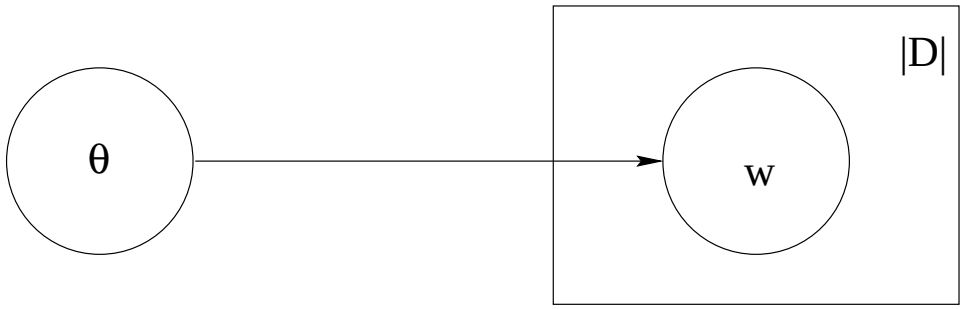

Figure 1.1. Graphical representation of the Multinomial distribution

For clarity, it is common to illustrate the generative process of a model using a graphical representation. The graphical representation of the generative process from a multinomial distribution is shown in figure 1.1. Each circle in the figure represents a random variable and the arrow represents dependency. A Plate represents repeated IID sampling and the number at the right hand corner of the plate indicates the number of times the sampling is done. As the graph shows, words are repeatedly sampled in an IID fashion from the underlying multinomial $\boldsymbol{\theta}$ to generate the document.

1.3.2.2 Maximum Likelihood Estimation

In machine learning, it is typical to estimate the parameters of a distribution using the maximum likelihood estimation. We will use the hat notation $\hat{\boldsymbol{\theta}}$ to represent the *Maximum Likelihood Estimate (MLE)* of a probability distribution. One can perform maximum likelihood estimation for any distribution, but in this subsection, we will illustrate it using the multinomial distribution again as a running example. As the name indicates, the MLE of a distribution from a set of examples $\mathcal{D} = \{D_1, \cdots, D_N\}$ corresponds to the parameter setting that maximizes the log-likelihood of the set given the parameters of the distribution as shown below:

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \arg\max_{\boldsymbol{\theta}} \log \prod_{i=1}^{N} P(D_i|\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \sum_i \sum_j f_j^i \log \theta_j \end{aligned} \tag{1.5}$$

where we assumed the multinomial distribution in step (1.5). If the log-likelihood function is convex, there exists a unique globally optimal solution for MLE and it corresponds to the point where the derivative of the log-likelihood function w.r.t. the parameter vector $\boldsymbol{\theta}$ vanishes:

$$\forall_j \; \frac{\partial \left(\sum_i \sum_k f_k^i \log \hat{\theta}_k \right)}{\partial \hat{\theta}_j} = 0 \tag{1.6}$$

The above relation coupled with the constraint that $\hat{\boldsymbol{\theta}}^Q$ satisfies the property in (1.3) can be solved using the Lagrange Multiplier technique [6]. We will not go into the details of the technique, but it suffices to say that it results in the following solution.

$$\hat{\boldsymbol{\theta}} = \frac{\sum_i \mathbf{f}^i}{\sum_i |D_i|} \tag{1.7}$$

Thus the MLE solution for the multinomial distribution corresponds to normalized counts of words in the example.

We will use $\hat{\boldsymbol{\theta}}^{GE}$ to represent the multinomial distribution of words in General English, where the hat represents that it is obtained as an MLE from the entire collection, given by $\hat{\boldsymbol{\theta}}^{GE} = \frac{\sum_{i=1}^{\mathcal{C}} \mathbf{f}^i}{|C|}$.

1.3.2.3 Generative models in IR

In information retrieval research, one typically encounters two types of generative models. The first way, which we call *likelihood based generative models*, uses the likelihood of the data to drive parameter estimation and inference (ranking). Generative classifiers are a good example for likelihood based generative models. These models use likelihood

of training data as an objective function to estimate their parameters as described in section 1.3.2.2. For inference, they use the Bayes' rule, which is expressed in terms of the class-conditional likelihood of the test data as shown in section 1.3.1.1.

The second way uses heuristic methods or other task-specific criterion for estimation and inference instead of the likelihood. Many probabilistic IR systems that seem to use generative models, such as BM25 [47], are actually using them only heuristically. Other techniques such as the Relevance model [26] choose their estimation and inference algorithms based on a series of theoretical assumptions or on empirical basis.

In this work, we are more interested in likelihood based generative models than the heuristic based methods. One important reason is that it allows us to compare different models based on the likelihood function on unseen data without resorting to evaluating them on external tasks. Secondly, likelihood is a meaningful function that can be applied to various tasks such as classification and ranking as we will show in the future and hence allows the model to generalize over various tasks.

1.3.3 Rank equivalence and Class equivalence

While comparing different ranking functions, we will extensively use the symbol $\overset{rank}{=}$ to denote rank-equivalence. Two ranking functions are rank-equivalent if the relative ordering of any pair of documents for a given query is the same w.r.t. both the functions. Throughout this thesis, we will make simplifications of ranking functions using the rank equivalence relation. For example, if a ranking function $f(T, D)$ that is a function of topic T and document D can be factored as $f(T, D) = g(T, D) + h(T) + l(D)$, one can simplify the function using rank-equivalence relationship as follows.

$$f(T, D) = g(T, D) + h(T) + l(D) \tag{1.8}$$

$$\overset{rank}{=} g(T, D) + l(D) \tag{1.9}$$

since $h(T)$ is independent of documents and will not influence the relative order of documents.

Likewise, in the context of text classification, we will use the symbol $\overset{class}{=}$ to indicate class-equivalence. Two classifiers are class-equivalent if both assign any given document to the same class. For example, using class equivalence, one can simplify $f(T, D)$ as follows:

$$f(T, D) = g(T, D) + h(T) + l(D) \tag{1.10}$$

$$\overset{class}{=} g(T, D) + h(T) \tag{1.11}$$

since $l(D)$ is independent of the class and hence will not influence the choice of the class for a given document.

1.4 Background

As mentioned in section 1.1, our main objective is to explain the successful cross-entropy ranking function in IR by constructing a generative model that directly explains it in terms of a likelihood function. We will present the cross-entropy function in the context of information retrieval in more detail in chapter 2. In this section, we discuss the background of one of our larger objectives mentioned earlier, namely, constructing a principled, empirically successful, likelihood based generative approach for information retrieval.

Applying machine learning based techniques to information retrieval is not a new idea. However hardly any of these approaches have been as successful as other heuristic based approaches in this domain.

One of the earliest machine learning models for IR is the Binary Independence Retrieval (BIR) model which considers ad hoc retrieval as a binary classification task [45]. This model used the multiple Bernoulli distribution shown in (1.12) as the underlying generative model.

$$P(\mathbf{x}|\boldsymbol{\beta}) = \prod_{j=1}^{V} \beta_j^{x_j}(1-\beta_j)^{1-x_j} \tag{1.12}$$

where $\mathbf{x}$ is a vector of binary random variables, each of which takes the values $x_j \in \{0, 1\}$ indicating the presence or absence of a term w_j in the document and $\boldsymbol{\beta}$ is the parameter vector. The BIR model uses the binary classification framework with class parameters $\boldsymbol{\beta}^R$ and $\boldsymbol{\beta}^N$ for relevance and non-relevance classes respectively. The posterior probability of relevance of a document using this distribution can be shown to be rank equivalent to the following:

$$P(R|\mathbf{x}) \overset{rank}{=} \prod_{j=1}^{V} \left(\frac{\beta_j^R}{\beta_j^N}\right)^{x_j} \left(\frac{1-\beta_j^R}{1-\beta_j^N}\right)^{1-x_j} \tag{1.13}$$

The main drawback of the BIR model is that its underlying generative distribution, namely the multiple-Bernoulli, ignores term frequencies and models only the presence or absence of terms in documents. To overcome this shortcoming, Robertson *et al*[44] proposed using the Poisson distributions to model term counts instead. They use a mixture of two Poissons instead of just one in order to model what they call 'eliteness' of terms. The two Poisson mixture components model elite and non-elite classes that represent the importance (or the lack of it) of a particular word in a given document. The probability of a document $\mathbf{f}^D$ w.r.t. this model is given by:

$$P(\mathbf{f}^D|\boldsymbol{\lambda}, \boldsymbol{\mu}) = \prod_{j=1}^{V} \left(p_j \frac{e^{-\lambda_j}\lambda_j^{f_j}}{f_j!} + (1-p_j)\frac{e^{-\mu_j}\mu_j^{f_j}}{f_j!}\right) \tag{1.14}$$

where λ_j and μ_j are the means of the Poisson mixture components for the word w_j in the vocabulary and p_j is its probability of occurrence in the elite class in document D. Robertson *et al* constructed a binary classifier for ad hoc retrieval using the two component Poisson mixture model. They assume the Poisson parameters $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the same for the relevant and non-relevant classes, but they assume different mixture probabilities $\mathbf{p}^R$ and

$\mathbf{p}^N$ for the relevant and non-relevant classes respectively. However they did not achieve any performance improvements compared to the BIR classifier. In [47], the authors attribute this lack of success to the estimation technique as well as the complexity of the model: there are 4 parameters namely λ_j, μ_j, p_j^R and p_j^N for each word in the vocabulary. Despite its lack of empirical success, the 2-Poisson model inspired a successful model called the BM25 [47], a heuristic based model that closely mimics the functional form of 2-Poisson and is still considered as an excellent baseline model in IR experiments.

There are also other advanced models that did not achieve much empirical success. Notable among them is the dependence tree model developed by van Rijsbergen [54] that relaxes the assumption of term independence made by the BIR model by capturing the most significant dependencies between words in a document. This model inspired other probabilistic models [35] later with a small improvement in performance on other tasks of information retrieval.

More recently, it has been shown by McCallum and Nigam [31] that the naïve Bayes classifier based on the multinomial distribution shown in (1.4) outperforms the multiple-Bernoulli based BIR classifier on the task of text classification, by capturing the extra information contained in term frequencies. Applying the multinomial based naïve Bayes to ad hoc retrieval, it can be shown that (the interested reader may read section 2.3 for details) the posterior probability of relevance of a document D represented as a counts vector $\mathbf{f}^D$, given two parameter vectors $\boldsymbol{\theta}^R$ and $\boldsymbol{\theta}^N$ for relevant and non-relevant classes respectively is given by

$$P(R|\mathbf{f}^D, \boldsymbol{\theta}^R, \boldsymbol{\theta}^N) \overset{rank}{=} \sum_j f_j \log \frac{\theta_j^R}{\theta_j^N} \tag{1.15}$$

The name 'naïve Bayes' refers to the 'naïve ' assumption that the features (words) occur independently of each other. In general, the naïve Bayes can use any underlying distribution. Note that in this work, when we use the term naïve Bayes classifier, we actually refer to the multinomial based naïve Bayes classifier.

Despite its improvement in performance w.r.t. the BIR classifier, the naïve Bayes classifier itself is found to be a poor performer on ad hoc retrieval compared to the more traditional vector space models [51], owing to poor modeling of text by the multinomial distribution. We will discuss this in more detail in chapter 3.

An an alternative, the language modeling approach has been proposed, which views queries and documents as multinomial distributions over the vocabulary called *language models* and ranks documents by their proximity to the queries as measured by the cross entropy function. This framework has proved very attractive and models based on this framework have been quite successful empirically [26, 59]. Although language modeling can be considered a generative approach, its estimation and inference techniques are not likelihood based.

There are also other machine learning based approaches for ad hoc retrieval in the discriminative framework. For example, in a series of publications, Cooper, Kantor and Gey [8, 19, 18, 12] apply the maximum entropy approach to ad hoc retrieval in which they use a binary classification approach. They learn weights of certain generalizable feature functions based on past queries and their corresponding relevant documents provided by the user. The learned model is then applied to rank documents in future queries. Following a similar approach, Nallapati applied Support Vector Machines [34] for ad hoc retrieval. One of the difficulties of these approaches is that there is no theoretical guidance in defining the feature functions, which are often defined in a heuristic manner.

In the recent past, a new class of generative models called *topic models* have been proposed [14, 5, 56, 4, 27]. These models relax the assumption of single topic per document [4] and model documents as being generated from a mixture of topics. These models can learn the topic structure in a collection of documents automatically and can also identify the topics discussed in a document through their learning and inference mechanisms respectively.

[4]Most previous generative models except the 2-Poisson model made this assumption.

Although topic models have proven very effective in discovering topic structure within a large document collection, they have not yet been shown to consistently outperform simpler and more traditional models such as naïve Bayes and SVMs on information retrieval tasks such as ad hoc retrieval and text classification.

Another emerging machine learning based approach for text modeling that has gained popularity in the recent past is the area of manifold learning, which deals with modeling non-linear high dimensional observation/model space. In the domain of text, Lafferty and Lebanon[21] presented a new diffusion kernel for the multinomial manifold mapped to a spherical geometry via the Fisher information metric. Their experiments showed that an SVM employing the new diffusion kernel yield significant performance improvements compared to the one using a linear or a Gaussian kernel. Other kernels have also been proposed exploiting the geometry of the multinomial manifold and have proven empirically successful [61]. Our work is also related to manifold learning in a subtle way as we will show in section 3.3.1.

In our work, we are mainly concerned with constructing a simple generative classifier for ad hoc retrieval that is based on document likelihood for its estimation as well as inference. As such our contribution is at a fundamental level, i.e., on choosing an appropriate distribution for text. We believe our work could be a precursor to new sophisticated topic modeling and manifold learning approaches.

1.5 Overview of the thesis

In chapter 2, we will introduce the language modeling approach to information retrieval and describe the cross entropy ranking function in information retrieval, pointing out its differences from the classification function used by its closest counterpart in text classification, the naïve Bayes classifier.

We present the new Smoothed Dirichlet distribution and its approximations in chapter 3. In addition, this chapter shows the correspondence of this distribution to the cross entropy

function and also demonstrates empirically that the distribution models text much better than the multinomial, which is considered a *de facto* distribution for text.

We define a simple generative classifier based on the SD distribution in chapter 4. Our results on three different test beds show that the SD based classifier significantly outperforms other generative classifiers for text.

Generative classifiers applied to the task of ad hoc retrieval have not performed well in the past, primarily owing to incorrect choice of the generative distribution. In chapter 5, we apply the SD based generative classifier to ad hoc retrieval. The classifier models pseudo feedback using the Expectation Maximization algorithm. Our results on four different TREC collections show that the model matches the performance of the Relevance Model, a state-of-the-art model for ad hoc retrieval, justifying our view of ad hoc retrieval as a classification problem. We also compare and contrast the SD based classifier and the Relevance Model. Our analysis shows that the highly effective query-likelihood based document weighting used in the Relevance model can be explained as a self adjusting mechanism of the variance of the SD distribution.

In chapter 6, we implement the SD based classifier to the online task of topic tracking in a pseudo feedback setting and compare its performance with that of the naïve Bayes classifier, Relevance Model and the vector space model. We adapt the learning algorithms of all the models to an online setting. Our results on TDT 2004 topics show that the SD classifier not only outperforms all the other models but also is relatively more robust to noise than the other models.

Chapter 7 summarizes this thesis with some concluding remarks and discussion on future research directions.

In chapter 3, we assumed one-to-one correspondence between the probability *density* of the smoothed language model representation of a document and probability *mass* of its bag-of-words representation for mathematical simplicity. In the appendix, we relax this as-

sumption and present a loose upper bound estimate of the probability mass of the document in terms of the probability density of its corresponding smoothed language model.

CHAPTER 2

THE CROSS ENTROPY FUNCTION

In this chapter, we will first introduce the language modeling framework for IR. We will also present the cross entropy ranking function and then discuss the motivating reasons behind our investigation in more detail.

2.1 Cross entropy ranking in Language Modeling for IR

Language modeling is a probabilistic framework for information retrieval that has become popular in the IR community in the recent past owing to its attractive theoretical framework [38, 22, 60]. It has also been empirically successful, achieving performance comparable to or better than the traditional vector space models [50, 49, 47]. There are several modeling variations to this approach, but the simplest and the one of the most effective models is the unigram approach in which each document D is modeled as a multinomial distribution $\boldsymbol{\theta}^D$ over the vocabulary V, called the *document language model* that represents the topic of the document [1]. Given a multinomial document language model, one can generate strings of text from the model, by randomly sampling words from the distribution as described in section 1.3.2.1 and estimate their probability using (1.4). Given the generative process of the multinomial distribution, it follows that the unigram language model assumes that words are generated independently.

[1]Clearly, this approach makes the modeling assumption that each document is about a single topic only or that one distribution can model all the topics.

2.1.1 Language models: estimation

One would expect that the language model assigned to each document is its MLE distribution, namely $\boldsymbol{\theta}^D = \hat{\boldsymbol{\theta}}^D$, where $\hat{\boldsymbol{\theta}}^D$ is obtained by maximizing the log-likelihood of the document $\log P(D|\boldsymbol{\theta})$ w.r.t. the parameter vector $\boldsymbol{\theta}$. However, in practice, we smooth the MLE distribution with the general English distribution to obtain the document language model as shown below.

$$\boldsymbol{\theta}^D = \lambda\hat{\boldsymbol{\theta}}^D + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE} = \lambda\frac{\mathbf{f}^D}{|D|} + (1-\lambda)\frac{\sum_{i=1}^{\mathcal{C}}\mathbf{f}^i}{|C|} \tag{2.1}$$

where $0 < \lambda < 1$ is a smoothing parameter that is used to smooth the MLE distribution of the document $\hat{\boldsymbol{\theta}}^D$ with the general English MLE distribution $\hat{\boldsymbol{\theta}}^{GE}$. Smoothing is done to force non-zero probability for all words in the vocabulary. The form of smoothing used in (2.1) is called Jelinek-Mercer smoothing. Jelinek Mercer smoothing achieves good performance on most IR tasks and as such, we will use this form of smoothing in this thesis. An extensive empirical study of smoothing techniques including the Jelinek Mercer, Dirichlet and Laplacian can be found in [60].

2.1.2 Basic ranking function: Query Likelihood

In the basic language modeling approach, given a query $Q \equiv \mathbf{f}^Q = \{f_1^Q, \cdots, f_V^Q\}$, where each f_j^Q is the count of the jth vocabulary word in the query, documents are ranked according to the likelihood that their respective language models generate the query as shown below.

$$\text{score}(D|Q) = P(\mathbf{f}^Q|\boldsymbol{\theta}^D) = \prod_{j=1}^{V}(\theta_j^D)^{f_j^Q} \tag{2.2}$$

The idea is that if a document is on the same topic as the query, then the document's language model is likely to generate the query with high probability.

Unlike previous machine learning approaches discussed in section 1.4 that modeled relevant and non-relevant classes and then ranked documents using the posterior probability of relevance, the query-likelihood model shifts the focus of modeling to the document side. As long as one uses information from only the query to model the relevance class, the query-likelihood model makes more sense than document likelihood. There are two reasons for this: firstly, queries tend to be very short while documents are typically longer and contain more information and hence it is easier to model them. Secondly, and more importantly, the query-likelihood model is a special case of the empirically successful cross-entropy function as we will show in section 2.1.4.

One main difficulty with the ranking by query-likelihood is that it models generation of only the query terms from the documents. Since key-word queries are very short and concise, basic query likelihood ranking may not retrieve all the relevant documents. For illustration, consider an example query 'cars'. The query likelihood ranking will not retrieve documents that contain the word 'automobile' although we know that they could be potentially relevant to the query on account of the similarity of the two words. In IR parlance, this is called the *synonymy* problem and arises out of the inherent ambiguity of natural language.

2.1.3 Advanced ranking function: Cross entropy

One way to overcome the synonymy problem is to shift some of the modeling effort back to the query, which is what the advanced language modeling techniques such as the Relevance models do. Since user feedback is typically not available in the context of ad hoc retrieval, query modeling is more complex than modeling topics in the task of text classification. Almost all models including the vector space models and language modeling approaches employ a three step process to model the query as outlined below:

1. An initial retrieval is first performed using a simple query term matching technique such as query likelihood.

2. The original query is then expanded by appending related terms borrowed from top-ranking documents.

3. A re-retrieval is performed using the expanded query.

This technique is called *query expansion* using pseudo relevance feedback [55]. In the above example, one expects the word 'automobile' to be added to the query after the initial retrieval. Consequently, the second retrieval is expected to retrieve documents that contain the term 'automobile' too.

In advanced language models such as Relevance Models [26], query expansion with pseudo feedback is modeled as estimating a probability distribution $\boldsymbol{\theta}^R$ for the query's topic over the entire vocabulary. This distribution, known as the *relevance model*, is estimated from the top ranking documents obtained from the query likelihood ranking of (2.2). A second retrieval is then performed in which documents are ranked according to the negative KL-divergence ranking [26, 22] as shown below:

$$\begin{aligned} \text{score}(D|Q) = -KL(\boldsymbol{\theta}^R||\boldsymbol{\theta}^D) &= -\sum_j \theta_j^R \log \frac{\theta_j^R}{\theta_j^D} \\ &\overset{rank}{=} \sum_j \theta_j^R \log \theta_j^D \\ &= -H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^D) \end{aligned} \tag{2.3}$$

where $KL(\boldsymbol{\theta}^R||\boldsymbol{\theta}^D)$ is the KL-divergence, $H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^D)$ is the cross-entropy and the symbol $\overset{rank}{=}$ indicates rank-equivalence. Since the term $\sum_j \theta_j^R \log \theta_j^R$ in the KL-divergence formula is document independent, it does not influence ordering of the documents, implying rank-equivalence of KL-divergence to cross-entropy. Hence, although the literature mentions KL-divergence as the ranking function, we will refer to only the cross-entropy function in this work for reasons of simplicity.

Cross-entropy is an information theoretic metric that measures the distance between two distributions. Its domain is the set of all non-negative real numbers and is minimum

when the two distributions are identical. Since we rank documents in the decreasing order of negative-cross entropy (see (2.3)), documents whose language models are closest to that of the query are ranked highest.

2.1.4 Query-likelihood as cross entropy

Interestingly, the query likelihood function of the basic language modeling approach shown in (1.4), can also be shown to be rank-equivalent to a cross-entropy function as shown below:

$$
\begin{aligned}
P(\mathbf{f}^Q|\boldsymbol{\theta}^D) \overset{rank}{=} \log P(\mathbf{f}^Q|\boldsymbol{\theta}^D) &= \sum_{j=1}^{V} f_j^Q \log \theta_j^D && (2.4)\\
&= |Q| \sum_{j=1}^{V} \frac{f_j^Q}{|Q|} \log \theta_j^D \overset{rank}{=} \sum_{j=1}^{V} \frac{f_j^Q}{|Q|} \log \theta_j^D && (2.5)\\
&= \sum_{j=1}^{V} \hat{\theta}_j^Q \log \theta_j^D = -H(\hat{\boldsymbol{\theta}}^Q, \boldsymbol{\theta}^D) && (2.6)
\end{aligned}
$$

where (2.4) follows from the fact that logarithmic function is monotonic w.r.t. its input and hence does not alter the rank-order of documents. Similarly, (2.5) is valid since dividing a ranking function by a term $|Q|$ that depends only on the query will preserve the rank-order of documents. Thus the query-likelihood function also corresponds to cross-entropy ranking but the estimate of the Relevance model $\hat{\boldsymbol{\theta}}^R$ in this case corresponds to the MLE of the query given by $\hat{\boldsymbol{\theta}}^Q = \frac{\mathbf{f}^Q}{|Q|}$ [22].

We had mentioned earlier that the query-likelihood model takes a document-centric approach, where the modeling effort is spent on the documents and queries are generated from the respective document models. Since query-likelihood suffers from the synonymy problem, advanced language models return to the query-centric approach used by earlier classification models by modeling queries as multinomial distributions, the only difference being that language modeling is not a classification model and uses a ranking function that is not based on likelihood.

The result in this subsection allows us to view query-likelihood model from the query-centric perspective, where the ranking function corresponds to the one used in advanced language models, but the query model is only a simple MLE of the query.

2.2 Cross entropy in Text Classification

Text classification (TC) is an area of research concerned with the task of automatically labeling documents into predefined classes or topics. Each class is provided with a set of labeled documents called the *training set*, based on which the system learns certain classification rules. These classification rules are then applied to label hitherto unseen documents, called the *test set*, into their respective classes.

One main difference between TC and IR is that TC requires explicit labeling of documents and not ranking as expected in IR. One also needs to keep in mind the fact that while the objective in TC is to find the best class/topic for each document, in IR, the objective is to find the best document(s) for each query/topic. An outcome of this dissimilarity is that the equivalence relationship $\overset{rank}{=}$ between ranking functions in IR is not applicable in TC. Instead we will talk about *class-equivalence* relationship ($\overset{class}{=}$) of classifiers. As defined in section 1.3, class-equivalence of two classifiers implies that both of them classify any given document into the same class.

In the domain of TC, the counterpart to the language modeling approach for IR is the naïve Bayes model [31]. The naïve Bayes model, like the language model, employs the multinomial distribution $\boldsymbol{\theta}^T$ to model each topic T. Given a new test document D, it is classified into one of the K topics that has the highest posterior probability of the topic $P(T|D)$. The posterior probability is estimated as shown below:

$$
\begin{aligned}
P(T|D) = \frac{P(D|T)P(T)}{P(D)} &\overset{class}{=} P(D|T)P(T) && (2.7)\\
&= P(T)\prod_{j=1}^{V}(\theta_j^T)^{f_j^D} && (2.8)\\
&\overset{class}{=} \prod_{j=1}^{V}(\theta_j^T)^{f_j^D} && (2.9)\\
&\overset{class}{=} \sum_{j=1}^{V} f_j^D \log \theta_j^T && (2.10)\\
&\overset{class}{=} \sum_{j=1}^{V} \frac{f_j^D}{|D|} \log \theta_j^T = -H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^T) && (2.11)
\end{aligned}
$$

where (2.7) is a direct application of the Bayes rule. In (2.9), we assume that all the topics(classes) have the same prior $P(T)$ and hence the prior term has no bearing on classification. Step (2.10) follows from the fact that the log transformation is a monotonic function of its input and hence does not affect the classification decision. In step (2.11), we divide the expression by a constant factor $|D|$ and still maintain class-equivalence, since $|D|$ is a function of the document and not the class and hence will not influence the choice of class for a *given* document (note that in the context of IR, such transformation of the ranking function will not preserve the rank-equivalence relation). Thus, we have shown that the naïve Bayes classification rule is class-equivalent to the cross entropy $H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^T)$ between the document's MLE distribution and the topic model [2].

2.3 Cross entropy in IR as Text Classification

Text classification is closely related to ad hoc retrieval in many respects. In fact, ad hoc retrieval can be looked at as a special case of a text classification task, a perspective taken by many previous IR researchers as described in section 1.4[3]. Each query in the ad hoc

[2]Under the assumption that the prior probability is uniformly distributed among the classes.

[3]Note that one can also view text classifi cation as a special case of ad hoc retrieval. In this work, we will consider only the former view.

retrieval task corresponds to a topic or class in TC. While in TC, each topic is provided with a set of labeled documents, in IR, one can think of the query as the lone, concise training example for the topic. For each query, the ad hoc retrieval task can now be looked at as that of classifying documents in the entire collection into two abstract classes R and N representing 'relevant' and 'non-relevant' classes respectively [45]. The test set would then correspond to the entire collection.

In this view, a natural ranking function would be the posterior probability of relevance for a given document $P(R|D)$ as per the *Probability Ranking Principle*[4] . In a generative approach, assuming we have two language models (multinomial distributions) $\boldsymbol{\theta}^R$ and $\boldsymbol{\theta}^N$ for the Relevant and Non-Relevant classes respectively, the posterior probability of Relevance is given by:

$$
\begin{aligned}
P(R|D, \boldsymbol{\theta}^R, \boldsymbol{\theta}^N) &= \frac{P(D|\boldsymbol{\theta}^R)\pi_R}{P(D|\boldsymbol{\theta}^R)\pi_R + P(D|\boldsymbol{\theta}^N)\pi_N} && (2.12)\\
&= \frac{\frac{P(D|\boldsymbol{\theta}^R)\pi_R}{P(D|\boldsymbol{\theta}^N)\pi_N}}{\frac{P(D|\boldsymbol{\theta}^R)\pi_R}{P(D|\boldsymbol{\theta}^N)\pi_N} + 1} && (2.13)\\
&\overset{rank}{=} \frac{P(D|\boldsymbol{\theta}^R)\pi_R}{P(D|\boldsymbol{\theta}^N)\pi_N} \overset{rank}{=} \frac{P(D|\boldsymbol{\theta}^R)}{P(D|\boldsymbol{\theta}^N)} && (2.14)\\
&= \frac{\prod_{j=1}^{V} \left(\theta_j^R\right)^{f_j^D}}{\prod_{j=1}^{V} \left(\theta_j^N\right)^{f_j^D}} && (2.15)\\
&\overset{rank}{=} \sum_{j=1}^{V} f_j^D \log \frac{\theta_j^R}{\theta_j^N} && (2.16)\\
&= |D| \sum_{j=1}^{V} \frac{f_j^D}{|D|} \log \frac{\theta_j^R}{\theta_j^N} && (2.17)\\
&= |D| \left(H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^N) - H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^R) \right) && (2.18)
\end{aligned}
$$

[4]"If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user,... the overall effectiveness of the system to its user will be the best that is obtainable..."[46]

where π_R and π_N are the prior probabilities of the Relevance and Non-relevance classes respectively. In the above derivation, step (2.12) follows from the application of Bayes rule, while step (2.13) is a simple algebraic manipulation of the RHS of step (2.12). Step (2.14) follows from the fact that $\frac{x}{x+1}$ is a monotonic function of x and is hence rank-equivalent to x. We also use the fact that the prior ratio π_R/π_N does not influence the ranking of documents. In step (2.15), we substitute the multinomial parameterization for the class-conditionals $P(D|\boldsymbol{\theta}^R)$ and $P(D|\boldsymbol{\theta}^N)$ and in step (2.16), we use a log transformation which maintains rank-equivalence. The next two steps are simple algebraic manipulations.

Thus the posterior probability of relevance is rank-equivalent to the difference between cross entropies $H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^N)$ and $H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^R)$, save a constant factor $|D|$ (which cannot be ignored in a rank-equivalence relation because it is document dependent). This ranking function would choose documents whose MLE language models $\hat{\boldsymbol{\theta}}^D$ are as 'close' to the relevant distribution $\boldsymbol{\theta}^R$ and as 'far' from the non-relevant distribution $\boldsymbol{\theta}^N$ as possible, where the distance is measured in terms of the respective cross entropy functions.

2.4 Anomalous behavior of Cross Entropy

To correlate the discussion in sections 2.1, 2.2 and 2.3, we will consider $\boldsymbol{\theta}^T$ in TC and $\boldsymbol{\theta}^R$ in IR analogous since they both represent topic models w.r.t. which the documents are classified into topics.

Examining the naive Bayes classifier in text classification in (2.11) and the classification based ranking function in ad hoc retrieval in (2.18), it is apparent that they both are based on the cross entropy of the document model w.r.t. the topic model as in $H(\boldsymbol{\theta}^D, \boldsymbol{\theta}^T)$ and $H(\boldsymbol{\theta}^D, \boldsymbol{\theta}^R)$ respectively. However the ranking function employed by the language models shown in (2.3) is the cross entropy of the topic model w.r.t. the document model $H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^D)$. Table 2.1 presents a comparative summary of the naïve Bayes model in classification and language modeling in information retrieval.

	naïve Bayes	Language modeling
Document representation	$\mathbf{f}^D$ or $\hat{\boldsymbol{\theta}}^D$	$\boldsymbol{\theta}^D$
Topic Representation	$\boldsymbol{\theta}^T$	$\boldsymbol{\theta}^R$
Inference	$P(T\|D) \stackrel{class}{=} -H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^T)$	$-H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^D)$

Table 2.1. Comparing Language modeling with naïve Bayes classifier

Note that cross entropy is an asymmetric function and hence $H(\boldsymbol{\theta}^D, \boldsymbol{\theta}^T) \neq H(\boldsymbol{\theta}^T, \boldsymbol{\theta}^D)$. For clarity, we will call the former version Document-Cross-Entropy (DCE) since the document model $\boldsymbol{\theta}^D$ is on the left-hand side. The latter version will be referred to as Topic-Cross-Entropy (TCE) since the topic model $\boldsymbol{\theta}^T$ is on its left hand side.

If one were to choose between these two versions, DCE seems a more natural choice by virtue of its correspondence to the posterior probability of topic $P(T|D)$ or $P(R|D)$ when the topic is modeled as a multinomial distribution. However, empirical evidence suggests that TCE, used in the language modeling approach, is a superior performer [23]. Related work by Teevan [51] also confirms that the performance of DCE is inferior to the more traditional vector space models in IR.

It is important to note that the cross-entropy ranking used in the language modeling approach is only an algorithmic choice and does not follow from the model. As such, there is no direct theoretical justification available for its choice as the ranking function. Even if we assume the applicability of cross-entropy function for ad hoc retrieval, it is still not clear why one should employ TCE instead of its asymmetric counterpart DCE as a ranking function.

Clearly, there is a need for some investigation and analysis to explain the choice of cross-entropy as a ranking function. In particular, we need better understanding of why TCE is empirically a superior ranking function. Additionally, it would be ideal if the ranking function employed followed directly from the underlying model.

The main motivation of the present work is to develop understanding of the differences between TCE, which is a successful ranking function in IR and DCE, which is used as a classification function in TC. One of our objectives is to provide a theoretical reasoning for the empirical success of TCE as a ranking function in IR. We will also investigate if the empirical success of TCE in IR can also be translated to TC. For this purpose, we aim to build a unified model for IR and TC that corresponds to the TCE function and test its effectiveness w.r.t. the naïve Bayes classifier on TC and language models in IR. In a larger perspective, we believe our work helps bridge the gap between IR and TC models and brings the two research communities closer together.

Our investigation unearths a new distribution called the smoothed Dirichlet distribution. We show that approximate inference w.r.t. this distribution is rank-equivalent to the TCE ranking. We also show empirically that this distribution models term occurrence counts better than the standard multinomial distribution, providing a justification for why TCE ranking should be a better performer. We also demonstrate that a simple classification model based on this distribution performs at least as well as the existing techniques on both text classification as well as ad hoc retrieval.

CHAPTER 3

THE SMOOTHED DIRICHLET DISTRIBUTION

In this chapter, we present a new distribution called the Smoothed Dirichlet distribution whose approximation corresponds to the T-cross-entropy ranking function. We also empirically demonstrate that this distribution models text much better than the multinomial, providing a justification for the usage of TCE function for document ranking.

3.1 Motivation: Dirichlet distribution and its rank-equivalence to TCE

In section 2.3, we argued that ad hoc retrieval can be viewed as a classification problem. In the rest of the discussion, we will present our arguments from this point of view. This not only allows us to treat both ad hoc retrieval and text classification in a unified manner, but also permits us the luxury of utilizing past work from both these domains in our discussion.

It has been discovered by researchers in the recent past that the multinomial distribution fails to capture the term occurrence characteristics in documents [52, 41]. In particular, it has been found that the multinomial distribution fails to predict the heavy tail behavior or burstiness of terms, a phenomenon where words tend to occur in bursts, at a rate much higher than that predicted by the multinomial [41]. We illustrate this phenomenon in our experiments in section 3.4.

Recall that the D-cross-entropy function is class-equivalent (and rank-equivalent save a constant factor) to the log-likelihood of documents w.r.t. a topic modeled by the multinomial distribution. Thus, the poor performance of D-cross entropy as a ranking function could be attributed to the fact that the distribution underlying this function, the multinomial, is a poor model for text. Adding strength to this argument is the observation that any ad

hoc transformations to the multinomial that fit the empirical term occurrence distribution better also lead to superior performance in text classification [41]. Inspired by this work, Madsen *et al* [30] proposed the Dirichlet Compound Multinomial (DCM) distribution to model text, in which the Dirichlet distribution, a distribution over the multinomial simplex, shown below in 3.1, is used as an empirical prior to the multinomial distribution.

$$P(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{1}{Z^{Dir}} \prod_{j=1}^{V} \left(\theta_j^D\right)^{\alpha_j - 1} = \frac{\Gamma(\sum_{j=1}^{V} \alpha_j)}{\prod_{j=1}^{V} \Gamma(\alpha_j^)} \prod_{j=1}^{V} \left(\theta_j^D\right)^{\alpha_j - 1} \tag{3.1}$$

The parametric form of the DCM distribution is as shown below.

$$P(\mathbf{f}^D|\boldsymbol{\alpha}) = \int_{\boldsymbol{\theta}} P(\mathbf{f}^D|\boldsymbol{\theta}) P(\boldsymbol{\theta}|\boldsymbol{\alpha}) d\boldsymbol{\theta} = \frac{\Gamma(\sum_{j=1}^{V} \alpha_j)}{\prod_{j=1}^{V} \Gamma(\alpha_j)} \frac{\prod_{j=1}^{V} \Gamma(\alpha_j + f_j^D)}{\Gamma\left(\sum_{j=1}^{V} (\alpha_j + f_j^D)\right)} \tag{3.2}$$

where Γ is the Gamma function. The generative process in this distribution involves sampling a multinomial $\boldsymbol{\theta}$ from the Dirichlet distribution first and then repeated sampling of words in an I.I.D. fashion from the multinomial to obtain the document as shown in figure 3.1. To compute the probability of the document given the Dirichlet parameters, we simply marginalize the multinomial parameters to obtain a closed form solution as shown in (3.2).

Madsen *et al* demonstrated empirically that DCM is a better fit to term occurrence distribution than the multinomial. They further showed that this distribution also translates to better performance on the text classification task than the multinomial.

These observations lead us to believe that the success of the T-cross entropy ranking function in IR may imply an underlying distribution that is a better model for text [1]. Upon inspection, an obvious candidate that roughly corresponds to the T-cross entropy is the Dirichlet distribution as shown in (3.1). The argument $\boldsymbol{\theta}$ of the Dirichlet distribution can be considered as the document language model. The Dirichlet parameters model the query's

[1]Note that it is not a strictly necessary condition for good classifiers to correspond to good models of text. For example, Domingos and Pazzani [10] showed that the naïve Bayes classifier performs well on the classification task under certain conditions although it completely fails to model feature dependence

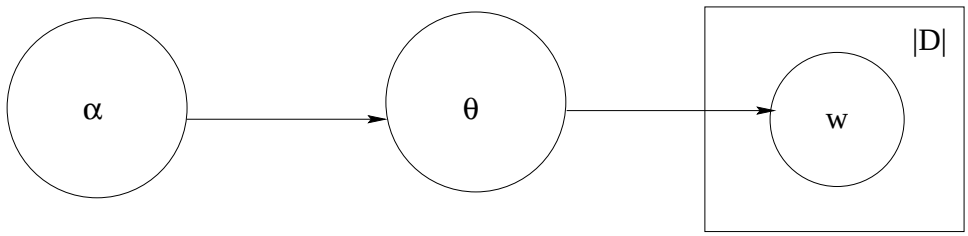

Figure 3.1. Graphical representation of the DCM distribution: A multinomial $\boldsymbol{\theta}$ is sampled from the Dirichlet distribution from which words are repeatedly sampled in an IID fashion to generate the document

Figure 3.2. Graphical representation of the Dirichlet and SD distributions: A multinomial $\boldsymbol{\theta}$ representing the document is sampled directly from the Dirichlet (SD) distribution.

topic (or the class's topic in the context of TC). Note that unlike the multinomial distribution that generates one word at each sampling, the Dirichlet distribution generates a whole multinomial distribution each time as shown in figure 3.2. This is convenient because, in the language modeling approach, documents are represented as smoothed multinomial distributions $\boldsymbol{\theta}^D$ and it is natural to generate them from a Dirichlet distribution.

Recall that in our perspective of IR as a binary text classification problem, documents are ranked by the posterior probability of relevance $P(R|D)$ as described in section 2.3. When the Dirichlet distribution is used to model the relevant and non-relevant classes using parameter vectors $\boldsymbol{\alpha}^R$ and $\boldsymbol{\alpha}^N$ respectively, the posterior probability corresponds to the difference in TCE's between the classes and the document model as shown below:

$$
\begin{aligned}
P(R|\boldsymbol{\theta}^D, \boldsymbol{\alpha}^R, \boldsymbol{\alpha}^N) &\overset{rank}{=} \frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)} && (3.3)\\
&= \frac{\frac{\Gamma(\sum_{j=1}^V \alpha_j^R)}{\prod_{j=1}^V \Gamma(\alpha_j^R)} \prod_{j=1}^V \left(\theta_j^D\right)^{\alpha_j^R - 1}}{\frac{\Gamma(\sum_{j=1}^V \alpha_j^N)}{\prod_{j=1}^V \Gamma(\alpha_j^N)} \prod_{j=1}^V \left(\theta_j^D\right)^{\alpha_j^N - 1}} && (3.4)\\
&\overset{rank}{=} \prod_{j=1}^V \left(\theta_j^D\right)^{\alpha_j^R - \alpha_j^N} && (3.5)\\
&\overset{rank}{=} \sum_{j=1}^V \left(\alpha_j^R - \alpha_j^N\right) \log \theta_j^D && (3.6)\\
&= H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D) && (3.7)
\end{aligned}
$$

where step (3.3) follows from steps (2.12) through (2.14) and step (3.4) is obtained from (3.3) using (3.1). Step (3.5) is obtained by ignoring all terms that are document independent and the remaining steps are straightforward.

Recall that cross-entropy is a distance metric between two probability distributions. The parameters of the Dirichlet distribution $\boldsymbol{\alpha}^R$ or $\boldsymbol{\alpha}^N$ do not constitute a probability distribution. In the special case when the Dirichlet scale, defined by $S = \sum_j \alpha = 1$, the Dirichlet parameter vector can be considered a probability distribution over the vocabulary since the Dirichlet parameters are always non-negative. In a general case, when $S \neq 1$, one can consider $\boldsymbol{\alpha}/S$ as a distribution. In such a case, the expression $-\sum_j \alpha_j \log \theta_j$ differs from the true cross entropy $H(\boldsymbol{\alpha}/S, \boldsymbol{\theta})$ by a constant factor S. In a slight abuse of the definition of cross entropy, we will ignore this factor, and refer to the expression as cross entropy even when $S \neq 1$.

Madsen *et al* argued in [30] that the Dirichlet distribution is desirable for text, since it is qualitatively similar to the Zipf's law for word occurrence distribution [64], which states that the probability of occurrence of a term in a document follows a power law, $P(w) \propto r(w)^{-a}$ where $r(w)$ is the rank of the word in the descending order of frequency of occurrence and a is a parameter. Note that the Dirichlet distribution shown in (3.1) has a similar form $data^{parameter}$. However, they stopped short of using it as a direct gener-

ative distribution for text citing document representation as a potential problem in using the distribution. Instead, they proposed using DCM shown in (3.2), which uses the same document representation as the multinomial and demonstrated better results on TC than the multinomial based naïve Bayes classifier.

The Dirichlet distribution has never been used as a generating distribution for text prior to this work. In the language modeling framework, Dirichlet has been used as a prior to the multinomial. A *Maximum a Posteriori* (MAP) estimate of the multinomial using the Dirichlet as the prior results in what is known as the Dirichlet smoothing estimates for the multinomial parameters [60]. In other related work, Zaragoza *et al* [58] used the same Dirichlet prior for the multinomial in computing the query-likelihood function in ad hoc retrieval. Instead of using a MAP estimate, they computed the full integral and showed that the new ranking function is more stable across various collections than simple Dirichlet smoothing. Even in more complex topic models such as the Latent Dirichlet Allocation [5], the multinomial is used as the generative distribution for documents while the Dirichlet distribution is used as a prior to the multinomial distributions or as a prior that generates mixing proportions of various topics.

3.2 Drawbacks of the Dirichlet distribution

We have shown that a simple binary classifier using the Dirichlet distribution as a generative distribution for document language models is rank-equivalent to the difference in TCEs of the two classes w.r.t. the document language models. In the context of text classification, however, Dirichlet distribution is only proportional but not class-equivalent to cross-entropy as shown below.

$$
\begin{aligned}
P(T|\boldsymbol{\theta}^D, \boldsymbol{\alpha}^T) &\overset{class}{=} P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^T)P(T) && (3.8)\\
&\overset{class}{=} \log P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^T) \text{ (assuming equal priors for all the classes)} && (3.9)\\
&= \log\left(\Gamma(\sum_{j=1}^{V}\alpha_j^T)\right) - \sum_{j=1}^{V}\log\Gamma(\alpha_j^T) + \sum_{j=1}^{V}\left(\alpha_j^T - 1\right)\log\theta_j^D && (3.10)\\
&\overset{class}{=} -\sum_{j=1}^{V}\log\Gamma(\alpha_j^T) - H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D) && (3.11)
\end{aligned}
$$

where in step (3.11), we ignored the term $-\sum_j \log\theta_j^D$, because it does not influence the choice of the class. We also ignored the term $\log\left(\Gamma(\sum_{j=1}^{V}\alpha_j^T)\right)$ with the assumption that $S = \sum_j \alpha_j^T$ is the same for all classes and hence does not influence the classification decision. The other term $\sum_{j=1}^{V}\log\Gamma(\alpha_j^T)$ cannot be ignored because it is class-dependent. Thus, in the classification context, the classification function corresponding to the Dirichlet distribution is proportional to the cross entropy but is not class-equivalent to it.

In addition, estimation of the parameters of the Dirichlet distribution is not straightforward. Unlike the multinomial distribution, the Maximum Likelihood Estimate (MLE) of the Dirichlet distribution has no simple closed form solution. To compute the MLE parameters of the Dirichlet distribution, one needs to use iterative gradient descent techniques as described in [33]. Such computationally expensive learning techniques render the distribution unattractive for many IR tasks where response time to the user is of critical importance. Hence one needs a distribution that is a better model of text than the multinomial, but also importantly, one that is as easy to estimate.

There is another fundamental problem arising out of the smoothed language model representation of documents (see (2.1)): the Dirichlet distribution assigns probability mass to the entire multinomial simplex $\Delta = \{\boldsymbol{\theta} \mid \forall_j \theta_j > 0; \sum_j \theta_j = 1\}$, while smoothed language models occupy only a subset Δ^s of the simplex. To illustrate this phenomenon, we generated 1000 documents of varying lengths uniformly at random using a vocabulary of size 3, estimated their MLEs $\hat{\boldsymbol{\theta}}^D$, smoothed them with $\hat{\boldsymbol{\theta}}^{GE}$ estimated from the entire document set, and plotted the smoothed language models $\boldsymbol{\theta}^D = \lambda\hat{\boldsymbol{\theta}}^D + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE}$ in

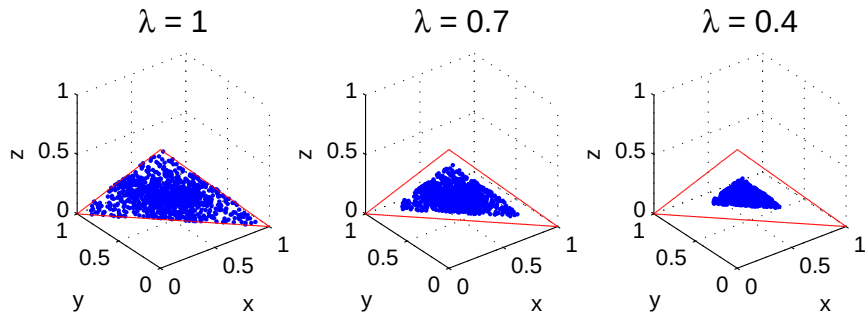

Figure 3.3. Domain of smoothed proportions Δ^s for various degrees of smoothing: dots are smoothed language models $\boldsymbol{\theta}^D$ and the triangular boundary is the 3-D simplex.

figure 3.3. The leftmost plot represents the MLE estimates $\hat{\boldsymbol{\theta}}^D$ corresponding to $\lambda = 1$. As shown in the plot, the documents cover the whole simplex Δ when not smoothed. But as we increase the degree of smoothing, the new domain Δ^s spanned by the smoothed documents gets compressed towards the centroid. Hence, the Dirichlet distribution that considers the whole simplex Δ as its domain is clearly incorrect given our smoothed language model representation for documents.

One way to overcome this problem is to use the MLE representation for documents $\hat{\boldsymbol{\theta}}^D$ instead of smoothed representation $\boldsymbol{\theta}^D$. We have seen in figure 3.3 that the documents span the entire simplex when their MLE representation is used. Hence, this representation would be consistent with the Dirichlet distribution. However, since most documents usually contain only a small fraction of the entire vocabulary, the MLE representation of documents $\hat{\boldsymbol{\theta}}^D = \frac{\mathbf{f}^D}{|D|}$ would result in many zero components corresponding to the words that do not occur in them. Inspecting the parametric representation of the Dirichlet distribution in (3.1), it is evident that this would result in assignment of zero probabilities to almost all docu-

ments. Hence smoothing the MLE estimates to ensure non-zero components is necessary if one were to use the Dirichlet distribution to model documents.

3.3 Our solution: The Smoothed Dirichlet distribution

In this section, we propose a novel variation to the Dirichlet distribution called the *Smoothed Dirichlet* (SD) distribution that overcomes some of its flaws. The SD distribution has the same parametric form as the Dirichlet distribution but corrects the probability mass distribution problem of the Dirichlet distribution stated above by defining a new corrected normalizer for smoothed language model representation of documents. We then construct an approximation to the SD distribution that allows us to compute the MLE parameters using a closed form solution, much like the multinomial.

3.3.1 SD normalizer

We start our analysis by examining the Dirichlet normalizer Z^{Dir} in (3.1), which is defined as follows:

$$Z^{Dir}(\bar{\alpha}) = \int_{\boldsymbol{\theta} \in \Delta} \prod_j \theta_j^{\alpha_j - 1} d\boldsymbol{\theta} = \frac{\prod_j \Gamma(\alpha_j)}{\Gamma(\sum_j \alpha_j)} \tag{3.12}$$

When we use a smoothed representation for documents, the integral in (3.12) should span only over the compressed domain Δ^s that contains all the smoothed language models as given by the following expression:

$$\Delta^s(\lambda, \hat{\boldsymbol{\theta}}^{GE}) = \{\boldsymbol{\theta}\} = \{\lambda \hat{\boldsymbol{\theta}} + (1 - \lambda)\hat{\boldsymbol{\theta}}^{GE} \mid \hat{\boldsymbol{\theta}} \in \Delta\} \tag{3.13}$$

Thus the exact normalizer for smoothed language models, Z_{SD} should be

$$Z^{SD} = \int_{\boldsymbol{\theta} \in \Delta^s} \prod_{j=1}^{V} (\theta_j)^{\alpha_j - 1} d\boldsymbol{\theta} \tag{3.14}$$

Exploiting the mapping from Δ^s to Δ in (3.13) by substituting it into (3.14), we get

$$\begin{aligned} Z^{SD} &= \int_{\hat{\boldsymbol{\theta}} \in \Delta} \prod_{j=1}^{V} \left\{ \lambda \hat{\theta}_j + (1-\lambda) \hat{\theta}_j^{GE} \right\}^{\alpha_j - 1} d(\lambda \hat{\boldsymbol{\theta}} + (1-\lambda) \hat{\boldsymbol{\theta}}^{GE}) \\ &= \lambda \int_{\hat{\boldsymbol{\theta}} \in \Delta} \prod_{j=1}^{V} \left\{ \lambda \hat{\theta}_j + (1-\lambda) \hat{\theta}_j^{GE} \right\}^{\alpha_j - 1} d\hat{\boldsymbol{\theta}} \end{aligned} \tag{3.15}$$

For fixed values of λ and $\hat{\boldsymbol{\theta}}^{GE}$, Z^{SD} can be transformed to an incomplete integral of the multi-variate Beta function. Thus the exact form of Smoothed Dirichlet distribution can now be defined as follows.

$$P(\boldsymbol{\theta}^D | \boldsymbol{\alpha}) = \frac{1}{Z^{SD}} \prod_{j=1}^{V} \left(\theta_j^D\right)^{\alpha_j - 1} = \frac{\prod_{j=1}^{V} \left(\theta_j^D\right)^{\alpha_j - 1}}{\lambda \int_{\hat{\boldsymbol{\theta}} \in \Delta} \prod_{j=1}^{V} \left\{ \lambda \hat{\theta}_j + (1-\lambda) \hat{\theta}_j^{GE} \right\}^{\alpha_j - 1} d\hat{\boldsymbol{\theta}}} \tag{3.16}$$

where we have explicitly included the superscript ^{D} to indicate that the distribution is applicable only for smoothed language model representation of documents. We may omit this in the future, where it is clear from the context.

From the perspective of manifold learning, the Smoothed Dirichlet distribution is the exact distribution corresponding to the data manifold occupied by the smoothed language model representation of the documents.

3.3.2 Generative Process

The generative process for the Smoothed Dirichlet distribution is same as that of the Dirichlet distribution except that the domain of the distribution is restricted to include only smoothed language models. Recall that the document representation under this distribution is significantly different from that of the Multinomial or the DCM distributions. As we have noted earlier, in both multinomial and the DCM, words are sampled one at a time in an I.I.D. fashion to generate a document. In the DCM, the multinomial distribution itself is sampled from a Dirichlet prior before words are sampled from the former. In SD

distribution, much like the Dirichlet, each sampling generates a smoothed language model $\boldsymbol{\theta}^D$ that represents documents directly as shown in figure 3.2. Thus SD and Dirichlet are *probability density functions* whereas the multinomial and DCM are discrete *probability mass functions*.

In this work, we view documents only as smoothed multinomials $\boldsymbol{\theta}^D$. Given a counts representation of a document $\mathbf{f}^D$, we project it to the smoothed simplex Δ^s by computing its smoothed language model given by $\boldsymbol{\theta}^D = \lambda \frac{\mathbf{f}^D}{|D|} + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE}$ and estimating its probability under the SD distribution.

$$P_{SD}(\mathbf{f}^D|\boldsymbol{\alpha}) \overset{\approx}{\equiv} P_{SD}(\boldsymbol{\theta}^D|\boldsymbol{\alpha}) \tag{3.17}$$

Note that we make the above assumption only for mathematical convenience. In fact the probability mass of counts vector $\mathbf{f}^D$ w.r.t. the SD distribution does depend on the document length as we demonstrate using an upper bound analysis in the appendix.

The correspondence relation in (3.17) allows us to define an equivalence from smoothed language models $\boldsymbol{\theta}^D$ to document counts vector $\mathbf{f}^D$ as follows.

$$\hat{\boldsymbol{\theta}}^D \overset{\approx}{\equiv} \frac{(\boldsymbol{\theta}^D - (1-\lambda)\hat{\boldsymbol{\theta}}^{GE})}{\lambda} \tag{3.18}$$

$$\mathbf{f}^D \overset{\approx}{\equiv} \text{int}(|D| \times \hat{\boldsymbol{\theta}}^D) \tag{3.19}$$

where (3.18) is an inverse relation of (2.1) and (3.19) is the inverse of (1.7). The approximate equivalence relation in (3.18) gives us an insight into the allowable values for the smoothed language models $\boldsymbol{\theta}^D$. Since $\hat{\boldsymbol{\theta}}^D$ lies in the multinomial simplex, its components cannot be negative. As indicated by (3.18), this means that allowable values $\boldsymbol{\theta}^D$ have to satisfy the following constraint:

$$\boldsymbol{\theta}^D - (1-\lambda)\hat{\boldsymbol{\theta}}^{GE} > \mathbf{0} \text{ and } \frac{(\boldsymbol{\theta}^D - (1-\lambda)\hat{\boldsymbol{\theta}}^{GE})}{\lambda} \in \Delta \tag{3.20}$$

One can easily relate the constraint in (3.20) to figure 3.3. When $\lambda = 1$, $\boldsymbol{\theta}^D$ can take any value in the simplex as indicated by (3.20). As the value of λ decreases, the number of allowable values that satisfy the constraint decreases, as indicated by the shrinking domain of $\boldsymbol{\theta}^D$ in figure 3.3.

3.3.3 Approximations to SD

We have succeeded in defining an appropriate distribution for smoothed language models representation of documents, but the new distribution faces the same problem that plagues the Dirichlet distribution too, namely non-existence of a simple closed form solution for maximum likelihood estimation of the parameters.

In this subsection, we will focus on developing a theoretically motivated approximation to the SD distribution. Our approach is mainly centered on finding an analytically tractable approximator Z_a^{SD} for the SD normalizer Z^{SD} of (3.15).

Figure 3.4(a) compares Z^{SD} with the Dirichlet normalizer Z of (3.12) for a simple case where the vocabulary size V is 2, *i.e.*, $\bar{\alpha} = \{\alpha_1, \alpha_2\}$. We imposed the condition that $\alpha_1 + \alpha_2 = 1$ and used $\lambda = 0.2$ and $\left\{\hat{\theta}_1^{GE}, \hat{\theta}_2^{GE}\right\} = \{0.5, 0.5\}$. The plot shows the value of Z^{SD} for various values of α_1 computed using the incomplete two-variate Beta function implementation of *Matlab*. Notice that Z^{SD} tends to finite values at the boundaries while Z^{Dir}, the Dirichlet normalizer is unbounded. We would like to define Z_a^{SD}, an approximation to Z^{SD} such that it not only shows similar behavior to Z^{SD}, but is also analytically tractable. Taking cue from the functional form of the Dirichlet normalizer Z^{Dir} in (3.12), we define Z_a^{SD} as:

$$Z_a^{SD}(\bar{\alpha}) \;=\; \prod_j \Gamma_a(\alpha_j) \;/\; (\Gamma_a(\sum_j \alpha_j)) \tag{3.21}$$

where $\Gamma_a(\alpha)$ is an approximation to $\Gamma(\alpha)$. Now all that remains is to choose a functional form for $\Gamma_a(\alpha)$ such that Z_a^{SD} closely approximates the SD normalizer Z^{SD} of (3.15).

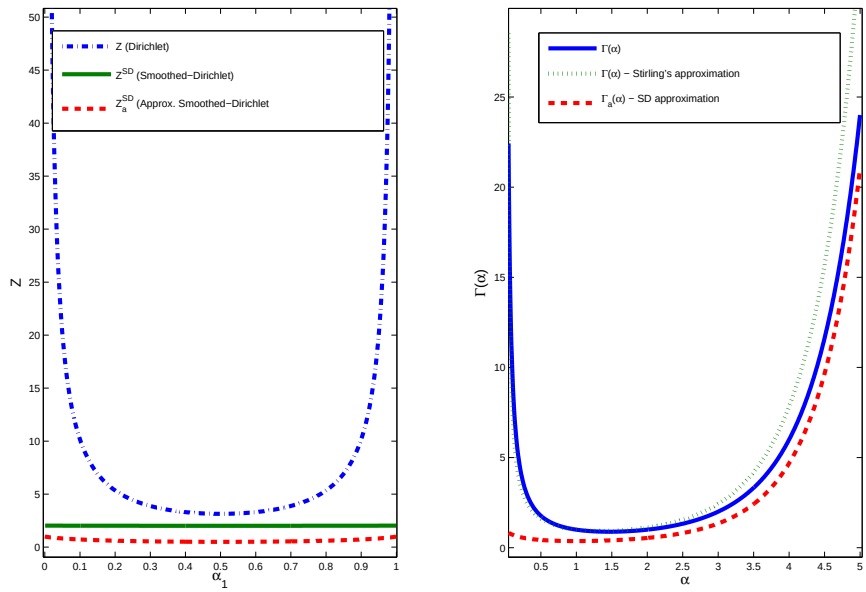

Figure 3.4. (a) Comparison of the normalizers (b) Gamma function and its approximators

We turn to the Stirling's approximation of the Gamma function [1], shown in (3.22) for guidance.

$$\Gamma(\alpha) \approx e^{-\alpha}\alpha^{\alpha-1/2}\sqrt{2\pi}(1+\frac{1}{12\alpha}+O(\frac{1}{\alpha^2})) \tag{3.22}$$

Figure 3.4(b) plots the Γ function and its Stirling approximation which shows that $\Gamma(\alpha) \to \infty$ in the limit as $\alpha \to 0$. Inspecting (3.12), it is apparent that this behavior of the Γ function is responsible for the unboundedness of Dirichlet normalizer at small values of α. Since our exact computation in low dimensions shows that the Smoothed Dirichlet normalizer Z^{SD} is bounded as $\alpha \to 0$, we need a bounded approximator of Γ. An easy way to define this approximation is to ignore the terms in Stirling's approximation that make it unbounded and redefine it as:

$$\Gamma_a(\alpha) \stackrel{\text{def}}{=} e^{-\alpha}\alpha^{\alpha} \tag{3.23}$$

While there are several ways to define a bounded approximation, we chose an approximation that is not only mathematically simple, but also yields a closed form solution to maximum likelihood estimation as we will show later. The approximate function Γ_a is compared to the exact function Γ again in figure 3.4(b). Note that the approximate function yields bounded values at low values of α but closely mimics the exact function at larger values. Combining (3.21) and (3.23), we have:

$$\begin{aligned} Z_a^{SD}(\bar{\alpha}) &= \frac{\prod_j e^{-\alpha_j} \alpha_j^{\alpha_j}}{e^{-\sum_j \alpha_j} (\sum_j \alpha_j)^{\sum_j \alpha_j}} \\ &= \frac{e^{-S} \prod_j \alpha_j^{\alpha_j}}{e^{-S} S^S} \\ &= \frac{\prod_j \alpha_j^{\alpha_j}}{S^S} \end{aligned} \tag{3.24}$$

where $S = \sum_j \alpha_j$. The approximation in (3.24) is independent of λ and $\hat{\boldsymbol{\theta}}^{GE}$ which is clearly an oversimplification of the exact SD normalizer Z^{SD} in (3.15). However our plot of the approximate SD normalizer Z_a^{SD} in figure 3.4(a) shows that it behaves very similar to Z^{SD}. Our new approximate Smoothed Dirichlet distribution can now be defined as:

$$P_a(\mathbf{f}^D | \hat{\boldsymbol{\theta}}^{GE}, \lambda, L, \boldsymbol{\alpha}) = P_a(\boldsymbol{\theta}^D | \boldsymbol{\alpha}) = \frac{S^S}{\prod_j \alpha_j^{\alpha_j}} \prod_j \left(\theta_j^D\right)^{\alpha_j - 1} \tag{3.25}$$

Henceforth, we will refer to the approximate SD distribution as the SD distribution for convenience. The subscript in P_a helps remind us that it is an approximate probability density function.

3.3.4 Maximum Likelihood Estimation of Approximate SD parameters

Given a set of N documents $\{\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^N\}$ on a topic T where each $\boldsymbol{\theta}^i$ is a smoothed language model representation of the ith document, the maximum likelihood estimates

(MLE) of the SD parameter vector $\hat{\boldsymbol{\alpha}}^T$ are given by the values that maximize the Smoothed-Dirichlet likelihood-function shown in (3.25).

$$\hat{\boldsymbol{\alpha}}^T = \arg\max_{\boldsymbol{\alpha}} \log\{\prod_{i=1}^{N} P_a(\boldsymbol{\theta}^i|\boldsymbol{\alpha})\} \tag{3.26}$$

$$= \arg\max_{\boldsymbol{\alpha}} \log\{\prod_{i=1}^{N}\prod_{j=1}^{V} \frac{(\sum_j \alpha_j)^{(\sum_j \alpha_j)}}{\prod_j \alpha_j^{\alpha_j}} (\theta_j^i)^{\alpha_j - 1}\} \tag{3.27}$$

Differentiating the log-likelihood function for N documents with respect to each α_j with an additional Lagrange multiplier term with the constraint that $\sum \alpha_j = S$ and equating to zero, treating S as a constant, gives us the following closed-form solution for $\boldsymbol{\alpha}$

$$\hat{\boldsymbol{\alpha}}^T = \frac{1}{Z}\{\prod_{i=1}^{N} \boldsymbol{\theta}^i\}^{\frac{1}{N}} \tag{3.28}$$

Here, Z is a normalizer that ensures $\sum_j \hat{\alpha}_j^T = S$. We consider S a free parameter that scales individual $\hat{\alpha}_j^T$'s proportionately. It is easy to verify that the second derivative of the log-likelihood function is always less than zero guaranteeing convexity of the log-likelihood function and thereby the global optimality of the MLE solution. Thus, the approximate SD distribution provides a closed form solution for training where our estimates of $\hat{\boldsymbol{\alpha}}^T$ are simply normalized geometric averages of the smoothed proportions of words in training documents.

As shown in (1.7), the MLEs of the parameters of the multinomial distribution, on the other hand, correspond to normalized sums of raw counts of a term in all documents. Thus, the SD model gives higher weight to terms that occur at high *relative* frequency in a large number of documents while the multinomial ignores the average distribution per document and assigns higher weight to terms that are highly frequent in the whole data set. In other words, one can think of the SD model as computing a micro-average while the multinomial computes a macro-average in parameter estimation.

3.3.4.1 Computationally efficient estimation

The MLE of the multinomial distribution is a normalized arithmetic average of the counts of words in documents, hence the summation needs to be performed only over words that occur in them. On the other hand, MLEs of the SD distribution correspond to the geometric averages of the smoothed language models each of which is a non-sparse vector of the size of the vocabulary with no non-zero components. Hence to estimate the SD parameters, it may seem that one needs to perform computations over the entire vocabulary for each document. However, it turns out there is an efficient way to do this estimation as shown below.

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}^T &= \frac{1}{Z}\left\{\prod_{i=1}^{N}\boldsymbol{\theta}^i\right\}^{\frac{1}{N}} && (3.29)\\
&= \frac{1}{Z}\left\{\prod_{i=1}^{N}\left(\lambda\hat{\boldsymbol{\theta}}^i + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE}\right)\right\}^{\frac{1}{N}} && (3.30)\\
&= \frac{1}{Z}\left\{\prod_{i=1}^{N}\left((\frac{\lambda\hat{\boldsymbol{\theta}}^i}{(1-\lambda)\hat{\boldsymbol{\theta}}^{GE}}+1)((1-\lambda)\hat{\boldsymbol{\theta}}^{GE})\right)\right\}^{\frac{1}{N}} && (3.31)\\
&= \frac{1}{Z}\left\{\prod_{i=1}^{N}\left(\frac{\lambda\hat{\boldsymbol{\theta}}^i}{(1-\lambda)\hat{\boldsymbol{\theta}}^{GE}}+1\right)((1-\lambda)\hat{\boldsymbol{\theta}}^{GE})^N\right\}^{\frac{1}{N}} && (3.32)\\
&= \frac{1}{Z}\left\{\prod_{i=1}^{N}\left(\frac{\lambda\hat{\boldsymbol{\theta}}^i}{(1-\lambda)\hat{\boldsymbol{\theta}}^{GE}}+1\right)\right\}^{\frac{1}{N}}(1-\lambda)\hat{\boldsymbol{\theta}}^{GE} && (3.33)
\end{aligned}$$

Notice that the term $\left(\frac{\lambda\hat{\boldsymbol{\theta}}^i}{(1-\lambda)\hat{\boldsymbol{\theta}}^{GE}}+1\right)$ in (3.33) is a vector that has component values of unity corresponding to all the terms that do not occur in the document D_i since $\hat{\theta}^i_j = 0$ for all such terms. Hence for all the components that correspond to the words that are absent in the document, the vector has no influence on the overall product. Hence we can perform the products by initializing the product vector to $\mathbf{1}$ and computing the products over only those words that occur in a document each time. The term $(1-\lambda)\hat{\boldsymbol{\theta}}^{GE}$, however, does involve product over the entire vocabulary and so does the normalizer Z, but they involve just one-time computation and hence can be considered constant in terms of the training

set size. Thus, one can estimate the parameters of the SD distribution nearly as efficiently as the parameters of the multinomial.

3.3.5 Inference using approximate SD distribution

In this section, we will look at the classification and ranking functions when SD is used as the underlying distribution to model topics.

In a classification task, for each document, the best topic T is chosen using the posterior probability of the topic $P(T|D)$ as shown below:

$$
\begin{aligned}
T_{best} &= \arg\max_T \log Pr_a(T|\boldsymbol{\theta}^D, \boldsymbol{\alpha}^T) = \arg\max_T \log Pr_a(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^T)P(T) && (3.34)\\
&\overset{class}{=} \arg\max_T \left(S\log S - \sum_j \{\alpha_j^T \log \alpha_j^T - (\alpha_j^T - 1)\log \theta_j^D\} \right) && (3.35)\\
&\overset{class}{=} \arg\max_T \{-\sum_j \alpha_j^T \log(\alpha_j^T/\theta_j^D)\} = \arg\max_T \{-KL(\boldsymbol{\alpha}^T||\boldsymbol{\theta}^D)\} && (3.36)
\end{aligned}
$$

where in step (3.36), we assume that $S = \sum_j \alpha_j^T$ is the same for all topics. Thus, we have shown that in case of TC, generative models based on the SD distribution result in a classifier that is class-equivalent to the KL-divergence between the class-parameters $\boldsymbol{\alpha}^T$ and the document language model $\boldsymbol{\theta}^D$. Note that this is proportional to the T-cross-entropy $H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D)$, but there is an additional term, namely the entropy of the class parameters $H(\boldsymbol{\alpha}^T) = -\sum_j \alpha_j^T \log \alpha_j^T$ that influences the classification. In effect, for any given document, the SD distribution chooses the class whose TCE w.r.t. the document language model is minimum but also one whose entropy of the class parameters $\boldsymbol{\alpha}^T$ is maximum (since $KL(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D) = -H(\boldsymbol{\alpha}^T) + H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D)$). It is not intuitively clear who one would have a preference for a topic whose parameters are as close to the uniform distribution (maximum entropy) as possible. This is a byproduct of our modeling approximations and it is not immediately clear if this property is necessarily desirable. Our empirical results in the next chapter will shed light on the utility of the KL-divergence function for classification.

Recall that in the case of ad hoc retrieval, we assume a binary classification framework where we have two parameter vectors $\boldsymbol{\alpha}^R$ and $\boldsymbol{\alpha}^N$ representing the relevant and non-relevant classes respectively. Ranking of documents is done using the posterior probability of relevance $P(R|D)$ as shown below.

$$
\begin{aligned}
P(R|\boldsymbol{\theta}^D, \boldsymbol{\alpha}^R, \boldsymbol{\alpha}^N) &\overset{rank}{=} \frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^N)} && (3.37)\\
&= \frac{\frac{S^S e^{-S}}{\prod_{j=1}^V \left(\alpha_j^R\right)^{\alpha_j^R} e^{-\alpha_j^R}} \prod_{j=1}^V \left(\theta_j^D\right)^{\alpha_j^R - 1}}{\frac{S^S e^{-S}}{\prod_{j=1}^V \left(\alpha_j^N\right)^{\alpha_j^N} e^{-\alpha_j^N}} \prod_{j=1}^V \left(\theta_j^D\right)^{\alpha_j^N - 1}} && (3.38)\\
&\overset{rank}{=} \prod_{j=1}^V \left(\theta_j^D\right)^{\alpha_j^R - \alpha_j^N} && (3.39)\\
&\overset{rank}{=} \sum_{j=1}^V \left(-\alpha_j^N \log \theta_j^D + \alpha_j^R \log \theta_j^D\right) && (3.40)\\
&\overset{rank}{=} H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D) && (3.41)
\end{aligned}
$$

where (3.39) follows from (3.38) by ignoring document independent terms that do not influence ranking and (3.40) uses the rank-equivalence property of the logarithmic function owing to its monotonicity w.r.t. its input. Thus SD distribution with the same parameterization as the Dirichlet distribution is rank-equivalent to the difference in T-cross entropies of the two class parameter vectors w.r.t. the document language model.

3.4 Data analysis

Recall our discussion in section 3.1 on previous research in text modeling that indicated that distributions that capture text better tend to perform better on classification tasks. In this work, we have followed the converse approach. Noticing the empirical success of the T-cross entropy ranking function and the absence of any justification for its particular choice, we constructed an approximate distribution that is rank-equivalent and nearly class-equivalent to the T-cross entropy function. It remains to be seen if this distribution also

models text well. If it indeed does model text accurately, it serves as a justification for the choice of T-cross entropy as a ranking function. In this section, we test this hypothesis empirically based on real data.

One of the popular metrics to measure the effectiveness of a distribution in modeling text is the perplexity measure used by Madsen et al [30]. In this measure, we estimate the parameters of the distribution from a training set of documents and compute the perplexity of an unseen set of test documents as follows:

$$\exp\left(\frac{-\sum_{i=1}^{N}\sum_{j=1}^{V}\log P(f_j^i)}{\sum_{i=1}^{N}|D_i|}\right) \tag{3.42}$$

where N is the number of documents in the test set. The lower the value of perplexity, the better is the ability of the distribution to predict test data. One can thus compare the ability of various distributions to model text by comparing their perplexity values on the same test data. In our case,the candidates are the multinomial, DCM, Dirichlet and SD distributions. Note that the former two are probability mass functions since they generate counts vectors $\mathbf{f}$, but the latter two are probability density functions since their domain is the set of multinomials (smoothed multinomials in the case of SD). Hence it is not very meaningful to compare the perplexity values of these distributions. Instead, we compared the ability of each of these distributions in fitting the empirical term occurrence distribution. One could construct an objective metric to measure the closeness of the predicted distributions to the empirical distributions. In this work, we generated comparative plots of predicted distributions versus empirical distributions and studied them only qualitatively.

We used a Porter-stemmed but not stopped version of Reuters-21578 corpus for our experiments. Similar to the work of Madsen *et al* [30],we sorted words based on their frequency of occurrence in the collection and grouped them into three categories, W_h, the high-frequency words, comprising the top 1% of the vocabulary and about 70% of the word occurrences, W_m, medium-frequency words, comprising the next 4% of the vocabulary and accounting for 20% of the occurrences and W_l, consisting of the remaining 95% low-

frequency words comprising only 10% of occurrences. We pooled within-document counts f of all words from each category in the entire collection and computed category-specific empirical distributions of proportions $Pr(f|W_h)$, $Pr(f|W_m)$ and $Pr(f|W_l)$. We used these distributions as ground truths in our experiments.

For our experiments, we first did maximum likelihood estimation of the parameters of Multinomial, DCM, Dirichlet and SD distributions using the entire collection. For Dirichlet and SD, we fixed the value of the smoothing parameter λ at 0.9. To train the Dirichlet and DCM distributions, we used iterative techniques to estimate the mean, keeping the precision S at constant, as described in [33] using the *fastfit*[2] toolkit.

In case of multinomial and DCM distributions, the probability that a word w_j occurs at count f_j in a document D, $P(f_j|\boldsymbol{\theta}, |D|)$ is given by their marginals, which are the binomial and the beta-binomial as shown below in (3.43) and (3.44) respectively.

$$P(f_j|\boldsymbol{\theta}, |D|) = \frac{(|D|)!}{f_j!(|D| - f_j)!}\theta_j^{f_j}(1-\theta_j)^{|D|-f_j} \tag{3.43}$$

$$Pr(f_j|\boldsymbol{\alpha}, |D|) = \frac{\Gamma(S)}{\Gamma(\alpha_j)\Gamma(S-\alpha_j)}\frac{\Gamma(\alpha_j + f_j)\Gamma(S + |D| - f_j)}{\Gamma(S + |D|)} \tag{3.44}$$

To compute the probability that it occurs at count f_j in any document, $P(f_j|\boldsymbol{\theta})$ or $P(f_j|\boldsymbol{\alpha})$, we marginalize the distribution over the document length using the following relations respectively:

$$P(f_j|\boldsymbol{\theta}) = \sum_{|D|} |D| \times P(f_j|\boldsymbol{\theta}, |D|)P(|D|) \tag{3.45}$$

$$P(f_j|\boldsymbol{\alpha}) = \sum_{|D|} |D| \times P(f_j|\boldsymbol{\alpha}, |D|)P(|D|) \tag{3.46}$$

where we estimated $P(|D|)$ empirically from the corpus.

[2]http://research.microsoft.com/∼minka/software/fastfit

Estimating the probability of count f_j is more tricky for Dirichlet and SD distributions because they generate language models and not counts. However, we can make use of the approximate equivalence relation in (3.17) to estimate the probability as follows:

$$P(f_j|\boldsymbol{\alpha}, |D|) \approx P(\theta_j = \lambda \frac{f_j}{|D|} + (1-\lambda)\hat{\theta}_j^{GE} \mid \boldsymbol{\alpha}, |D|) \tag{3.47}$$

where the probability $P(\theta_j|\boldsymbol{\alpha}, |D|)$ is given by the marginals of the Dirichlet and SD distributions. The marginal of the Dirichlet is the Beta distribution given by:

$$P(\theta_j|\boldsymbol{\alpha}, |D|) = \frac{\Gamma(S)}{\Gamma(\alpha_j)\Gamma(S-\alpha_j)} \theta_j^{\alpha_j - 1}(1-\theta_j)^{S-\alpha_j-1} \tag{3.48}$$

We assume that the marginal of the SD distribution has the same parametric form:

$$P_a(\theta_j|\boldsymbol{\alpha}) = \frac{S^S}{\alpha_j^{\alpha_j}(S-\alpha_j)^{S-\alpha_j}} (\theta_j)^{\alpha_j - 1}(1-\theta_j)^{S-\alpha_j-1} \tag{3.49}$$

For these distributions, the probability that a word w_j occurs at count f_j in a random document is given by

$$P(f_j|\boldsymbol{\alpha}) = \sum_{|D|} P(\theta_j = \lambda \frac{f_j}{|D|} + (1-\lambda)\hat{\theta}_j^{GE} \mid \boldsymbol{\alpha}, |D|) P(|D|) \tag{3.50}$$

Next, for each distribution, we compared average probabilities over the set of unique words in each category and normalized them over different values of f_j. We also tuned the value of the free-parameter S in DCM, Dirichlet and SD distributions until their plots were as close a visual-fit as possible to the empirical distributions. We caution that since we did not use any objective function to optimize the plots, they are only for illustration purposes.

Figure 3.5 compares the predictions of each distribution with the empirical distributions for each category. The data plots corresponding to empirical distribution exhibit a heavy tail on all three categories W_h, W_m and W_l as noticed by earlier researchers [41, 30].

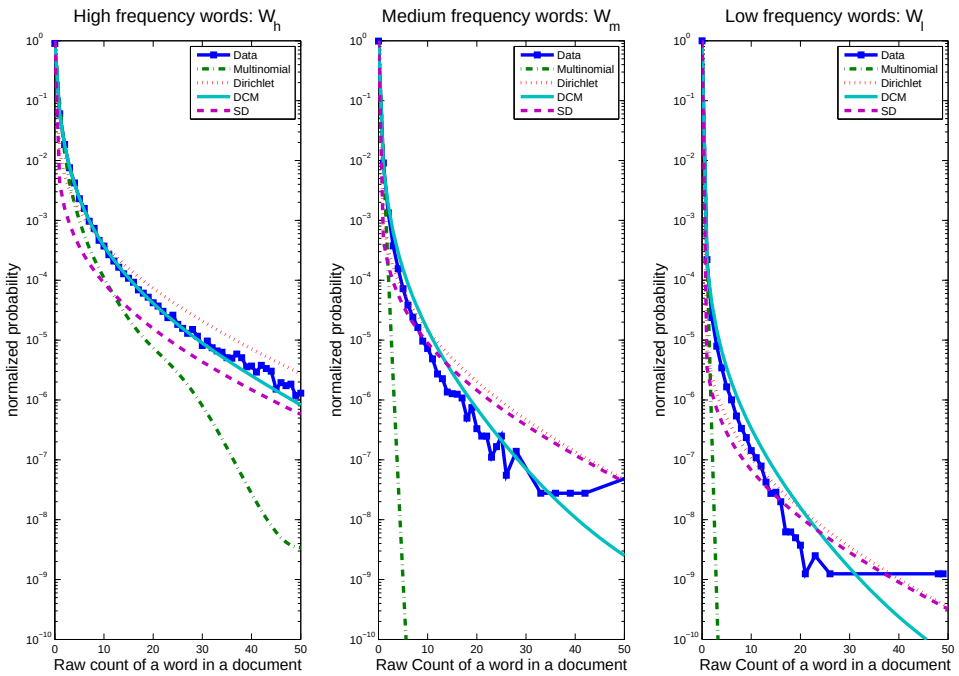

Figure 3.5. Comparison of predicted and empirical distributions

The multinomial distribution predicts the high frequency words well while grossly under-predicting the medium and low frequency words. High frequency words such as 'because', 'that', 'and' etc. are merely function words that carry no content while medium and low frequency words are content bearing. Since the multinomial fails to predict the burstiness of content bearing words, it is not surprising that the D-cross entropy function, that corresponds to the log-likelihood of documents w.r.t this distribution, is a poor performer in ad hoc retrieval. The plots also indicate that the DCM distribution is an excellent fit to data as shown by Madsen *et al* [30].

Notice that the Dirichlet and SD distributions fit the data much better than the multinomial on all three sets, validating our choice of the particular functional form to model text. The plots of SD, Dirichlet as well as DCM distributions also agree quite closely with each other on all three categories of words. Since this is only a qualitative comparison, it is hard to place one above the other in terms of their effectiveness in capturing the empirical distribution. Experiments on text classification in the next chapter will allow us to compare the distributions more objectively.

Most importantly, the plots allow us to justify the empirical success of the T-cross entropy function. Recall that a simple generative model based on the SD distribution is rank-equivalent to T-cross entropy, while the naïve Bayes classifier based on the multinomial distribution is approximately rank equivalent (save the document length factor) to D-cross entropy. Since the SD distribution models text much better than the multinomial, it is not surprising that its corresponding ranking function T-cross entropy is a better performer than its multinomial counterpart D-cross entropy. Thus our analysis offers a justification for the empirical success of T-cross entropy.

Recall that T-cross entropy is a popular ranking function in the language modeling approach to ad hoc retrieval, but it has never been used as a classifier in text classification. Now that we have built an approximate distribution underlying the T-cross entropy function, it is straightforward to apply a simple generative classifier based on this distribution

to text classification. The next chapter compares the performance of SD based generative classifier with that of the multinomial based naïve Bayes classifier as well as a classifier based on the DCM on various datasets.

CHAPTER 4

TEXT CLASSIFICATION

In the previous chapter, we defined a new SD distribution which is rank-equivalent to the T-cross entropy function. T-cross entropy is a successful ranking function in IR, but its effectiveness as a classifier remains to be tested. The new SD distribution allows us to define a simple generative classifier for text classification much like the naïve Bayes model. Since we have demonstrated that SD is a better model for text than the multinomial, we expect that a generative classifier based on SD will outperform the multinomial based naïve Bayes classifier. In this chapter, we will investigate the applicability of the SD distribution in text classification.

4.1 Indexing and preprocessing

We used the 20 Newsgroups [1], Reuters-21578 [2] and Industry-Sector[3] corpora for our experiments.

Stopping and stemming are two standard preprocessing steps in any IR system. Stopping consists of removing highly frequent non-content words such as 'the','at' *etc*. This operation not only saves space but also improves performance by focusing the model on content bearing words. We did stopping using a standard list of about 400 stop words.

Stemming involves collapsing morphological variants of words such as 'reads' and 'reading' into the same token 'read'. This not only makes our representation more compact,

[1]http://people.csail.mit.edu/jrennie/20Newsgroups/

[2]http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

[3]http://www.cs.umass.edu/ mccallum/code-data.html

but is also known to improve recall. All three collections used in our experiments were stemmed using the Porter stemmer [39].

For any of the collections or models, we did not do any feature selection, as we consider it a separate problem altogether. We indexed the collection using the *Lemur*[4] toolkit, version 3.0. We performed all our experiments on *Matlab* using the document-term matrix obtained from *Lemur*'s output.

The version of the 20 Newsgroups collection we used has 18,828 documents and 20 classes. We included the subject lines in the index since they do not reveal the topic directly. Our index consists of 116,199 unique tokens and an average document length of 150. The Industry-sector corpus has 9569 documents and 104 classes. Since documents in this collection are web pages, we used the HTML parser of *Lemur* to pre-process the documents. Our indexing resulted in 69,296 unique tokens and an average document length of 235. We randomly split documents in each class into train-test subsets at a ratio of 80:20 on both of these collections. We repeated this process 25 times to obtain versions of train-test splits to experiment on.

In case of Reuters-21578 collection, we used the standard Mod-Apte [3] subset of the Reuters-21578 collection that consists of 12,902 documents and a predefined train-test split. Our indexing resulted in 27,545 unique terms and an average document length of 81. We used only the 10 most popular classes for our experiments as done in [31]. In addition, to allow significance tests, we generated 25 random train test pairs of the Mod-Apte subset in the ration of 80:20, on which we performed separate experiments.

Additionally, to facilitate learning the values of free parameters, for each of the three collections, we randomly picked one of the training sets (the Mod-Apte in case of Reuters) and further randomly split them class-wise, in the same ratio as the corresponding train-test split, into sub-training and validation sets. The presence of free parameters in the models

[4]http://www.lemurproject.org

that cannot be learnt directly from Maximum Likelihood (ML) training necessitates such split.

4.2 Evaluation

While a number of metrics are in use to evaluate the performance of classifiers, we use the standard classification accuracy as the evaluation metric on the Industry sector and 20 Newsgroups data. Classification accuracy is defined as the percentage of test documents that are correctly labeled by the classifier. In other words, if for a document D_i, T_i is the true class and the class chosen by the classifier $\mathcal{F}$ is $T_i(\mathcal{F})$, then the accuracy is defined as

$$\text{Classification accuracy} = \frac{\sum_{i=1}^{N} \delta(T_i, T_i(\mathcal{F}))}{N} \tag{4.1}$$

where δ is an indicator function which is equal to 1 if $T_i = T_i(\mathcal{F})$ and 0 otherwise and N is the number of documents in the test set.

In the case of the Reuters collection, documents can belong to multiple classes. Classification accuracy is not an appropriate evaluation metric in this case because it implicitly assumes that each document can only belong to one class. Hence in this case, researchers recommend the IR approach: for each class, we rank all the test documents in the decreasing order of relevance to that class and measure the effectiveness of each ranked list and average this measure over all classes. Thus, if a document belongs to more than one class, we expect it to be placed high in the ranked lists corresponding to those classes. In effect,experiments on the Reuters collection correspond to the relevance feedback setting in IR where labeled documents are available for each query's topic.

To measure the effectiveness of each ranked list, Break Even Precision (BEP) is recommended as the evaluation metric [16]. BEP is defined in terms of Precision and Recall, which are defined as a function of the rank R as follows:

$$\begin{aligned} \text{Precision(R)} &= \frac{\text{Rel(R)}}{R} \\ \text{Recall(R)} &= \frac{\text{Rel(R)}}{N_{Rel}} \end{aligned} \tag{4.2}$$

where Rel(R) is the number of relevant documents in the ranked list $\{1, \cdots, R\}$ and N_{Rel} is the total number of documents relevant to the class. Thus precision measures the accuracy of the ranked list up to the rank R while recall measures the fraction of the total number of relevant documents covered up to the rank R. Break-even Precision, BEP, is now defined as the value of precision at which precision equals recall. It is easy to see from (4.2) that it is achieved when $R = N_{Rel}$. BEP is also sometimes referred to in the IR community as R-precision. We compute the BEP for each class and then average it over all classes. This averaging can be done in two different ways:

$$\text{Macro-BEP} = \frac{1}{|T|} \sum_{T} \text{BEP(T)} \tag{4.3}$$

$$\text{Micro-BEP} = \frac{1}{\sum_T N_{rel(T)}} \sum_{T} N_{rel}(T) \times \text{BEP(T)} \tag{4.4}$$

where the summation is over the set of classes whose size is given by $|T|$. Thus Macro-BEP considers all classes as equally important while Micro-BEP considers all documents to be equally important. In our evaluation, we chose Macro-BEP. We note that since we chose only the top 10 Reuters categories which roughly have equal number of relevant documents per class, the values of micro and macro BEPs should be comparable.

Experiments on the Reuters collection are of special interest to us considering its correspondence to IR, since we do ranking and not hard labeling of documents in this case. Hence, the parameter estimation and inference techniques used in the Reuters collection are directly applicable to the ad hoc retrieval task.

4.3 Parameter estimation

The parameters of the distribution associated with each topic are typically estimated using maximum likelihood estimation. As described in section 1.3.2.2, given a set of training set of documents $\mathcal{D}^T$ for each topic T, the maximum likelihood estimates of an underlying generative distribution $\hat{\boldsymbol{\beta}}^T$ is given by

$$\hat{\boldsymbol{\beta}}^T = \arg\max_{\boldsymbol{\beta}} \prod_{i=1}^{|\mathcal{D}^T|} P(D_i|\boldsymbol{\beta}) \tag{4.5}$$

In case of the Reuters collection, for each topic T, we assume two classes R and N corresponding to the relevance and non-relevance classes as in the IR setting. The parameters for the relevance class of the underlying distribution $\boldsymbol{\beta}^R$ are estimated from the training documents of topic T as shown above in (4.5). The parameters of the non-relevant class are estimated from training documents of all other classes as shown below:

$$\hat{\boldsymbol{\beta}}^N(T) = \arg\max_{\boldsymbol{\beta}} \prod_{k:T_k \neq T} \prod_{i=1}^{|\mathcal{D}^{T_k}|} P(D_i|\boldsymbol{\beta}) \tag{4.6}$$

In this work, we consider generative classifiers employing the multinomial (naïve Bayes classifier), the DCM, Dirichlet and SD distributions as candidates for comparison. For the multinomial, the MLEs $\hat{\boldsymbol{\theta}}^T$ of class T correspond to the normalized counts of words in training documents as shown below.

$$\hat{\boldsymbol{\theta}}^T = \frac{\sum_{i=1}^{\mathcal{D}^T} \mathbf{f}^i}{\sum_{i=1}^{\mathcal{D}^T} |D_i|} \tag{4.7}$$

For the DCM and Dirichlet distributions, there is no closed form solution for maximum likelihood estimation. We fixed the scale $S = \sum_j \alpha_j$ at a constant value for all classes and we estimated the mean MLE values $\hat{\boldsymbol{\alpha}}^T$ using a conjugate gradient descent technique as described in [33], using the *fastfit* toolkit. The MLE values of SD parameters $\hat{\boldsymbol{\alpha}}^T$ are given

by the geometric averages of the smoothed language models of documents in the training set as shown in (3.28).

One problem that is characteristic of text classification is the Out-Of-Vocabulary (OOV) problem, which is the possibility that the test documents contain words that are unseen in the training data. MLE training assigns non-zero parameter values to only those words seen in the training and the rest get zero values by default. This may result in assignment of zero probability to many test documents that may contain OOV words, which is clearly undesirable. To overcome this problem, MLE estimates are typically smoothed. We describe smoothing techniques employed for various distributions below.

For the multinomial, we used two kinds of smoothing as shown below:

$$\text{Laplacian:} \qquad \boldsymbol{\theta}^T = \frac{\sum_{i \in \mathcal{D}^T} \mathbf{f}^i + \delta}{\sum_{i \in \mathcal{D}^T} \sum_j f_j^i + V\delta} \tag{4.8}$$

$$\text{Jelinek-Mercer:} \qquad \boldsymbol{\theta}^T = \lambda \hat{\boldsymbol{\theta}}^T + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE} \tag{4.9}$$

where λ and δ are free parameters. $\hat{\boldsymbol{\theta}}^{GE}$ is estimated as follows:

$$\hat{\boldsymbol{\theta}}^{GE} = \frac{\sum_i \mathbf{f}^i + \beta}{\sum_i \sum_j f_j^i + V\beta} \tag{4.10}$$

where the index i ranges only over the entire set of training documents and β is another free parameter. Note that $\hat{\boldsymbol{\theta}}^{GE}$ is also smoothed in this case because the MLE distribution would result in zeros for OOV words. Laplacian smoothing shown in (4.8) is more common in text classification research while Jelinek-Mercer (JM) smoothing shown in (2.1) is popular in IR and is shown to boost performance [60]. We tried both techniques in our experiments for comparison. For the DCM model, smoothing is done as follows:

$$\boldsymbol{\alpha}^T = S\frac{\hat{\boldsymbol{\alpha}}^T + \delta}{\sum_j \hat{\alpha}_j^{MLE} + V\delta} \tag{4.11}$$

For the Dirichlet and SD distributions, we use smoothed document proportions as shown in (2.1) but $\hat{\boldsymbol{\theta}}^{GE}$ used in smoothing corresponds to the estimate in (4.10), so we do not expect any zeros in our parameter estimates.

For DCM and Dirichlet, we consider S as a free parameter. In SD, the value of S does not influence parameter estimation. Hence we fix $S = 1$, allowing us to treat the SD parameter vector $\boldsymbol{\alpha}$ as a probability distribution over the vocabulary.

To learn the optimal values of the free parameters of the models, we performed a simple hill-climbing on the domain of the free parameters until the evaluation criterion is optimized on the validation set. We then performed regular maximum likelihood training and testing on all train-test splits, fixing the free parameters at these optimal values. On Industry sector and 20 Newsgroups corpora, we performed statistical significance tests using the two-tailed paired T-test as well as the sign test, both at a confidence level of 95%.

4.4 Inference

As explained in section 1.3.1.1 generative classifiers consist of two components for each class T: the class conditional $P(D|\boldsymbol{\beta}^T)$ and the prior $P(T)$, where $\boldsymbol{\beta}^T$ are the parameters of the underlying generative distribution. The classifier chooses the class that maximizes the posterior $P(T|D)$ which can be computed using the Bayes rule as follows.

$$\begin{aligned} P(T|D) &\overset{class}{=} P(D|\boldsymbol{\beta}^T)P(T) \\ &\overset{class}{=} P(D|\boldsymbol{\beta}^T) \text{ assuming uniform prior P(T) over all classes} \end{aligned} \tag{4.12}$$

In case of the Reuters collection, the setting corresponds to an IR scenario: the test documents are not classified into one of the topics, but are are ranked against each topic. In this case, as described by several other authors[31, 30, 41], for each topic T, the documents are then ranked according to the posterior probability of relevance w.r.t. the topic as shown below.

$$P(R|D) \overset{rank}{=} \log P(D|\boldsymbol{\beta}^R) - \log P(D|\boldsymbol{\beta}^N) \tag{4.13}$$

where the rank-equivalence shown above follows from the derivation presented in section 2.3 (see (2.14)). We have already shown in section 2.2 that in case of hard classification, the posterior probability of class when the topics are modeled by the multinomial distribution is class-equivalent to D-cross entropy $H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^T)$ (see (2.11)). In case of Reuters, the posterior probability of relevance is rank-equivalent to $|D| \left(H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^N) - H(\hat{\boldsymbol{\theta}}^D, \boldsymbol{\theta}^R)\right)$ as shown in (2.18). Note that since the document model in this case is the unsmoothed MLE model $\hat{\boldsymbol{\theta}}^D$, the summation in computing the cross-entropy terms is only over the terms that occur in the document, and hence is relatively inexpensive to compute.

In case of the DCM distribution, the classification function, assuming uniform prior $P(T)$ again, is given by the document log-likelihood as shown below:

$$\begin{aligned}
P_{DCM}(T|D) &\overset{class}{=} \log P(D|\boldsymbol{\alpha}^T) = \log \frac{\Gamma(\sum_{j=1}^V \alpha_j^T)}{\prod_{j=1}^V \Gamma(\alpha_j^T)} \frac{\prod_{j=1}^V \Gamma(\alpha_j^T + f_j^D)}{\Gamma\left(\sum_{j=1}^V (\alpha_j^T + f_j^D)\right)} && (4.14)\\
&\overset{class}{=} \log \frac{\prod_{j=1}^V \Gamma(\alpha_j^T + f_j^D)}{\prod_{j=1}^V \Gamma(\alpha_j^T)} && (4.15)\\
&\overset{class}{=} \log \prod_{j:w_j \in D} \left(\frac{\Gamma(\alpha_j^T + f_j^D)}{\Gamma(\alpha_j^T)}\right) && (4.16)\\
&= \sum_{j:w_j \in D} \left(\log \Gamma(\alpha_j^T + f_j^D) - \log \Gamma(\alpha_j^T)\right) && (4.17)
\end{aligned}$$

where (4.15) follows from (4.14) using the assumption that $S = \sum_j \alpha_j^T$ is the same for all topics. In (4.15), the numerator and denominator are identical for terms that do not occur in the document, *i.e.*, $f_j^D = 0$, resulting in (4.16) that goes over only terms that occur in the document. Thus the classification function of DCM based classifier is only marginally more expensive than the multinomial because of the computation of Gamma functions. Recall however, that the parameter estimation of DCM is much more expensive than multinomial.

The ranking function corresponding to DCM based classifier can be simplified as follows:

$$
\begin{aligned}
P_{DCM}(R|D) &\overset{rank}{=} \log P(D|\boldsymbol{\alpha}^R) - \log P(D|\boldsymbol{\alpha}^N) && (4.18)\\
&\overset{rank}{=} \log\left\{\frac{\prod_{j=1}^{V}\Gamma(\alpha_j^R + f_j^D)}{\Gamma\left(\sum_{j=1}^{V}(\alpha_j^R + f_j^D)\right)}\right\} - \log\left\{\frac{\prod_{j=1}^{V}\Gamma(\alpha_j^N + f_j^D)}{\Gamma\left(\sum_{j=1}^{V}(\alpha_j^N + f_j^D)\right)}\right\} && (4.19)\\
&\overset{rank}{=} \sum_{j=1}^{V}\left(\log\Gamma(\alpha_j^R + f_j^D) - \log\Gamma(\alpha_j^N + f_j^D)\right) && (4.20)\\
&= \sum_{j=1}^{V}\left(\log\frac{\Gamma(\alpha_j^R + f_j^D)}{\Gamma(\alpha_j^R)} - \log\frac{\Gamma(\alpha_j^N + f_j^D)}{\Gamma(\alpha_j^N)} + \log\frac{\Gamma(\alpha_j^R)}{\Gamma(\alpha_j^N)}\right) && (4.21)\\
&\overset{rank}{=} \sum_{j:w_j\in D}\left(\log\frac{\Gamma(\alpha_j^R + f_j^D)}{\Gamma(\alpha_j^R)} - \log\frac{\Gamma(\alpha_j^N + f_j^D)}{\Gamma(\alpha_j^N)}\right) && (4.22)
\end{aligned}
$$

where in (4.19) we ignored document independent terms in the generative probability w.r.t the DCM distribution while in (4.20), we assumed $\sum_j \alpha_j^R = \sum_j \alpha_j^N = S$. Step (4.21) is a simple algebraic manipulation of (4.20) which allows us to write the ranking function purely in terms of the words that occur in the document. Notice that the ranking function w.r.t. the DCM in (4.22) is very similar to the classification function in (4.17) except for the fact that there is an extra similar term corresponding to the non-relevant class.

Using a similar analysis for the Dirichlet distribution and assuming the same value of S for parameters of all classes, we have already shown that the corresponding classification and ranking functions are given by (3.11) and (3.7) respectively. They are reproduced below for convenience.

$$
\begin{aligned}
P_{Dir}(T|D) &\overset{class}{=} -\sum_{j=1}^{V}\log\Gamma(\alpha_j^T) - H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D)\\
P_{Dir}(R|D) &\overset{rank}{=} H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D) && (4.23)
\end{aligned}
$$

Similarly, the classification and ranking functions w.r.t the SD distribution are derived in (3.36) and (3.41) respectively and are reproduced below.

$$P_{SD}(T|D) \overset{class}{=} -KL(\boldsymbol{\alpha}^T||\boldsymbol{\theta}^D)) \tag{4.24}$$

$$P_{SD}(R|D) \overset{rank}{=} H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D) \tag{4.25}$$

Since we use a smoothed representation for documents, none of the components of $\boldsymbol{\theta}^D$ is a zero and as such computing any entropy term involves summation over the entire vocabulary. Computing such summation over the entire vocabulary is very expensive. We used the following algebraic simplification of cross entropy that makes the computational effort almost the same as that for the multinomial and DCM distributions.

$$\begin{aligned}
-H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D) &= \sum_j \alpha_j^T \log \theta_j^D && (4.26)\\
&= \sum_j \alpha_j^T \log \left(\lambda \hat{\theta}_j^D + (1-\lambda)\hat{\theta}_j^{GE}\right) && (4.27)\\
&= \sum_j \alpha_j^T \left\{\log\left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1\right) + \log\left((1-\lambda)\hat{\theta}_j^{GE}\right)\right\} && (4.28)\\
&= \sum_{j:w_j \in D} \alpha_j^T \log\left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1\right) + \sum_j \alpha_j^T \log\left((1-\lambda)\hat{\theta}_j^{GE}\right) && (4.29)
\end{aligned}$$

Notice that the first term in (4.28) vanishes for all the terms that don't occur in the document because $\hat{\theta}_j^D = 0$ for all such terms. This observation allows us to rewrite the first term as a summation over only those words that occur in the document as shown in (4.29). Only the second term involves summation over the entire vocabulary, but this term is document independent and can be safely ignored in ranking. Using the result in (4.29), one can rewrite (4.25) as follows.

$$\begin{aligned}
P(R|D) &\overset{rank}{=} H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D) && (4.30)\\
&\overset{rank}{=} \sum_{j:w_j \in D} \left(\alpha_j^R - \alpha_j^N\right) \log\left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1\right) && (4.31)
\end{aligned}$$

Notice the close resemblance of the term weighting in this ranking function with TFIDF weights in the vector space model given by

$$TFIDF(w_j, D) = \log(TF) \times \log(IDF) = \log(f_j^D) \times \log(\frac{N}{N_j}) \tag{4.32}$$

where N is the number of documents in the collection and N_j is the number of documents that the word w_j occurs in. In our case, $\hat{\theta}_j^D$ and $1/\hat{\theta}_j^{GE}$ closely correspond to TF and IDF respectively and the weight of a word in the document roughly corresponds to $\log(TF \times IDF)$.

On similar lines, using the simplification in (4.29), the classification function in (4.24) can be rewritten using class equivalence relationship as follows:

$$\begin{aligned}
P_{SD}(T|D) &\overset{class}{=} -KL(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D) = H(\boldsymbol{\alpha}^T) - H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D) && (4.33)\\
&\overset{class}{=} -\sum_j \alpha_j^T \log \alpha_j^T + \sum_{j:w_j \in D} \alpha_j^T \log\left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1\right)\\
&\quad + \sum_j \alpha_j^T \log\left((1-\lambda)\hat{\theta}_j^{GE}\right) && (4.34)\\
&\overset{class}{=} -\sum_j \alpha_j^T \log \frac{\alpha_j^T}{\hat{\theta}_j^{GE}} + \sum_{j:w_j \in D} \alpha_j^T \log\left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1\right) && (4.35)\\
&= -KL(\boldsymbol{\alpha}^T, \hat{\boldsymbol{\theta}}^{GE}) + \sum_{j:w_j \in D} \alpha_j^T \log\left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1\right) && (4.36)
\end{aligned}$$

In (4.35), the term $\sum_j \alpha_j^T \log(1-\lambda) = S \log(1-\lambda)$ drops out under the assumption that all the classes have the same value of S. Note that the first term in (4.36) prefers classes whose topic models $\boldsymbol{\alpha}^T$ are distinct from the general English model $\hat{\boldsymbol{\theta}}^{GE}$ as measured by the KL-divergence function. The second term can be related to the usual TF-IDF weights in the document. Note that the KL-divergence term needs summation over the entire vocabulary but need not be computed for every document. It can be pre computed once and stored in memory for each class and can be reused for all documents. Hence inference in both SD and Dirichlet distributions is as nearly as fast as that of the multinomial and DCM distributions. Recall that training the Dirichlet is computationally intensive since it has no closed form solution. The SD however resolves this problem: its MLE solution is a

simple geometric averages of document language models and is as efficient to compute as the multinomial as discussed in section 3.3.4.1).

In addition to the generative classifiers described above, we also tested a linear Support Vector Machine (SVM) as a standard discriminative baseline using a one-versus-all SVM^{light} toolkit for Reuters and SVM^{multiclass} toolkit for the other two data-sets [17]. SVMs are considered the state of the art machines in text classification and it only makes sense to include them as a benchmark along with other generative classifiers. Note that our objective is not necessarily to outperform the best performing classifier in the market, but more importantly to test the performance of the SD distribution and thereby the T-cross entropy function in the context of text classification, in relation to other existing generative classifiers. As features of the SVM, we used normalized TF-IDF weights defined by tf$\times \log((N+1)/(n+0.5))$ where tf is the raw count of a term in a document, N is the total number of training documents and n is the number of training documents the term occurs in. We used the parameter C that represents trade-off between margin maximization and training error as a free parameter during training. Although SVM is known to achieve its best performance on text classification when used in combination with a feature selection algorithm, we did not use it in this case so that the comparison is fair, since the generative classifiers use the full vocabulary set. Hence our SVM results may not reflect the best performing SVM but it still serves as a good baseline.

4.5 Results

In this section, we present the results of our experiments on the three collections. We present the results on Reuters separately because it involves ranking while the other two involve hard classification.

The notation in tables 4.1 and 4.2 is as follows. Mult-L and Mult-JM correspond to Multinomial with Laplace and Jelinek-Mercer smoothing respectively and Dir is Dirichlet, while the rest have their usual meaning. The symbols in the parentheses in column 2

	Model (par)	Opt Par	BEP (%) Mod-Apte	Mean BEP (%) rand. splits
1	Mult-L (δ)	10^{-2}	71.42	$60.05 \pm 0.78_6$
2	Mult-JM (λ, β)	$0.7, 10^{-3}$	72.78	$60.00 \pm 0.83_6$
3	DCM (S, δ)	$1500, 0.1$	72.38	$60.21 \pm 0.78_6$
4	Dir (S, λ, β)	$3000, 0.2, 10^{-3}$	74.87	$64.24 \pm 0.82_{A-5}$
5	SD (λ, β)	$10^{-2}, 1$	$\mathbf{79.24}$	$\mathbf{65.64 \pm 0.84_A}$
6	SVM (C)	20	76.21	59.25 ± 1.13

Table 4.1. Performance comparison on Reuters Corpus

indicate the free parameters of each distribution. For reproducibility of our experiments, we present the respective optimal parameter settings learned from training in each data set in columns titled "Opt Par". Bold-face number indicates the best performing model on the corresponding data set. A subscript $_i$ on an entry in columns 4 and 6 represents that the corresponding model is significantly better than the model whose serial number is i according to both the paired 2-tailed T-test and the sign test at 95% Confidence interval on the 25 random train-test splits. The notation $_A$ in the subscript implies the model is significantly better than all other models, while $_{A-i}$ indicates the model is better than all but the model numbered i.

4.5.1 Ranking:Reuters Collection

Table 4.1 presents the results on the Reuters corpus. We report the values of BEP on the standard Mod Apte train-test split. On the 25 random train-test splits of this corpus, we also present standard deviation and statistical significance results. (The statistical significance results are identical w.r.t. both the T-test and the sign test.)

Our experiments show that the multinomial based naïve Bayes classifier using the two kinds of smoothing and the DCM based classifier are statistically indistinguishable from one another. Note that Jelinek Mercer smoothing is expected to perform better than Laplacian smoothing in IR [60] since it partially overcomes the scarce feature problem by inducing features from neighbors. We believe the main reason for its indistinguishability

from the Laplacian in this case is the relative distinctness of classes from one another and the discriminative power of the features, thereby obviating the need for feature induction. The fact that the DCM does not register any significant improvements over the multinomial naïve Bayes on the Reuters collection is also observed by [30] (compare the entries of the multinomial with Laplacian smoothing $\epsilon = 0.01$, and DCM in table 3).

The Dirichlet based classifier performs better than all the aforementioned classifiers confirming our intuition behind the choice of the distribution. Since the Dirichlet distribution is rank equivalent to T-cross entropy, the result once also demonstrates the effectiveness of T-cross entropy as a ranking function. The best classifier turns out to be the one based on SD. Our results on the 25 random train-test splits show that SD based classifier's improvement in performance compared to all other classifiers is statistically significant w.r.t. both the tests, thus validating the choice of our distribution. We also note that the SD classifier significantly outperforms even better than the SVM baseline on this dataset.

Notice that although the results on Mod-Apte and the random splits follow similar trend, the results on the latter are numerically lower than those on the former. While we do not have an intuitive explanation for this phenomenon, we note that Koller and Tong [53], in their work on active learning, report similar low values when random test-train splits are used on the Reuters corpus (60% BEP for SVM when 100 labeled documents are used for training in each of the top 10 categories). Most importantly, notice that the SD based classifier significantly outperforms all other generative classifiers as well as the SVM baseline on both Mod Apte as well as random splits.

4.5.2 Classification: 20 Newsgroups and Industry Sector

In this subsection, we present the results of our experiments on the 20 Newsgroups and industry sector datasets, which involves hard classification.

Note that the ranking function of the SD classifier is equivalent to the T-cross entropy function but the classifier is class-equivalent to the negative KL-divergence as shown in

#	Dataset →	20 Newsgroups		Industry Sector	
	Model ↓	Opt Params	% Accuracy	Opt Params	% Accuracy
1	Mult-L (δ)	10^{-3}	$87.02 \pm 0.46_{2,4,7}$	10^{-4}	$73.92 \pm 0.65_{3}$
2	Mult-JM (λ, β)	$10^{-2}, 10^{-3}$	86.41 ± 0.46	$10^{-3}, 10^{-3}$	$84.91 \pm 0.67_{1,3,4}$
3	DCM (S, δ)	$900, 10^{-4}$	$88.04 \pm 0.51_{1,2,7}$	$1200, 10^{-3}$	71.01 ± 1.07
4	Dir (S, λ, β)	$400, 0.1, 10^{-2}$	86.54 ± 0.44	$1800, 10^{-1}, 10^{-2}$	$76.97 \pm 0.91_{1,3}$
5	SD (λ, β)	$10^{-4}, 0.1$	$89.72 \pm 0.47_{A-6}$	$10^{-3}, 10^{-3}$	$80.82 \pm 0.93_{1,3,4}$
6	SD-CE (λ, β)	$10^{-4}, 10^{-3}$	$\mathbf{90.56} \pm 0.42_{A}$	$10^{-5}, 10^{-3}$	$86.22 \pm 0.62_{A-7}$
7	SVM (C)	1.0	86.48 ± 0.65	1.0	$\mathbf{88.20 \pm 0.71}_{A}$

Table 4.2. Performance comparison on the 20 News groups and Industry sector data sets: subscripts reflect significance as described at the start of section 4.5.2 (Statistical significance results are identical w.r.t. the T-test and the sign test.)

(4.24). Its maximization results in minimization of the T-cross entropy $H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D)$ similar to a ranking setting, but also results in maximizing the entropy $H(\boldsymbol{\alpha}^T)$. Noting the success of cross-entropy in our experiments on the Reuters collection, we also tested a variant of the KL-divergence based SD classifier, called SD-CE, which uses only the cross-entropy term $H(\boldsymbol{\alpha}^T, \boldsymbol{\theta}^D)$ for classification, the rest being the same as SD.

Table 4.2 presents the results of our experiments on the two datasets. DCM performs better than the Laplacian based multinomial on 20 news groups however, it is marginally lower in the industry sector. The Jelinek Mercer based multinomial, although marginally worse than the Laplacian on 20 news groups, achieves significant improvement on the industry sector corpus. These results are in line with those of [32] wherein the authors performed a similar smoothing in a hierarchical classification setting which they called shrinkage. We believe the main reason for the remarkable improvement in the Industry sector corpus is the relatedness of many classes where 'borrowing' of features from other classes through the Jelinek Mercer smoothing helps in learning a good classification rule. This should not be very surprising since research suggests that a multinomial mixture, such as the one we use in Jelinek Mercer smoothing, captures informative words much better

than a single multinomial [40]. In the IR community, it is a known fact that Jelinek Mercer smoothing typically outperforms the Laplacian smoothing [60].

The results also show that SD outperforms the Laplace smoothed multinomial, the DCM and ordinary Dirichlet on all collections. On these two collections on which we could do significance tests, the difference with the nearest model is found to be statistically significant. In addition, the SD distribution is also consistently and significantly better than the ordinary Dirichlet distribution, justifying the intuition behind our definition of the new distribution.

Our approximation to the SD inference, SD-CE, outperforms all distributions including JM smoothed Multinomial and on all collections justifying our intuition behind the modified inference formula in SD-CE. Note that the SD based classifier we used in our experiments is based on the approximate SD distribution shown in (3.25). Unlike in ranking where SD is simply equivalent to T-cross-entropy as shown in (3.41), in classification, the normalizer of the SD distribution does influence the decision (see (3.34)). The approximate SD normalizer, although qualitatively similar to the exact normalizer, is quantitatively not accurate. Hence the classification rule of the approximate SD does not correspond to the classification rule w.r.t. the true SD distribution. Although SD-CE has the same approximate parameter estimates as the approximate SD, we believe it achieves improvement in performance by ignoring the inaccurate normalizer of the approximate SD. Ignoring the normalizer altogether invariably results in new inaccuracies. Hence we expect the exact SD distribution, being the true distribution for smoothed language models, to outperform SD-CE on classification. However estimating the parameters of the exact SD distribution and computing its normalizer can be computationally expensive since the exact SD is not analytically tractable. We do not address this issue in this thesis and consider this as part of our future work.

The results also show that the SD distribution performs better than the linear SVM baseline on 2 of the 3 datasets confirming its effectiveness as a classifier. We hasten to add

that it is possible to further boost the performance of SVMs by defining better features or by doing good feature selection. The main aim of our experiments is not to outperform the best classifier but to demonstrate the effectiveness of the SD distribution as an elegant and effective distribution for text.

4.5.3 Comparison with previous work

Comparing with results from other work, we note that our multinomial results agree quite closely with the results in [31] on all three collections. Our SVM results on 20 Newsgroups agree very well with the SVM baseline in [41]. Our results are slightly lower on Industry sector (our 88.20% vs. their 93.4%) while higher on Reuters (our 79.24% vs. their 69.4%). The difference in Reuters is primarily because we used the top 10 classes while they used 90 classes. Our SVM results on Reuters are slightly lower than those reported in [16] (our 76.21% vs. their 82.51% in Macro-BEP). For SVM features, we computed IDF values from only the training documents to make for a fair comparison with the generative distributions that used smoothing only with the training documents. It is not clear how IDF is computed in [41] and [16]. Also, while we used basic TF-IDF features, they used several ad-hoc transformations to the features that resulted in improved performance. Further, our preprocessing and indexing resulted in a significantly higher number of unique tokens on all collections than in [16], making comparison difficult. However, the trends are quite similar in that, SVM outperforms multinomial distribution (the 20 Newsgroups data being an exception in our case).

The work that is most related to ours is that of Madsen *et al* [30]. Their results on Reuters are not exactly comparable because they used 90 classes with at least one training and one test document while we used only the top 10 classes for faster experimentation, following several other researchers ([53] for example). On other collections, they used precision as the evaluation metric while we used the more popular classification accuracy.

But our results are consistent with theirs in that, in general, DCM is shown to be better performing than the Laplace smoothed multinomial.

4.6 Conclusions

Our results clearly demonstrate that the SD distribution, underlying the successful T-cross entropy function in IR, is also a successful performer in text classification. The results on all three collections show that SD based classifier is better than other known generative classifiers such as the multinomial based naïve Bayes classifier, the DCM and Dirichlet distributions.

We would also like to emphasize that besides performance, another attractive property of the SD distribution is its relatively inexpensive training owing to its closed form MLE solution: SD takes at least an order of magnitude less computational time than DCM and Dirichlet and the SVM models and almost the same time as the multinomial, while performing at least as well as any of these models.

CHAPTER 5

INFORMATION RETRIEVAL

In this chapter, we treat information retrieval as a binary classification problem and apply the SD based classifier we presented in the last few chapters to the task of ad hoc retrieval. Since we have shown that SD distribution overcomes some of the weaknesses of the multinomial distribution, we expect an SD based generative classifier to perform well on this task.

We compare the performance of the SD based classifier with one of the state-of-the-art language modeling approaches for ad hoc retrieval. The likely candidates are the Relevance Model (RM) [26] and model based feedback [59]. In this work, we choose the Relevance Model for comparison since it has not only been very successful in the ad hoc retrieval task but also has been widely popular in other tasks such as tracking [24], cross-lingual retrieval [25] and image annotation [15].

In addition, we will analyze the estimation techniques of the RM and offer new insights into the approach by drawing parallels to the generative SD classifier.

Since both the RM and SD based generative classifier are based on the T-cross-entropy function, we expect similar performance on the ad hoc retrieval task. Our experiments in this chapter serve to establish this equivalence. In addition, they also demonstrate that our view of ad hoc retrieval as a classification problem is justified, provided the choice of the underlying distribution is appropriate.

5.1 Relevance model for ad hoc retrieval

In this section, we will describe the Relevance Model (RM), one of the state-of-the-art models for ad hoc retrieval based on the language modeling framework. In this approach, documents are initially ranked w.r.t. their query-likelihood as shown in (1.4). The language model associated with the query's topic, called the relevance model θ^R is then estimated from the top ranking documents in the initial retrieval as follows [23].

$$\begin{aligned}
\boldsymbol{\theta}^R &= E[\boldsymbol{\theta}|Q] = \int_{\boldsymbol{\theta}} \boldsymbol{\theta} P(\boldsymbol{\theta}|Q) d\boldsymbol{\theta} && (5.1)\\
&\approx \sum_{i=1}^{N} \boldsymbol{\theta}^i P(\boldsymbol{\theta}^i|Q) && (5.2)\\
&= \sum_{i=1}^{N} \boldsymbol{\theta}^i \frac{P(Q|\boldsymbol{\theta}^i)P(\boldsymbol{\theta}^i)}{\sum_{i=1}^{N} P(Q|\boldsymbol{\theta}^i)P(\boldsymbol{\theta}^i)} && (5.3)\\
&= \sum_{i=1}^{N} \boldsymbol{\theta}^i \frac{P(Q|\boldsymbol{\theta}^i)}{\sum_{i=1}^{N} P(Q|\boldsymbol{\theta}^i)} && (5.4)
\end{aligned}$$

As shown in (5.1), the RM is defined as the expected value of the language model given the evidence that the user's query is generated from it. In step (5.2), we approximate the expectation over the entire multinomial simplex to an average over N language models corresponding to the top N documents retrieved during the initial query-likelihood retrieval. The language model for each document D_i is computed using a smoothed estimated as shown in (2.1). In step (5.3), we use Bayes rule to express the posterior probability in terms of the prior $P(\boldsymbol{\theta}^i)$ and the class-conditional $P(Q|\boldsymbol{\theta}^i)$, while in step (5.4), we assume a uniform prior for all document language models. Thus in effect, the RM is a weighted average of the document language models corresponding to the top ranking documents in the query-likelihood run. One can also think of the RM as a weighted nearest neighbor approach where the neighbors (top ranking documents) are weighted by the closeness of the document models to the query as measured by query-likelihood.

Let us examine the query-likelihood weight assigned to the documents by the RM in more detail. One can further simplify the query-likelihood weights assigned by the RM as

follows.

$$\log P(Q|\boldsymbol{\theta}^D) = \sum_{j=1}^{V} f_j^Q \log(\theta_j^D) \tag{5.5}$$

$$= \sum_{j=1}^{V} f_j^Q \left\{ \log(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1) + \log\left((1-\lambda)\hat{\theta}_j^{GE}\right) \right\} \tag{5.6}$$

$$= \sum_{j:w_j \in D,Q} f_j^Q \log(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1) + \sum_{j:w_j \in Q} f_j^Q((1-\lambda)\hat{\theta}_j^{GE}) \tag{5.7}$$

Using the above result, the weight assigned by the RM to each document can be written as follows.

$$W_{RM}(D_i) = \frac{P(Q|\boldsymbol{\theta}^i)}{\sum_{k=1}^{N} P(Q|\boldsymbol{\theta}^k)}$$

$$= \frac{1}{Z} \exp\left\{ \sum_{j:w_j \in D,Q} f_j^Q \log(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1) \right\} \tag{5.8}$$

where the document independent term in (5.7) drops away in the normalization. The normalizer Z in (5.8) simply sums up the numerator term corresponding to all top ranking documents. We will compare this weight to the weight assigned to the documents by SD in the subsequent discussion.

Once the RM is estimated, a second retrieval step is executed in which documents are ranked according to T-cross entropy as shown below.

$$\text{Score(D,Q)} = -H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^D) = \sum_j \theta_j^R \log \theta_j^D \tag{5.9}$$

5.2 SD based generative classifier

As described in the introductory chapter, we consider IR as a problem of classifying documents into relevant and non-relevant classes with corresponding SD parameters $\boldsymbol{\alpha}^R$ and $\boldsymbol{\alpha}^N$ respectively. For simplicity, we assume that both the classes have the same precision $S = \sum_j \alpha_j^R = \sum_j \alpha_j^N$, which is considered a free-parameter of the model. Since in

ad hoc retrieval, we do not have any information about the non-relevant class, we fix the parameters of the non-relevant class $\boldsymbol{\alpha}^N$ proportional to the general English proportions as shown below.

$$\boldsymbol{\alpha}^N = S\boldsymbol{\theta}^{GE}. \tag{5.10}$$

Although its a crude approximation in this case, in general our binary classifier allows us to model non-relevance when such information is explicitly available, whereas the language modeling framework does not model non-relevance at all.

The other major difference of ad hoc retrieval with text classification is the non availability of labeled training data. Hence instead of Maximum Likelihood training, we use the Expectation Maximization (EM) algorithm to learn the parameters of the relevance class. EM is a popular algorithm in machine learning that is used to learn the parameters of a mixture model from unlabeled examples in an iterative manner [9]. This algorithm starts by initializing the parameters of the mixture components to random values. The probabilities of class membership of the unlabeled examples are computed using the initial estimates of class parameters. These probabilities are used to re-estimate the parameters of the mixture components once again. This iterative process is repeated until some convergence criterion is met. It can be shown that the EM algorithm always increases the likelihood of the observed data with each iteration.

This algorithm is directly applicable to our model in the ad hoc retrieval scenario because firstly, our binary classifier is essentially a two component mixture model and secondly, there in no labeled data available in ad hoc retrieval except the query.

At the beginning of a retrieval session, the only information available about the topic of relevance is the user's query. Queries are different from documents in many respects. Queries are usually very concise while documents tend to be more verbose. Queries are usually focused on a specific topic while documents can digress from the main topic or discuss multiple topics. For this reason, relevance model treats documents and queries dif-

ferently. In RM, the estimation of the query's language model is not done directly from the query, but instead from the language models of the top ranking documents, while only conditioning them on the query. However, in our work, we assume that the query is just another labeled document available to the system for training purposes. The only distinction we make between queries and documents is the smoothing parameter we use in estimating their language models as described in section 5.3.4 below. This is clearly an oversimplification, which we resorted to, for modeling convenience. We will show in section 5.4 this simplification does not adversely affect the performance of the SD classifier in relation to the Relevance Model.

EM training using the query's language model as the only training example corresponds to maximum likelihood training which results in the following estimator.

$$\boldsymbol{\alpha}^R = S\boldsymbol{\theta}^Q = S\left(\lambda\hat{\boldsymbol{\theta}}^Q + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE}\right) \tag{5.11}$$

We then perform an initial retrieval using posterior probability of relevance $P(R|\boldsymbol{\theta}^D)$ which corresponds to the E-step of the EM algorithm. We have shown that the posterior probability w.r.t. the SD distribution is rank-equivalent to the difference in T-cross entropies between the non-relevant and relevant classes w.r.t. the document language models, as reproduced below.

$$P_{SD}(R|\boldsymbol{\theta}^D) \overset{rank}{=} H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D) \tag{5.12}$$

As in the Relevance Model, we re-estimate the parameters of the relevant class from the top ranking documents using the M-step of the EM algorithm which results in the following estimator.

$$\boldsymbol{\alpha}^R = \frac{S}{Z}\prod_{i=1}^{N}\left(\boldsymbol{\theta}^i\right)^{\frac{P(R|\boldsymbol{\theta}^i)}{\Sigma_{k=1}^{N} P(R|\boldsymbol{\theta}^k)}} \text{ where} \tag{5.13}$$

$$Z = \sum_{j=1}^{V}\prod_{i=1}^{N}\left(\theta_j^i\right)^{\frac{P(R|\boldsymbol{\theta}^i)}{\Sigma_{k=1}^{N} P(R|\boldsymbol{\theta}^k)}} \tag{5.14}$$

where the denominator Z is a normalization constant. In this case, the estimate of the relevance class is a geometric weighted average, where the weights correspond to the respective posterior probabilities of relevance. In contrast, the maximum likelihood estimate given a set of labeled documents is a simple geometric average as shown in (3.28). This observation tells us how to combine labeled and unlabeled data: when the document is not labeled, we estimate the posterior probability $P(R|\boldsymbol{\theta}^D)$ and when it is explicitly judged relevant by the user (true relevance-feedback), $P(R|\boldsymbol{\theta}^i)$ can be simply plugged in as unity. We will discuss this scenario in chapter 6 in more detail.

5.2.1 Approximations

As shown in (5.13), the weight $W_{SD}(D)$ assigned by the SD classifier to a top ranking document D is given by:

$$W_{SD}(D) = \frac{P(R|\boldsymbol{\theta}^D)}{\sum_{i=1}^{N} P(R|\boldsymbol{\theta}^i)} \text{ where} \tag{5.15}$$

$$P(R|\boldsymbol{\theta}^D) = \frac{\frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)\pi_R}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^N)(1-\pi_R)}}{\frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)\pi_R}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^N)(1-\pi_R)}+1} \tag{5.16}$$

where π_R is the prior probability of relevance. The posterior probability is a monotonic function of log-likelihood ratio as shown in (5.16). Let us first examine the log-likelihood ratio in more detail as shown below.

$$
\begin{aligned}
\frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^N)} &= \frac{\frac{S^S}{\prod_j (\alpha_j^R)^{\alpha_j^R}} \prod_j (\theta_j^D)^{\alpha_j^R - 1}}{\frac{S^S}{\prod_j (\alpha_j^N)^{\alpha_j^N}} \prod_j (\theta_j^D)^{\alpha_j^N - 1}} \text{ or} && (5.17) \\
\log \frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^N)} &= H(\boldsymbol{\alpha}^R) - H(\boldsymbol{\alpha}^N) + \sum_{j=1}^{V} (\alpha_j^R - \alpha_j^N) \log \theta_j^D && (5.18) \\
&= H(\boldsymbol{\alpha}^R) - H(\boldsymbol{\alpha}^N) + \sum_{j:w_j \in D} (\alpha_j^R - \alpha_j^N) \log \left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1 \right) \\
&\quad + \sum_{j=1}^{V} (\alpha_j^R - \alpha_j^N) \log \hat{\theta}_j^{GE} && (5.19) \\
&= H(\boldsymbol{\alpha}^R) - H(\boldsymbol{\alpha}^N) + \sum_{j:w_j \in D} (\alpha_j^R - \alpha_j^N) \log \left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1 \right) \\
&\quad + H(\boldsymbol{\alpha}^N, \hat{\boldsymbol{\theta}}^{GE}) - H(\boldsymbol{\alpha}^R, \hat{\boldsymbol{\theta}}^{GE}) && (5.20) \\
&= KL(\boldsymbol{\alpha}^N, \hat{\boldsymbol{\theta}}^{GE}) - KL(\boldsymbol{\alpha}^R, \hat{\boldsymbol{\theta}}^{GE}) + \\
&\quad \sum_{j:w_j \in D} (\alpha_j^R - \alpha_j^N) \log \left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1 \right) && (5.21) \\
&= S \log S - KL(\boldsymbol{\alpha}^R, \hat{\boldsymbol{\theta}}^{GE}) + \\
&\quad \sum_{j:w_j \in D} (\alpha_j^R - \alpha_j^N) \log \left(\frac{\lambda \hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1 \right) && (5.22)
\end{aligned}
$$

where step (5.19) follows from (5.18) using simplifications shown in (4.29). Step (5.22) uses (5.10) in the term $KL(\boldsymbol{\alpha}^N, \hat{\boldsymbol{\theta}}^{GE})$. Since $\boldsymbol{\alpha}^N$ is approximated to be proportional to the general English distribution $\hat{\boldsymbol{\theta}}^{GE}$, the KL-divergence term $KL(\boldsymbol{\alpha}^N, \hat{\boldsymbol{\theta}}^{GE}) = S \log S$ is very small and is equal to zero in the special case of $S = 1$. On the other hand, $KL(\boldsymbol{\alpha}^R, \hat{\boldsymbol{\theta}}^{GE})$ tends to be large because any particular topic is usually much different from the general English distribution. In practice, we have noticed that this term dominates the last term in (5.22) too and as a consequence, the log ratio of likelihoods on the LHS of (5.18) tends to be a large negative number. Thus, its exponent, the ratio of likelihoods on the LHS of (5.17) is always a small number much less than one. The prior ratio $\pi_R/(1-\pi_R)$ is also a small number because we expect relevant documents to be much smaller in number than the non-relevant ones. Using these two observations and the result $\lim_{x \to 0} \frac{x}{x+1} = x$, we can

approximate the posterior probability of relevance in (5.16) as follows.

$$\begin{aligned}
\log P(R|\boldsymbol{\theta}^D) &\approx \log \frac{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^R)\pi_R}{P(\boldsymbol{\theta}^D|\boldsymbol{\alpha}^N)(1-\pi_R)} && (5.23)\\
&= S\log S - KL(\boldsymbol{\alpha}^R, \hat{\boldsymbol{\theta}}^{GE}) \\
&\quad + \sum_{j:w_j\in D} (\alpha_j^R - \alpha_j^N)\log\left(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}}+1\right) + \log\frac{\pi_R}{1-\pi_R} && (5.24)
\end{aligned}$$

where we substituted (5.22) in (5.23) to obtain (5.24). Using this result, the weight assigned to the documents by the SD model shown in (5.15) can be approximated to the following.

$$\begin{aligned}
W_{SD}(D) &= \frac{P(R|\boldsymbol{\theta}^D)}{\sum_{i=1}^N P(R|\boldsymbol{\theta}^i)} && (5.25)\\
&\approx \frac{1}{Z}\exp\left\{\sum_{j:w_j\in D}(\alpha_j^R - \alpha_j^N)\log\left(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}}+1\right)\right\} && (5.26)\\
&= \frac{1}{Z}\exp\left\{\sum_{j:w_j\in D} S\lambda(\hat{\theta}_j^Q - \hat{\theta}_j^{GE})\log\left(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}}+1\right)\right\} && (5.27)
\end{aligned}$$

where the document independent terms in the posterior probability shown in (5.24) cancel out in the normalization of (5.25) and Z is a normalizer that sums up the numerator term over all the top ranking documents. We obtained (5.27) by substituting (5.10) and (5.11) in (5.26).

Table 5.1 presents a comparative summary of the above discussion on estimation and inference formulae of the RM and the SD based generative classifier. Notice the striking similarity of the weight of the RM in (5.8) and that of the SD classifier in (5.27). It should not be surprising because both the weights are based on the T-cross entropy function. SD has an additional term for the negative class since it uses a binary classification perspective. It turns out that the weighting scheme of the RM is not only highly effective in terms of performance but also highly impervious to noise. Comparing the RM weight to the term in

	Condition	RM	SD
1	Initial estimate	Q or $\hat{\theta}^Q$	$\boldsymbol{\alpha}^R = S\boldsymbol{\theta}^Q; \boldsymbol{\alpha}^N = S\hat{\boldsymbol{\theta}}^{GE}$
2	Initial Ranking	$P(Q\|\boldsymbol{\theta}^D) \overset{rank}{=} -H(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^D)$	$H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D)$
3	Document Weight (W)	$\frac{1}{Z}\exp\{\sum_{j:w_j \in Q,D} f_j^Q \log(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1)\}$	$\frac{1}{Z}\exp\{\sum_{j:w_j \in D} S\lambda(\hat{\theta}_j^Q - \hat{\theta}_j^{GE}) \log(\frac{\lambda\hat{\theta}_j^D}{(1-\lambda)\hat{\theta}_j^{GE}} + 1)\}$
4	Final estimate	$\boldsymbol{\theta}^R = \frac{1}{Z}\sum_{i=1}^N \boldsymbol{\theta}^i W(i)$	$\boldsymbol{\alpha}^R = \frac{1}{Z}\prod_{i=1}^N (\boldsymbol{\theta}^i)^{W(i)}$
5	Final Ranking	$-H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^D)$	$H(\boldsymbol{\alpha}^N, \boldsymbol{\theta}^D) - H(\boldsymbol{\alpha}^R, \boldsymbol{\theta}^D)$

Table 5.1. Comparison of RM and SD

SD weight corresponding to the relevant class, it is clear that they are equivalent when the following condition holds.

$$S\lambda = |Q| \tag{5.28}$$

In other words, if the SD based classifier were to assign weights to documents that are equivalent to the RM weights, then the precision of the SD distribution S has to be made proportional to the length of the query. We know that the variance of the Dirichlet distribution (and consequently the SD distribution) is inversely related to its precision as shown below.

$$\text{Var}[\theta_j|\boldsymbol{\alpha}] = \frac{\alpha_j(S-\alpha_j)}{S^2(S+1)} \tag{5.29}$$

This behavior of the Dirichlet distribution is illustrated in figure 5.1 which plots a two dimensional Dirichlet for various values of S with $\boldsymbol{\alpha} = (S/2, S/2)$. Notice that the distribution gets more peaky (less variance) as the value of S increases, but the maximum probability is always centered about the mean $(0.5, 0.5)$. Hence, imposing proportionality of the Dirichlet precision with query length would mean that the distribution has low variance for long queries. In other words, the distribution of weights assigned to documents by the model would be very peaked for long queries and relatively evenly distributed for shorter

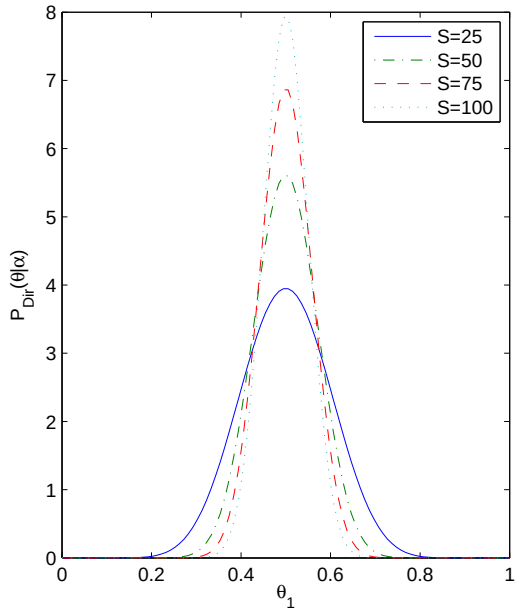

Figure 5.1. Dirichlet distribution has lower variance for higher values of precision S

queries. This is intuitively very meaningful since longer queries contain more information and one would tend to have high confidence in documents that are nearest to the query. When the query is short, there is less information and hence one would rather distribute the weight more evenly among the top ranking documents. Thus, the query-likelihood formula offers a flexible approach of adjusting the weight distribution based on the length of the query.

Despite the similarity of the RM and the SD model, the consistent generative framework of the latter allowed us to interpret the query-likelihood based weight distribution of the RM in terms of the variance of the SD distribution.

Collection	$\|\mathcal{C}\|$	$\|D\|_{Avg}$	$\|V\|$	Query set
AP (88-90)	242,918	257	245,746	51-150
WSJ (87-92)	173,252	258	174,736	1-200
LAT	131,896	269	187,263	301-400
FT (91-94)	210,158	223	223,299	251-400

Table 5.2. Data Collections

5.3 Experiments

5.3.1 Models considered

In this section, we compare the performance of three cross entropy based techniques, namely simple query-likelihood ranking that corresponds to $H(\hat{\boldsymbol{\theta}}^Q, \boldsymbol{\theta}^D)$, the Relevance model and the SD based generative classifier. We also used an untrained version of the TF-IDF model [50, 49] with pseudo relevance feedback as a baseline in our experiments. the query-likelihood model, as the name indicates, is based only on the query and hence does not model pseudo relevance feedback. For the remaining models, we performed pseudo relevance feedback of top 100 documents from the initial retrieval.

5.3.2 Data sets

We used standard TREC[1] collections and queries in our experiments, the details of which are presented in table 5.2. We performed stopping and stemming and indexed each collection using the *Lemur*[2] toolkit. We used title version of the queries 51-100 on the AP collection as our training queries. We tested our models on title queries 101-150 on the AP corpus and on the entire title query sets on the remaining collections. Since all the collections are of similar sizes, one can expect models trained on one collection to perform more or less optimally on the other collections.

[1] http://trec.nist.gov

[2] http://www.lemurproject.org

5.3.3 Evaluation measures

We first define $Pr(r)$, precision at rank r, as follows.

$$Pr(r) = \frac{N_{rel}(r)}{r} \tag{5.30}$$

where $N_{rel}(r)$ is the number of relevant documents found in the ranked list up to rank r. Now *AvgP*, the average precision is defined as follows.

$$\text{AvgP} = \frac{\sum_{r=1}^{N} Pr(r)\delta(r)}{N} \tag{5.31}$$

where $\delta(r)$ is a binary function that takes a value of unity if the document at rank r is relevant and zero otherwise and N is the number of documents retrieved by the system. Thus, *AvgP* is the average of the precision after each relevant document is retrieved. This method emphasizes returning more relevant documents earlier in the ranked list than later.

Now we define MAP, the mean average precision as the mean of the average precision over all the queries under consideration. We used MAP as our primary evaluation metric.

We also report $Pr(5)$, precision at rank 5 as a secondary evaluation measure. This measure is considered important particularly in the web context where the quality of the retrieval results on the first page are of critical value. We however trained the models by optimizing only MAP on the training set of queries.

In addition, we also use plots of precision vs. recall, averaged over several queries, to illustrate the distribution of the relevant documents in the ranked lists. Recall at rank r, denoted by $Re(r)$ is defined as

$$Re(r) = \frac{N_{rel}(r)}{N_{rel}} \tag{5.32}$$

where N_{rel} is the total number of relevant documents. In general the farther these plots are away from the origin, the better is considered the model's performance.

We also performed statistical significance tests using the standard paired T-test, the Wilcoxon test and the sign test, all at 95% confidence level. We used the Wilcoxon as an additional test as it is more commonly used in IR experiments [23].

5.3.4 Training the models

Model training consists of optimizing the free parameters of the models on the training set of queries. We furnish details of the free parameters of each of the models considered below.

The query-likelihood (QL) model shown in (1.4) consists of only one free parameter, namely, the smoothing parameter. For this model, we chose to use Dirichlet smoothing for documents as shown below in (5.33), since it is known to give better performance than Jelinek Mercer smoothing [60] for short queries.

$$\boldsymbol{\theta}^D = \frac{|D|}{\mu + |D|}\hat{\boldsymbol{\theta}}^D + \frac{\mu}{\mu + |D|}\hat{\boldsymbol{\theta}}^{GE} \tag{5.33}$$

where $\mu > 0$ is the smoothing parameter that is to be optimized.

The relevance model has three free parameters as enumerated below.

1. smoothing parameter λ_Q for documents in the initial query likelihood ranking shown in step 2 of table 5.1, which also corresponds to the λ in computation of document specific weights in step 3.

2. smoothing parameter λ_M used to smooth documents during the computation of $\boldsymbol{\theta}^R$ in step 4 of table 5.1

3. smoothing parameter λ_{CE} to used smooth documents used in final T-cross-entropy ranking shown in step 5 in the same table.

One could also consider N, the number of top ranking documents for pseudo relevance feedback, as a free parameter. In our experiments, we fixed this value at 100 for all the models.

Note that although a single smoothing parameter can be used in all these steps, it has been empirically found that using different smoothing parameters during different steps improves performance significantly. Following the example of relevance model, we define the following free parameters in the generative SD based classifier:

1. Smoothing parameter λ_{M1} used to smooth the query's MLE distribution in the estimation of $\boldsymbol{\alpha}^R$ in step 2 of table (5.1). The subscript $_{M1}$ denotes that this parameter is used in the first M-step of the EM algorithm.

2. Smoothing parameter λ_{E1} to smooth documents used in ranking as shown in step 3 in the table. This also corresponds to the 1st E-step as indicated by the subscript.

3. Smoothing parameter λ_{M2} to smooth documents in the 2nd M-step shown in step 4 in the table.

4. Smoothing Parameter λ_{E2} to smooth documents in the final ranking in step 4, that corresponds to the second E-step as denoted by the subscript.

5. Following (5.28), we define precision of the SD distribution as $S = K \times |Q|$ and consider K as an additional free parameter.

5.4 Results and Discussion

The results of our experiments are presented in table 5.3. Both RM and SD outperform the QL model on all the collections. This is not surprising because QL model is based only on the query terms while both RM and SD expand the query using top ranking documents from an initial retrieval. Although TFIDF model models pseudo feedback, it outperforms QL only on the AP corpus. We believe this is mainly because we did not tune the parameters of the TFIDF model. Since we are interested in models that use the T-cross entropy ranking function, namely QL, RM and SD, we included the TFIDF model only as a low baseline.

		QL	RM	SD	TFIDF
TRAINING					
Parameters		μ	$\lambda_Q, \lambda_M, \lambda_{CE}$	$\lambda_{M1}, \lambda_{E1}, \lambda_{M2}, \lambda_{E2}, K$	Default
opt. values		900	0.7,0.6,0.1	0.99,0.6,1e-4,0.8,0.8	
AP (Q:51-100)	MAP	25.30	30.31	30.61	26.75
	Pr(5)	48.09	53.19	**58.30**	51.49
TESTING					
AP (Q:101-150)	MAP	19.72	28.82	28.49	26.06
	Pr(5)	42.80	47.20	50.40	50.80
WSJ (Q:1-200)	MAP	25.78	29.65	29.19	23.47
	Pr(5)	44.85	47.47	**49.09**	43.23
LAT (Q:301-400	MAP	22.77	24.77	24.73	17.69
	Pr(5)	31.72	30.71	31.52	23.03
FT (Q:251-400)	MAP	21.22	21.66	21.28	17.96
	Pr(5)	30.68	27.16	27.16	18.77

Table 5.3. Performance comparison of various models on 4 different TREC collections: the values of Mean Average Precision (MAP) and Precision at 5 documents (Pr(5)) are in percentage. Significance results are reported only between RM and SD in the table. Bold faced numbers indicate statistical significance w.r.t the Wilcoxon test as well as a paired two tailed T-test at 95% confidence level. Note that the sign test did not indicate any significant differences between the two models on any of the runs.

The results on mean average precision show little difference between RM and SD.We also performed statistical significance tests between RM and SD using a paired two tailed T-test, the sign-test as well as the Wilcoxon test, both at 95% confidence level. As shown in the table, on all the runs, RM and SD are found to be statistically indistinguishable. The identical behavior of SD and RM is also demonstrated in the precision-recall plots on various data sets in figures 5.2 through 5.6. The curves corresponding to SD and RM coincide at almost all levels of precision and recall in all the plots. This is again not surprising because both the models are very similar to each other as discussed in section 5.2.1. Note that although SD is slightly better than RM on the training set in terms of average precision, it is marginally lower than RM on the test queries. We believe this is a result of over-fitting due to the higher number of free parameters in SD than RM.

Although SD and RM are comparable in terms of mean average precision, note that SD performs consistently better than RM in terms of precision at top 5 documents except on FT corpus where their performance is tied. In case of the training set of queries on the AP corpus as well as the test queries on WSJ corpus, the difference is found to be statistically significant w.r.t. the T-test and the Wilcoxon test (but not the sign test) at 95% confidence level. Thus SD could be useful in situations where high precision is required.

We believe the performance of SD model can be further improved in the future since it allows us to model the non-relevance class unlike the RM which models only the relevance class. In this work, we trivially assumed the non-relevant class parameters to be proportional to the general English parameters. If the user provides negative feedback, this class can be modeled more accurately which may result in better retrieval.

The most important lesson from the results on ad hoc retrieval we presented in this chapter is that generative likelihood based models (classifiers in this case) are suitable for ad hoc retrieval task as long as the underlying distribution chosen is appropriate for text. Previous generative classifiers such as the BIR model [45] and the multinomial based naïve Bayes [51] failed in the task of ad hoc retrieval mainly due the usage of incorrect distributions for

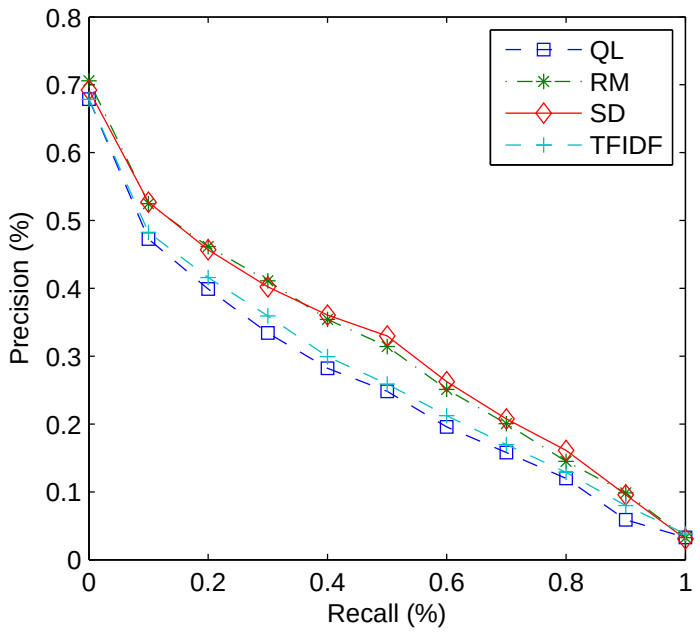

Figure 5.2. Comparison of precision recall curves on AP training queries

text. We hope our work will encourage researchers to build more sophisticated machine learning models for ad hoc retrieval.

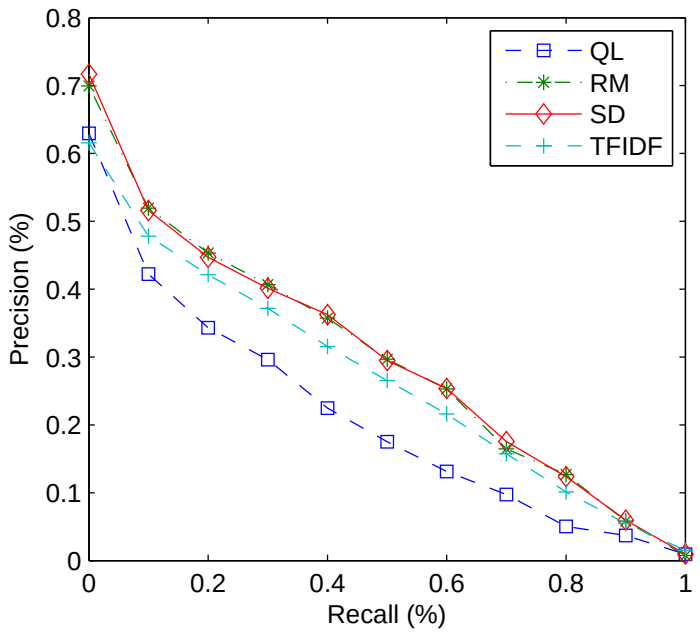

Figure 5.3. Comparison of precision recall curves on AP test queries

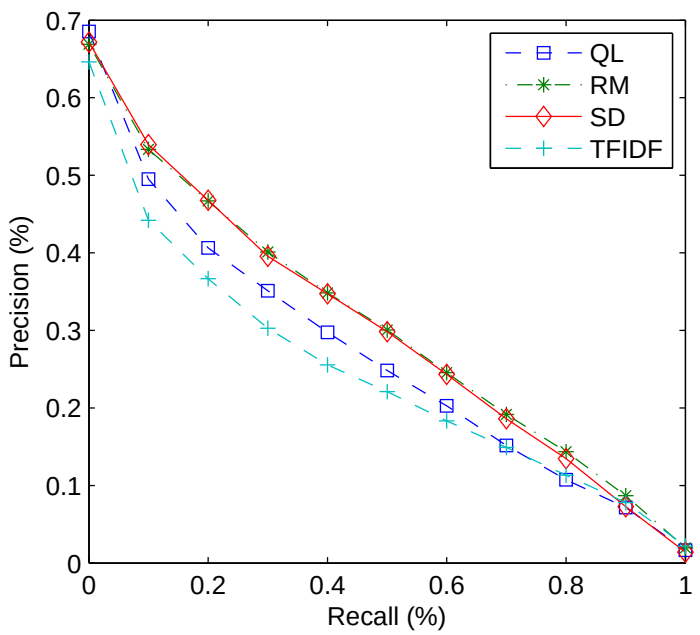

Figure 5.4. Comparison of precision recall curves on WSJ queries

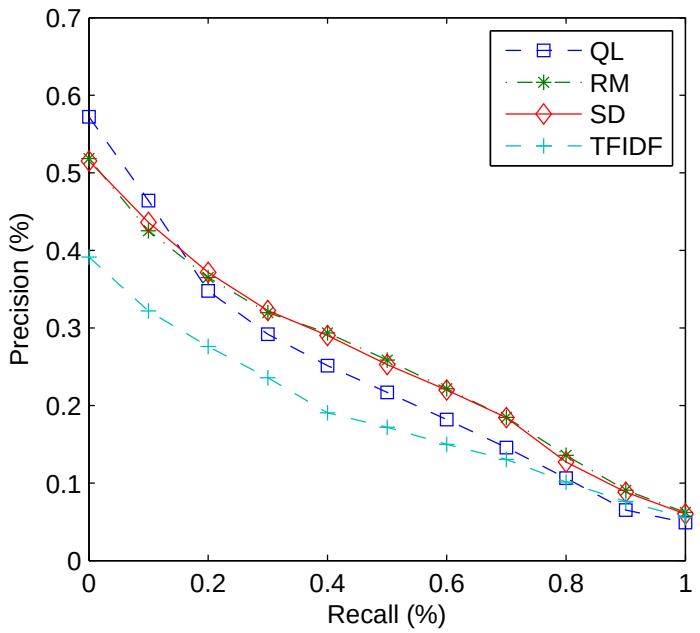

Figure 5.5. Comparison of precision recall curves on LAT queries

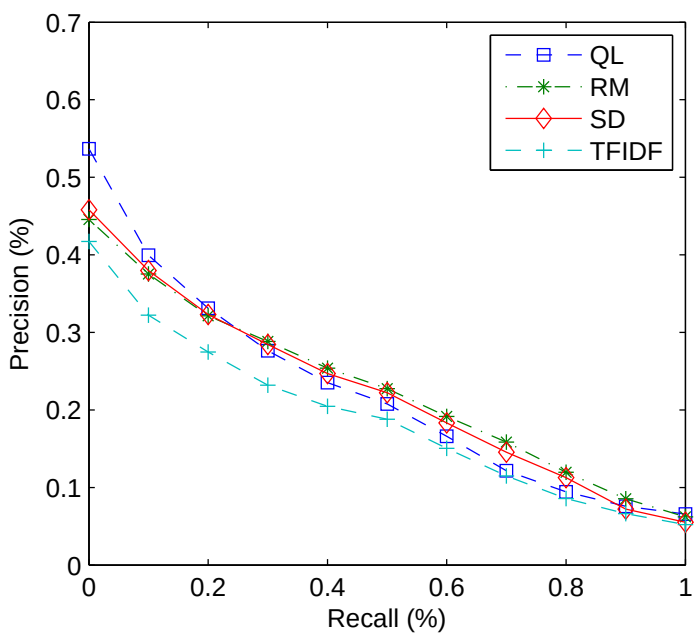

Figure 5.6. Comparison of precision recall curves on FT queries

CHAPTER 6

TOPIC TRACKING: ONLINE CLASSIFICATION

In this chapter, we consider the task of topic tracking, an online task of the Topic Detection and Tracking (TDT) research program[1]. In this task, each topic is provided with an initial seed training document D_0 (corresponding to the condition $N_t = 1$ in the official evaluation [11]). A stream of documents is provided for each topic and the task is to identify all the documents in the stream that belong to this topic in an online fashion. Unlike the TREC task of filtering [42], there is no user feedback available in this task. Owing to the evolving nature of news topics, it has been empirically observed that the system needs to adapt to the incoming stories in order to achieve optimal performance, although no human feedback is available. A typical TDT tracking system assigns a confidence score $f(D)$ to each incoming document D and also sets two thresholds, namely the decision threshold T_d and adaptation threshold $T_a > T_d$. The document is marked as relevant ("on topic") if $f(D) \geq T_d$ and the system adapts the model using D if $f(D) \geq T_a$. The adaptation threshold is fixed at a higher value than the decision threshold in order to ensure that the system adapts to only documents that are highly relevant. This ensures that the model does not deviate too much from the main topic of discussion.

In this work, we simplify the traditional tracking problem by ignoring the issue of adaptation threshold. Instead, we mark certain documents and ask the system to adapt itself using these documents. Note that for these documents, we provide neither the scores assigned by any model nor their labels, hence it is considered unsupervised learning. In terms of the IR parlance, this scenario is related to pseudo relevance feedback. One of the

[1] http://www.nist.gov/TDT/

reasons for this simplification is the fact that our main interest is only to compare the ability of various models to adapt automatically to evolving news and identify relevant documents. Modeling the adaptation threshold is in itself an active area of research [43, 57, 62], but is an independent problem that can be considered separately. Secondly, using an adaptation threshold will add an extra dimension to the problem and one will not be able to infer whether a model's superior performance is due to its superior ability to model text or simply a result of better thresholding. We used this simplification which guarantees that all models receive the same information to adapt from. Hence, one can state with high degree of confidence that superior performance of any model is a result of its better text modeling.

6.1 Data sets

We did our experiments on the TDT-4 corpus. The corpus consists of multi-lingual news stories from 8 English sources, 7 Mandarin sources and 5 Arabic sources and covers the period from October 1 2001 to January 31 2002. An extra feature of the TDT corpora is that they consist of data from multimedia sources such as audio and video apart from regular newswire. When the data is audio and video, transcripts from an Automatic Speech Recognition system as well as manual transcriptions (closed caption quality) are available. Also, when the news source is non-English, output of a Machine Translation system to English is made available. We used only native English documents as the initial seed documents for all topics. Where applicable, we used manually transcribed data and machine translation output in our experiments. We used 33 topics from the TDT 2002 evaluation that contained at least 5 relevant documents in their respective streams as training topics. A similar criterion yielded 29 topics from the TDT 2003 evaluation which we used for testing. Note that both the topics are defined on the same TDT-4 collection. Unlike ad hoc retrieval, in the tracking task, the entire collection is not available to us at testing time. Hence, to compute general English statistics required for most of our models, we used data from the TDT-3 corpus. The statistics of both TDT-3 and TDT-4 corpora are presented in table 6.1.

	TDT3	TDT4
$\|\mathcal{C}\|$	101,765	98,245
$\|D\|_{avg}$	153	201
$\|V\|$	996,839	135,305
Coverage time	Oct. 1998 - Dec. 1998	Oct. 2001 to Jan. 2002
Sources	8 English 3 Mandarin	8 English 7 Mandarin 5 Arabic
Foreign languages Audio/Video source	Machine translated Manually transcribed	Machine translated Manually transcribed

Table 6.1. Statistics TDT data

Preprocessing the corpora consisted of stop-word removal and stemming. Although we had used the Porter stemmer in our previous experiments, we stemmed the TDT corpora using the K-stemmer. It has been shown that this choice makes no difference from a performance standpoint [20]. We indexed the collection using the *Lemur* toolkit version 3.0.

6.2 Evaluation

All of the TDT tasks are cast as detection tasks. Detection performance is characterized in terms of the probability of miss and false alarm errors (P_{Miss} and P_{FA}). These error probabilities are then combined into a single detection cost, C_{track}, by assigning costs to miss and false alarm errors:

$$C_{Track} = P_{target} P_{Miss} C_{Miss} + (1 - P_{target}) P_{FA} C_{FA} \tag{6.1}$$

where P_{target} is the prior probability of a relevant document and C_{Miss} and C_{FA} are the costs of miss and false alarm respectively. C_{Track} is normalized so that it is no less than one for trivial algorithms that do not use information in the documents [11].

For the evaluation of topic tracking, each topic was evaluated separately. Results were then combined for all topics either by pooling all trials (story-weighted) or by weighting

the trials so that each topic contributed equally to the result (topic-weighted). In this work, we used topic-weighted averaging. For cost-based evaluation of topic tracking, 0.02 was assigned as the *a priori* probability P_{target} of a story discussing a target topic. The ratio C_{Miss}/C_{FA} is fixed at 10.

The values of P_{Miss} and P_{FA} are functions of the value of the decision threshold T_d. As examples, when $T_d = \infty$, all the documents are considered non-relevant by the system, hence $P_{Miss} = 1$. On the other hand, if $T_a = -\infty$, all documents are marked relevant, hence $P_{FA} = 1$ in this case. One can plot P_{FA} and P_{Miss} against each other smoothly varying the value of T_d. A curve thus obtained is called the Detection Error Trade-off (DET) curve. One can also define a point on the curve where C_{Det} is minimum. We call the value of the normalized cost at this point *MinCost* and we will use this value to evaluate our models. This evaluation eliminates the need to define the threshold T_d for our models since we measure the value of the cost at the optimum threshold. Hence all that matters in this evaluation is the relative scores of documents with respect to one another. Thus when *MinCost* is used as the evaluation metric, rank equivalence relationship of scoring functions is applicable. It is important to remember that unlike the evaluation metrics used in ad hoc retrieval and text classification, low values of *MinCost* mean better performance.

We also performed statistical significance tests between any two models as follows. For each model, using the decision threshold T_d corresponding to the *MinCost* over all test topics, we computed the costs for each topic at that threshold. We compared these topic-specific costs pairwise between two models over all topics to measure statistical significance using a paired T-test as well as a signed test.

6.3 Models considered

We considered the naïve Bayes classifier, the SD based classifier and the Relevance Model for comparison. For the former two classifiers, we considered this problem as a binary classification problem of relevance and non-relevance as described in chapter 5 and

used the EM algorithm to learn from and score the unlabeled documents. We also used the traditional vector space model with Rocchio style adaptation [48] as an additional baseline.

We also ran a non-adaptive version of the vector space model that trains only on the initial seed document. The top ranking documents from this run are used for pseudo relevance feedback for all the models.

Since tracking is an online task, we defined online versions of the EM algorithm for both the the naïve Bayes and SD classifiers. The Relevance model also needs a few approximations owing to the fact that the training document, unlike the query in ad hoc retrieval is one of the observed examples. We describe these details in the following subsections.

6.3.1 Vector Space model

In the traditional vector space model [49], each document D is considered a vector $\mathbf{v}(D)$ in term-space where each component value is equal to the TF-IDF weight defined as follows.

$$v_j(D) = f_j^D \log(\frac{|\mathcal{C}| + 1}{N_j + 0.5}) \tag{6.2}$$

where $|\mathcal{C}|$ is the number of documents in the background corpus and N_j is the number of documents in the background corpus that the term occurs in. The topic model is also a vector that is initialized to the vector of the training document as in $\mathbf{v}_0^R = \mathbf{v}(D_0)$. The score of a document D_i in the stream is given by the cosine of the angle between the model vector $\mathbf{v}^R$ and the document vector $\mathbf{v}(D_i)$ as follows.

$$\cos(\mathbf{v}^R, \mathbf{v}(D_i)) = \frac{\mathbf{v}^R \cdot \mathbf{v}(D_i)}{||\mathbf{v}^R|| \times ||\mathbf{v}(D_i)||} \tag{6.3}$$

When the tth pseudo feedback document D_t arrives in the stream, the model vector is updated using a Rocchio style adaptation [48] as follows.

$$\mathbf{v}_t^R = \frac{1}{t+1} t \mathbf{v}_{t-1}^R + \mathbf{v}(D_t) \tag{6.4}$$

This basic version of the model consists of no free parameters and hence requires no training. We note that there are more advanced models that weight the pseudo feedback documents adaptively [7], use document comparison in native languages [2] or use document expansion to model topics [28] *etc.* Since our main interest lies in comparing the RM and the naïve Bayes model with the SD based classifier, we used only a simple version of the vector space model as a baseline.

6.3.2 Relevance model

As shown in (5.1), estimation in Relevance model involves computing a conditional expectation given the training document D_0 as reproduced below.

$$\boldsymbol{\theta}^R = E[\boldsymbol{\theta}|D_0] = \int_{\boldsymbol{\theta}} \boldsymbol{\theta} P(\boldsymbol{\theta}|D_0) d\boldsymbol{\theta} \tag{6.5}$$

$$\approx \frac{\sum_i \boldsymbol{\theta}^i P(D_0|\boldsymbol{\theta}^i)}{\sum_i P(D_0|\boldsymbol{\theta}^i)} \tag{6.6}$$

The summation in (6.6) is over all the observed documents. In the case of tracking, since the training document D_0 is one of the observed documents, it is included in the summation. As a result, the weight distribution tends to be a highly skewed and centered around $\boldsymbol{\theta}^0$, since the probability of the observed document D_0 w.r.t. to its own language model $\boldsymbol{\theta}^0$ is usually much higher than w.r.t. other language models. Thus the estimate of the Relevance model collapses to the language model of the training document D_0 as shown below.

$$\boldsymbol{\theta}_0^R = \frac{1}{Z} \sum_i \boldsymbol{\theta}^i P(D_0|\boldsymbol{\theta}^i) = \boldsymbol{\theta}^0 \tag{6.7}$$

As a result, Relevance Model fails to learn from the unlabeled documents in this scenario. To remedy this situation, we assume the pseudo feedback documents as true feedback documents. This allows us to consider each of those documents D_t as evidence and a build a

relevance model w.r.t. that document. An average of these models is then computed which is treated as the relevance model. More formally, if $\{D_1, \cdots, D_N\}$ are the pseudo feedback documents, then the relevance model $\boldsymbol{\theta}^R$ is computed as follows.

$$\begin{aligned}
\boldsymbol{\theta}^R &= \frac{1}{N+1}\sum_{t=0}^{N} E[\boldsymbol{\theta}|D_t] && (6.8)\\
&\approx \frac{1}{N+1}\sum_{t=0}^{N} \frac{\sum_{k=0}^{N}\boldsymbol{\theta}^k P(D_t|\boldsymbol{\theta}^k)}{\sum_{l=0}^{N} P(D_t|\boldsymbol{\theta}^l)} && (6.9)\\
&\approx \frac{1}{N+1}\sum_{t=0}^{N} \boldsymbol{\theta}^t && (6.10)
\end{aligned}$$

In an online situation, the relevance model $\boldsymbol{\theta}^R$ is initialized to the language model of the training document $\boldsymbol{\theta}^0$. As the tth pseudo feedback document D_t arrives in the document stream, the relevance model in (6.10) can be computed as a moving average as shown below.

$$(\boldsymbol{\theta}^R)_t = \frac{1}{t+1}\left\{t\boldsymbol{\theta}^R_{t-1} + \boldsymbol{\theta}^t\right\} \tag{6.11}$$

Scoring of any document D_i in the stream is done by using the T-cross entropy function as usual.

$$\text{score}(D_i) = -H(\boldsymbol{\theta}^R, \boldsymbol{\theta}^i) \tag{6.12}$$

The free parameters of the Relevance model consist of the smoothing parameter λ_M used for document language models during estimation in (6.11) and λ_E used in computing the document scores in (6.12) which we optimize during training.

6.3.3 Naïve Bayes classifier

The naïve Bayes classifier for tracking views the problem as a binary classification problem of relevance. The relevance class distribution $\boldsymbol{\theta}^R$ is initialized to the smoothed

language model of the seed document $\boldsymbol{\theta}^0$ and the non-relevant class distribution $\boldsymbol{\theta}^N$ is fixed to the general English distribution estimated from the TDT3 corpus $\hat{\boldsymbol{\theta}}^{GE}$. Note that we estimate only the relevant class distribution during the online phase while we keep the non-relevant distribution fixed. Given the two class parameters, the score of a document D_i in the stream is given by the E-step, which is equivalent to computing its posterior probability of relevance $P(R|D_i)$. We have shown this to be proportional to the D-cross entropy function in (2.18). We reproduce the exact expression for posterior probability below.

$$P(R|D_i) = \frac{L(D_i)}{L(D_i)+1} \text{ where} \tag{6.13}$$

$$\log L(D_i) = \log \frac{\pi_R}{1-\pi_R} + \sum_{j=1}^{V} f_j^i \log \frac{\theta_j^R}{\theta_j^N} \tag{6.14}$$

Since our evaluation only considers the ranking of documents, one can simply score the documents using the expression in (2.18). However, in the estimation step, we need the full probability as explained below. Given N pseudo feedback documents $\{D_1 \cdots, D_N\}$, the estimate for $\boldsymbol{\theta}^R$ is given by the M-step of the EM algorithm as follows.

$$\hat{\boldsymbol{\theta}}^R = \frac{\sum_{i=0}^{N} P(R|D_i)\mathbf{f}^i}{\sum_{i=0}^{N} P(R|D_i)|D_i|} \tag{6.15}$$

where we use $P(R|D_0) = 1$ since the initial seed document is known to be relevant. This estimate $\hat{\boldsymbol{\theta}}^R$ however needs to be smoothed to avoid zero probabilities as shown below.

$$\boldsymbol{\theta}^R = \lambda\hat{\boldsymbol{\theta}}^R + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE} \tag{6.16}$$

In an online setting, given the tth pseudo feedback document, one can express the estimate for the relevant distribution $\boldsymbol{\theta}_t^R$ in terms of the previous estimate $\boldsymbol{\theta}_{t-1}^R$ as follows.

$$\hat{\boldsymbol{\theta}}_t^R = \frac{\hat{\boldsymbol{\theta}}_{t-1}^R \sigma_{t-1} + P(R|D_t)\mathbf{f}^t}{\sigma_{t-1} + P(R|D_t)|D_t|} \text{ where} \tag{6.17}$$

$$\sigma_{t-1} = \sum_{i=0}^{t-1} P(R|D_i)|D_i| \tag{6.18}$$

Again, smoothing shown in (6.16) is applied after every M-step.

The free parameters of this model consist of the smoothing parameter λ shown in (6.16) and the prior ratio $\pi_R/(1-\pi_R)$ shown in (6.14).

6.3.4 SD based classifier

Similar to the naïve Bayes classifier, we treat the training document D_0 as a labeled document and the pseudo feedback documents as unlabeled documents and we apply the EM algorithm as done in the case of ad hoc retrieval. We initialize the parameters as $\boldsymbol{\alpha}^R = S\boldsymbol{\theta}^0$ and $\boldsymbol{\alpha}^N = S\hat{\boldsymbol{\theta}}^{GE}$. The score of a document D_i is given by the posterior probability of relevance which we know is proportional to the difference in T-cross entropies w.r.t. the relevant and non-relevant class parameters as shown in (3.41).

As discussed in section 5.2, one should ideally assign a weight of $P(R|\boldsymbol{\theta}^0) = 1$ to the training document D_0 in the M-step. However since we make approximations to the posterior probability for unlabeled documents as shown in section (5.2.1), we can no longer plug in a value of unity for the training document D_0. Instead, following the intuition that the labeled document is more important than any one of the pseudo feedback documents, we define the training document weight as follows:

$$W(D_0) = W_0(\max_{i=1}^{N} W(D_i)) \tag{6.19}$$

where $W(D_i)$ is the weight of the unlabeled document D_i as defined in (5.27) and $W_0 \geq 1$ is a free parameter that ensures that the training document is weighted higher than the pseudo feedback documents. The initial value of W_0, when there is no pseudo relevance feedback is fixed at 1. In an online setting, since we do not have access to the entire set of

N pseudo feedback documents until the end of the stream, we update this weight online as follows.

$$W(D_0)_t = W_0(\max_{i=1}^{t} W(D_i)) \tag{6.20}$$

One can adapt the estimation formula in M-step shown in (5.16) to an online setting as follows.

$$\boldsymbol{\alpha}_t^R = \frac{1}{Z_t}\left(\left\{Z_{t-1}\boldsymbol{\alpha}_{t-1}^R\right\}^{\sigma_{t-1}}(\boldsymbol{\theta}^0)^{\Delta(t)}(\boldsymbol{\theta}^t)^{W(D_t)}\right)^{\frac{1}{\sigma_t}} \text{ where} \tag{6.21}$$

$$\Delta(t) = W(D_0)_t - W(D_0)_{t-1} \text{ and} \tag{6.22}$$

$$\sigma_t = \sigma_{t-1} + W(D_t) + \Delta(t) \tag{6.23}$$

The free parameters of the model include λ_E used to smooth documents in the E-step, λ_M used in document smoothing in M-step, W_0 and S.

6.4 Results

We ran all the algorithms for different settings of N, the number of pseudo feedback documents. For each these settings, we trained the free parameters of the model by optimizing the minimum cost on the training topics. Using the corresponding optimal settings of the free parameters, we ran the models on the test topics. The results of these experiments are shown in table 6.2. The results show that the naïve Bayes classifier performs better than the baseline TFIDF model for $N \leq 5$ but the performs deteriorates rapidly for larger values of N. The Relevance Model outperforms the naïve Bayes classifier once again consistently for all values of N. This once again demonstrates the superiority of the T-cross entropy ranking function. The SD based classifier outperforms the Relevance Model for most values of N. It achieves optimal performance at $N = 5$, where the performance improvement compared to the Relevance Model is the maximum at 13.35%. The performance

N	TFIDF	naïve Bayes		Relevance Model		Smoothed Dirichlet	
	MinCost	Opt Par $\lambda, \frac{\pi_R}{1-\pi_R}$	MinCost	Opt Par λ_E, λ_M	MinCost	Opt Par $\lambda_E, \lambda_M, S, W_0$	MinCost
0	0.2888	0.1,1e-3	0.2607	0.05,0.5	0.2378	0.05,0.99,1,1	**0.2373**
2	0.2705	0.2,1e-3	0.2385	0.05,0.8	**0.1903**	0.01,0.05,1,2	0.1974
5	0.2716	0.3,1e-6	0.2451	0.05,0.4	0.1939	1e-3,0.1,2,1	**0.1680**
25	0.3076	0.3,1e-7	0.3384	0.05,0.7	0.2024	5e-5,0.2,15,2	$\mathbf{0.1792}^*$
50	0.3094	0.5,1e-4	0.3635	0.05,0.3	0.2139	1e-5,0.2,20,5	$\mathbf{0.1883}^*$
100	0.3102	0.7,1e-8	0.4806	0.05,0.5	0.2240	1e-5,0.1,250,5	**0.1945**

Table 6.2. Comparison of performance at various levels of pseudo feedback: N is the number of pseudo feedback documents and bold faced indicates best performing model for the corresponding run. The superscript * statistical significance compared to the nearest model w.r.t. the paired one-tailed T-Test at 90% confidence interval. Note that the sign test at 95% confidence level did not indicate any significant differences.

N	Num. of topics
0	12
5	17
25	15
50	18
100	15

Table 6.3. Number of topics of the 29 test topics on which SD outperforms RM at various levels of pseudo feedback

improvement w.r.t. the Relevance Model is statistically significant at $N = 25$ and $N = 50$ as measured by a paired one tailed T-test at 90 % confidence interval, but not significant w.r.t. the sign test.

Although the differences between RM and SD are not statistically significant, SD is shown to be consistently better than RM. We illustrate this using table 6.3 which presents the number of topics on which SD outperforms RM on the 29 test topics for various levels of pseudo feedback. The table shows that except when there is no feedback ($N = 0$), SD is better than on more than 50% of the topics. In addition, we also plot a DET curve in figure 6.1 that compares the models at a pseudo feedback count of $N = 5$. The plot shows that

except in regions of low false alarm, the SD model is consistently better than the Relevance Model. The middle region in the plot is generally considered the 'region of interest' in TDT evaluation, where SD is clearly better than the RM.

In the case of tracking, the Relevance Model fails to weight the documents optimally as discussed in section 6.3.2. SD model, on the other hand, does not suffer from this problem owing to a more consistent generative framework, thereby registering consistently better performance. Another interesting observation from table 6.2 is that while all other models tend to deteriorate in performance with increasing pseudo feedback count, the performance of the SD classifier remains more stable. To illustrate this fact, note that the performance of SD at $N = 100$ is almost comparable to that of RM at $N = 5$. SD achieves this stability by using a low variance distribution for high values of N, thereby concentrating its weights only the nearest neighbors even when a large number of documents are provided for pseudo feedback. The fact that the precision S increases with increasing values of N supports this hypothesis.

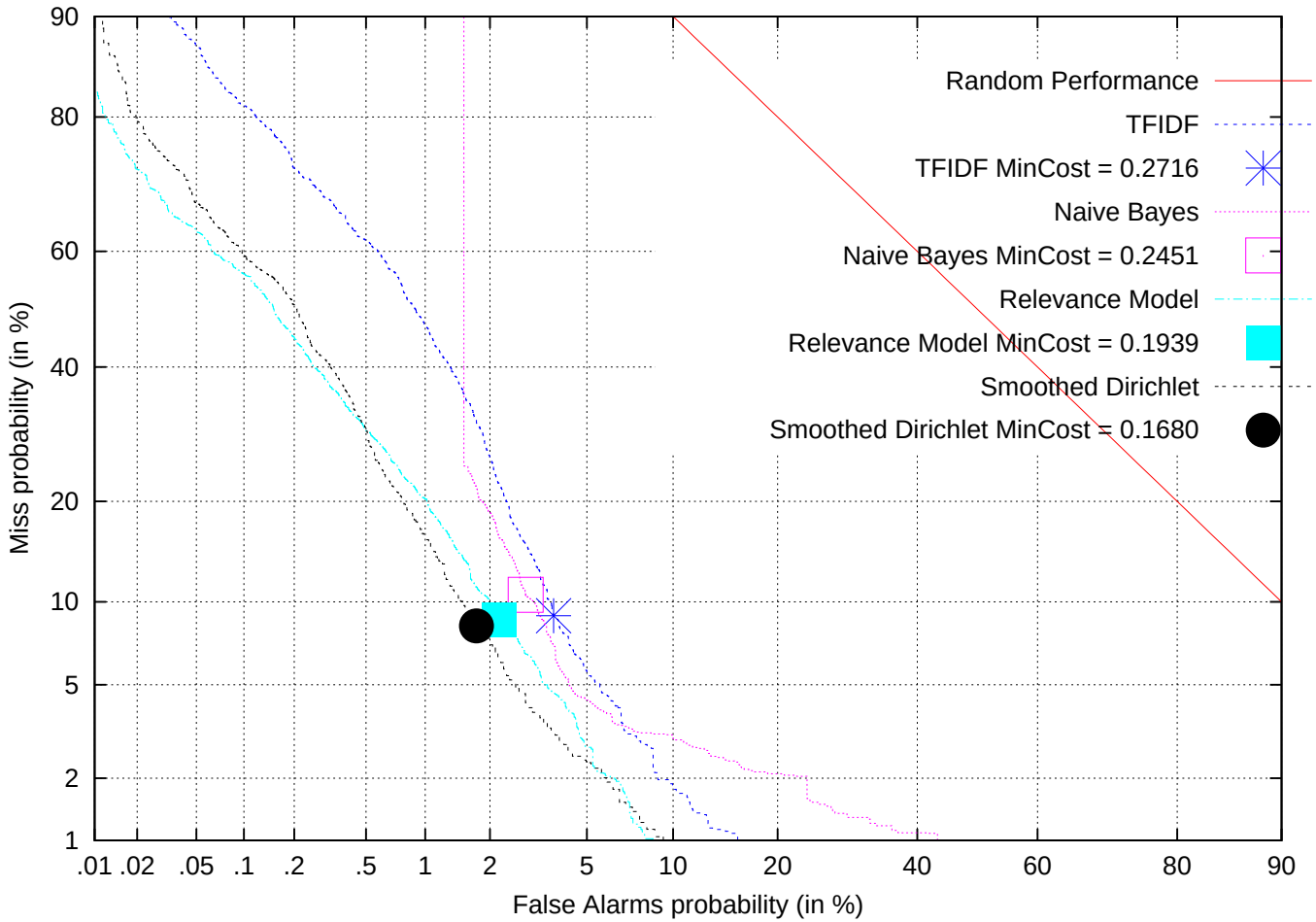

Figure 6.1. A comparison of DET curves for various models for the case where 5 pseudo feedback documents are provided for feedback.

CHAPTER 7

CONCLUSIONS

The main contribution of this work is our justification of the T-cross entropy function, hitherto used heuristically in information retrieval. We showed that the T-cross entropy ranking function corresponds to the log-likelihood of documents w.r.t. the approximate Smoothed-Dirichlet (SD) distribution, a novel variant of the Dirichlet distribution. We also empirically demonstrated that this new distribution captures term occurrence patterns in documents much better than the multinomial, thus offering a reason behind the superior performance of the T-cross entropy ranking function compared to the multinomial document-likelihood.

Our experiments in text classification showed that a classifier based on the Smoothed Dirichlet performs significantly better than the multinomial based naïve Bayes model and on par with the Support Vector Machines (SVM), confirming our reasoning. In addition, this classifier is as quick to train as the naïve Bayes and several times faster than the SVMs owing to its closed form maximum likelihood solution, making it ideal for many practical IR applications.

We also applied the SD based classifier to IR based using the EM algorithm to learn from pseudo-feedback and showed that its performance is essentially equivalent to the Relevance Model (RM), a state-of-the-art model for IR in the language modeling framework that uses the same cross-entropy as its ranking function. We overcame the problems of previous generative classification approaches for ad hoc retrieval by choosing SD as the generative distribution, which is more appropriate for text than the ones used earlier such as the multiple Bernoulli and the multinomial. Our experiments therefore show that our per-

spective of ad hoc retrieval as a classification problem is justified as long as an appropriate distribution is used to model text. In addition, our work also shows that likelihood based generative models can be successful in IR and we hope this work encourages researchers to consider IR essentially as a machine learning problem.

In addition, the SD based classifier overcomes the document weighting problem of the Relevance Model owing to a consistent generative framework and consistently outperforms the latter on the task of topic tracking.

We believe the success of our SD based generative classifier in several problems of information retrieval offers a promising unified perspective of information retrieval as a classification problem. Such perspective can be beneficial not only to promote our understanding of the domain but also to borrow ideas from one problem to apply to the other. For example, the unified perspective offered by our work allowed us to apply the T-cross entropy function to text classification via the SD distribution achieving performance better than any existing generative model in text classification. We hope that our work brings the text classification and information retrieval communities closer together, resulting in a fruitful exchange of ideas between the two active research areas.

7.1 Future work

As part of our future work, we intend to do more extensive experiments using the SD classifier on ad hoc retrieval and topic tracking. One way to further improve the performance is to model the non-relevant class appropriately. This is a difficult but interesting research challenge [29] and it needs careful investigation.

In addition, we also intend to apply an SD based mixture model to document clustering that clusters documents using the EM algorithm. Clustering techniques based on the multinomial distribution and the EM algorithm have been studied by researcher earlier [63]. Following the superior performance of the SD based classifier compared to the multinomial

based naïve Bayes model in classification, we expect the SD mixture model to outperform multinomial based techniques on the task of clustering too.

We also believe the SD distribution can potentially be used as the generative distribution for text in various topic models such as Latent Dirichlet Allocation [5], Correlated Topic Models [4] and the latest Pachinko Allocation model [27]. All these hierarchical generative models are based on the multinomial distribution as the basic building block to generate text. Since we have shown that SD distribution models text better than the multinomial, replacing the latter with the former in these models may lead to improved performance of these models. However, such replacement is not straightforward and needs further investigation. Some of the research issues involved in this problem are choosing an appropriate prior for the SD distribution and developing efficient sampling techniques from this distribution for inference and parameter estimation.

APPENDIX: ESTIMATING THE PROBABILITY MASS OF DOCUMENTS USING SD DISTRIBUTION

In chapter 3, we assumed the following approximate equivalence relationship between the counts representation of a document and its smoothed language model representation for simplicity.

$$P_{SD}(\mathbf{f}^D|\boldsymbol{\alpha}) \overset{\approx}{\equiv} P_{SD}(\boldsymbol{\theta}^D|\boldsymbol{\alpha}) \tag{7.1}$$

In this appendix, we will relax this assumption and present some analysis on how to provide upper bounds on the probability mass for documents, given the probability densities of the corresponding smoothed language models w.r.t. the SD distribution.

First we define the sets S_{MLE} and S_{LM} as follows.

$$S_{MLE} = \{\hat{\boldsymbol{\theta}} \mid \hat{\boldsymbol{\theta}} \in \Delta \ \& \ \text{int}(\hat{\boldsymbol{\theta}} \times |D|) = \mathbf{f}^D\} \tag{7.2}$$

$$S_{LM} = \{\lambda\hat{\boldsymbol{\theta}} + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE} \mid \hat{\boldsymbol{\theta}} \in S_{MLE}\} \tag{7.3}$$

where int() is a function that rounds each component of its vector argument to its nearest integer value. Thus S_{MLE} is the set of all points in the multinomial simplex which map to the word counts in the document when un-normalized by document length and rounded-off. S_{LM} is set of language models obtained by smoothing all the elements of S_{MLE}. Now the probability mass of the counts vector $\mathbf{f}^D$ is obtained by integrating over the densities of all elements in S_{LM} as given by the SD distribution as shown below:

$$P(\mathbf{f}^D|\boldsymbol{\alpha}) = \int_{\boldsymbol{\theta} \in S_{LM}} P_{SD}(\boldsymbol{\theta}|\boldsymbol{\alpha}) d\boldsymbol{\theta} \tag{7.4}$$

The set S_{LM} is a small continuous region in the multinomial simplex. Assuming the probability is uniform and is equal to the value at the centroid of the region, one may approximate the integral as follows.

$$\int_{\boldsymbol{\theta} \in S_{LM}} P_{SD}(\boldsymbol{\theta}|\boldsymbol{\alpha}) d\boldsymbol{\theta} \approx P_{SD}(\boldsymbol{\theta}^D|\boldsymbol{\alpha}) \mathcal{V}_{LM} \tag{7.5}$$

where $\boldsymbol{\theta}^D = \lambda \frac{\mathbf{f}^D}{|D|} + (1-\lambda)\hat{\boldsymbol{\theta}}^{GE}$ is the centroid of the region and $\mathcal{V}_{LM}$ is its volume. Since each $\boldsymbol{\theta}^D \in S_{LM}$ is a linear transformation of $\hat{\boldsymbol{\theta}} \in S_{MLE}$ as shown in (7.3), $\mathcal{V}_{LM}$ can be expressed in terms of the volume of the region S_{MLE} as follows.

$$\mathcal{V}_{LM} = (\lambda^V) \mathcal{V}_{MLE} \tag{7.6}$$

where V is the size of the vector $\boldsymbol{\theta}$ which is also equal to the vocabulary size. Let us now examine $\mathcal{V}_{MLE}$ is more detail. We will first consider the 2D case where $V = 2$ and generalize our observations to higher dimensions. As an example, let us consider a document of size $|D| = 10$ whose counts vector $\mathbf{f}^D$ is $\{5, 5\}$. Clearly, $\hat{\boldsymbol{\theta}}^D = (0.5, 0.5)$ lies in the set S_{MLE} of this document. Any point $\hat{\boldsymbol{\theta}}$ on the 2-D multinomial simplex such that $0.45 < \hat{\theta}_1 < 0.55;\ \hat{\theta}_2 = 1 - \hat{\theta}_1$ lies in the set S_{MLE} since it satisfies the condition int$(\hat{\boldsymbol{\theta}}|D|) = \mathbf{f}^D$. More formally, if $\hat{\boldsymbol{\theta}}^l$ and $\hat{\boldsymbol{\theta}}^r$ are the left-extremum and right-extremum 2-D multinomial points in the set S_{MLE} defined by

$$\hat{\theta}_1^l \ < \ \hat{\theta}_1^D \ \& \ \forall_{\hat{\boldsymbol{\theta}} \in \Delta} (\hat{\theta}_1 < \hat{\theta}_1^l \Rightarrow \hat{\boldsymbol{\theta}} \notin S_{MLE}) \tag{7.7}$$

$$\hat{\theta}_1^r \ > \ \hat{\theta}_1^D \ \& \ \forall_{\hat{\boldsymbol{\theta}} \in \Delta} (\hat{\theta}_1 > \hat{\theta}_1^r \Rightarrow \hat{\boldsymbol{\theta}} \notin S_{MLE}), \tag{7.8}$$

then they must satisfy the following condition:

$$|D|(\hat{\theta}_1^r - \hat{\theta}_1^l) \leq 1 \tag{7.9}$$

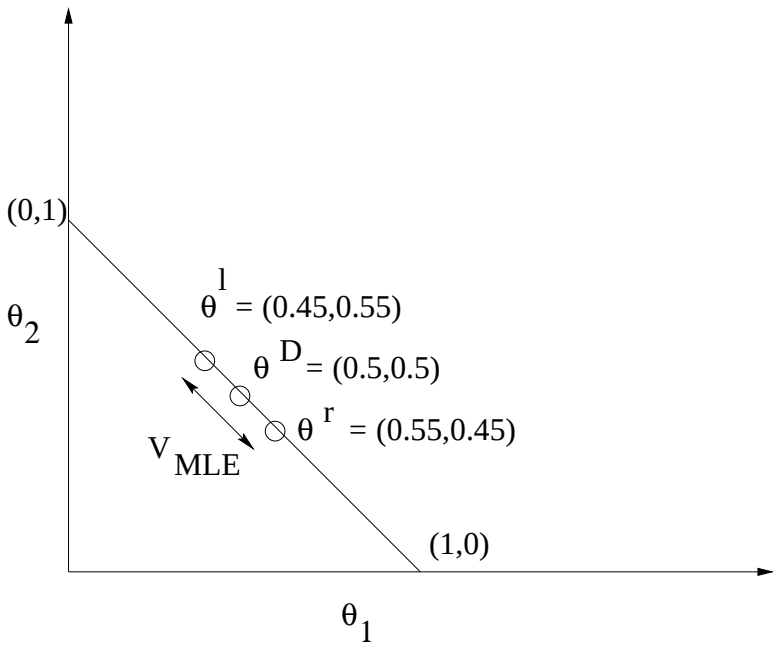

Figure 7.1. Volume of S_{MLE} in two dimensional case

This follows directly from the observations that the maximum difference between two real numbers that round to the same integer is unity. The difference $(\theta_1^r - \hat{\theta}_1^l)$ corresponds to the volume $\mathcal{V}_{MLE}$ of the set S_{MLE} since in two dimensions, this set is a straight line as shown in figure 7.1.

Thus, for the two-dimensional case, we can write

$$\mathcal{V}_{MLE}^{2D} \leq \frac{1}{|D|} \tag{7.10}$$

where the superscript ^{2D} implies that the relation is valid in two dimensions only.

Note that the above relation is only an upper bound since for points on the simplex near the boundaries, this volume is much smaller. More precisely, for points on the edges such as $(0,1)$ and $(1,0)$ in the figure, this volume is reduced by one-half since the domain of $\hat{\boldsymbol{\theta}}$ extends only one side of the edge. Thus a tighter upper bound for an edge e is given by:

$$\mathcal{V}_{MLE}^{2D}(e) \leq \frac{1}{2|D|} \tag{7.11}$$

where the notation (e) refers to the volume at the edge e.

Extending this logic to V dimensions, an upper bound for $\mathcal{V}_{MLE}$ would be the $V-1$ dimensional cube whose side l is given by $1/(|D|)$. Thus a loose upper bound for $\mathcal{V}_{MLE}$ is given by:

$$\mathcal{V}_{MLE} \leq \left(\frac{1}{|D|}\right)^{V-1} \tag{7.12}$$

As a special case, for an edge e, a tighter upper bound is given by the following.

$$\mathcal{V}_{MLE}(e) \leq \frac{1}{2^{n_e}} \left(\frac{1}{|D|}\right)^{V-1} \tag{7.13}$$

where n_e is the number of zero components in the vector representation of the edge e [1].

Using (7.12) in (7.6), we can write

$$\mathcal{V}_{LM} \leq \lambda^V \left(\frac{1}{|D|}\right)^{V-1} \tag{7.14}$$

Using this result in (7.5), the upper bound for the probability mass for the document counts vector can be written as:

$$P(\mathbf{f}^D|\boldsymbol{\alpha}) \approx P_{SD}(\boldsymbol{\theta}^D|\boldsymbol{\alpha})\mathcal{V}_{LM} \leq P_{SD}(\boldsymbol{\theta}^D|\boldsymbol{\alpha})\lambda^V \left(\frac{1}{|D|}\right)^{V-1} \tag{7.15}$$

Notice that the probability mass of a document $\mathbf{f}$ w.r.t. the SD distribution is inversely related to its document length as shown in (7.15). It remains to be seen if this approximation translates into better performance, which we intend to test as part of our future work.

[1] Clearly, in the two dimensional case, for the two edges (0,1) and (1,0), $n_e = 1$

BIBLIOGRAPHY

[1] Abramowitz, M., and Stegun, I. A. *Handbook of Mathematical Functions, National Bureau of Standards Applied Math. Series*. National Bureau of Standards, 1972.

[2] Allan, J., Bolivar, A., Connell, M., Cronen-Townsend, S., Feng, A., Feng, F., Kumaran, G., Larkey, L., Lavrenko, V., and Raghavan, H. UMass TDT 2003 research summary. In *Proceedings of the Topic Detection and Tracking workshop* (2003).

[3] Apte, C., Damerau, F., and Weiss, S. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems 12*, 3 (1994), 233–251.

[4] Blei, D., and Lafferty, J. Correlated topic models. In *Advances in Neural Information Processing Systems (NIPS)* (2005).

[5] Blei, D., Ng, A., and Jordan, M. Latent Dirichlet allocation. In *Proceedings of Neural Information Processing Systems* (2002), pp. 601–608.

[6] Boyd, S., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

[7] Carbonell, J., Yang, Y., Zhang, J., and Ma, J. CMU at TDT 2003. In *Proceedings of the Topic Detection and Tracking workshop* (2003).

[8] Cooper, W. Exploiting the maximum entropy principle to increase retrieval effectiveness. *Journal of the American Society for Information Science 34*, 1 (1983), 31–39.

[9] Dempster, A.P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* (1977), 1–38.

[10] Domingos, P., and Pazzani, M. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *International Conference on Machine Learning* (1996), pp. 105–112.

[11] Fiscus, J., and Wheatley, B. Overview of the Topic Detection and Tracking 2004 evaluation and results. In *Proceedings of the Topic Detection and Tracking workshop* (2004).

[12] Gey, F. Inferring probability of relevance using the method of logistic regression. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (1994), pp. 222–231.

[13] Harman, D. Overview of the third Text REtrieval Conference. In *TREC-3* (1994).

[14] Hofmann, T. Unsupervised learning by probabilistic latent semantic analysis. *Journal of Machine Learning 42*, 1-2 (2001), 177–196.

[15] Jeon, J., Lavrenko, V., and Manmatha, R. Automatic image annotation and retrieval using cross-media relevance models. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval* (2003), pp. 119–126.

[16] Joachims, T. Text categorization with support vector machines: learning with many relevant features. In *European Conference on Machine Learning* (1998), pp. 137–142.

[17] Joachims, T. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning* (1999), MIT Press.

[18] Kantor, P.B., and Lee, J. J. Testing the maximum entropy principle for information retrieval. *Journal of the American Society for Information Science 49*, 6 (1998), 557–566.

[19] Kantor, P.B., and Lee, J.J. The maximum entropy principle in information retrieval. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (1986), pp. 269–274.

[20] Krovetz, R. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 1993), ACM Press, pp. 191–202.

[21] Lafferty, J., and Lebanon, G. Diffusion kernels on statistical manifolds. *Journal of Machine learing research 6* (2005), 129–163.

[22] Lafferty, J., and Zhai, C. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (2001), pp. 111–119.

[23] Lavrenko, V. A generative theory of relevance. In *Ph.D. thesis* (2004).

[24] Lavrenko, V., Allan, J., DeGuzman, E., LaFlamme, D., Pollard, V., and Thomas, S. Relevance models for topic detection and tracking. In *Proceedings of the Conference on Human Language Technology (HLT)* (2002), pp. 104–110.

[25] Lavrenko, V., Choquette, M., and Croft, W. B. Cross-lingual relevance models. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2002), ACM Press, pp. 175–182.

[26] Lavrenko, V., and Croft, W. B. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2001), ACM Press, pp. 120–127.

[27] Li, W., and McCallum, A. Pachinko allocation: DAG-structured mixture models of topic correlations. In *International Conference on Machine Learning* (2006).

[28] Lo, Y., and Gauvain, J. The LIMSI topic tracking system for TDT. In *Proceedings of the Topic Detection and Tracking workshop* (2002).

[29] Losada, D. E., and Barreiro, A. An homogeneous framework to model relevance feedback. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (2001), pp. 422–423.

[30] Madsen, R. E., Kauchak, D., and Elkan, C. Modeling word burstiness using the Dirichlet distribution. In *International Conference on Machine Learning* (2005), pp. 545–552.

[31] McCallum, A., and Nigam, K. A comparison of event models for naive Bayes text classification. In *In AAAI-98 Workshop on Learning for Text Categorization* (1998), pp. 41–48.

[32] McCallum, A., Rosenfeld, R., Mitchell, T., and Ng, A. Improving text classification by shrinkage in a hierarchy of classes. In *International Conference on Machine Learning* (1998), pp. 359–367.

[33] Minka, T. Estimating a dirichlet distribution, 2003.

[34] Nallapati, R. Discriminative models for information retrieval. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2004), ACM Press, pp. 64–71.

[35] Nallapati, R., and Allan, J. Capturing term dependencies using a language model based on sentence trees. In *Proceedings of the eleventh international conference on Information and Knowledge Management* (New York, NY, USA, 2002), ACM Press, pp. 383–390.

[36] Ng, A., and Jordan, M. On discriminative vs. generative classifiers: A comparison of logistic regression and naïve bayes. In *Advances in Neural Information Processing Systems* (2001), pp. 605–610.

[37] Nigam, K., Lafferty, J., and McCallum, A. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering* (1999), pp. 61–67.

[38] Ponte, J., and Croft, W. B. A language modeling approach to information retrieval. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (1998), pp. 275–281.

[39] Porter, M.F. An algorithm for suffix stripping. *Program 14*, 3 (1980), 130–137.

[40] Rennie, J., and Jaakkola, T. Using term informativeness for named entity detection. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2005), ACM Press, pp. 353–360.

[41] Rennie, J., Shih, L., Teevan, J., and Karger, D. Tackling the poor assumptions of naive Bayes text classifiers. In *International Conference on Machine Learning* (2003), pp. 616–623.

[42] Robertson, S., and Hull, D.A. The TREC-9 filtering track final report. In *Proceedings of the 9th Text Retrieval Conference* (2000).

[43] Robertson, S., and Walker, S. Threshold setting in adaptive filtering. *Journal of Documentation* (2000), 312–331.

[44] Robertson, S. E., Rijsbergen, C. J. Van, and Porter, M. F. Probabilistic models of indexing and searching. *Information Retrieval Research* (1981), 35–56.

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

[46] Robertson, S.E. The probability ranking principle in information retrieval. *Journal of Documentation 33* (1977), 294–304.

[47] Robertson, S.E., and Walker, S. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (1994).

[48] Rocchio, J. J. Relevance feedback in information retrieval. *The Smart Retrieval System - Experiments in Automatic Document Processing* (1971), 313–323.

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

[50] Salton, G., and McGill, M. J. Computer evaluation of indexing and text processing. *Journal of the ACM 15*, 1 (1968), 8–36.

[51] Teevan, J. *Improving information Retrieval with textual analysis: Bayesian models and Beyond*. Master's thesis, MIT, 2001.

[52] Teevan, J., and Karger, D. Empirical development of an exponential probabilistic model for text retrieval: Using textual analysis to build a better model. In *Proceedings of the annual international ACM SIGIR conference on Research and development in information retrieval* (2003), pp. 18–25.

[53] Tong, S., and Koller, D. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res. 2* (2002), 45–66.

[54] van Rijsbergen, C. J. *Information Retrieval*. Butterworths, London, 1979.

[55] Xu, J., and Croft, W. B. Query expansion using local and global document analysis. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (1996), pp. 4–11.

[56] Y. W. Teh, M. Jordan, M. Beal, and Blei, D. Hierarchical dirichlet processes. In *Technical Report 653, UC Berkeley Statistics* (2004).

[57] Yang, Y., and Kisiel, B. Margin-based local regression for adaptive filtering. In *Proceedings of the twelfth international conference on Information and knowledge management* (New York, NY, USA, 2003), ACM Press, pp. 191–198.

[58] Zaragoza, H., Hiemstra, D., and Tipping, M. Bayesian extension to the language model for ad hoc information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (2003), pp. 4–9.

[59] Zhai, C., and Lafferty, J. Model-based feedback in the language modeling approach to information retrieval. In *ACM Conference on Information and Knowledge Management* (2001), pp. 403–410.

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

[61] Zhang, D., Chen, X., and Lee, W. S. Text classification with kernels on the multinomial manifold. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2005), ACM Press, pp. 266–273.

[62] Zhang, Y., and Callan, J. Maximum likelihood estimation for filtering thresholds. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2001), ACM Press, pp. 294–302.

[63] Zhong, S., and Ghosh, J. Generative model-based document clustering: A comparative study. *Knowledge and Information Systems 8* (2005), 374–384.

[64] Zipf, G. *Human behavior and the principle of least effort: An introduction to human ecology*. Addison Wesley, 1949.