NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

DISSERTATION

NEW FRAMEWORK FOR CROSS-DOMAIN DOCUMENT CLASSIFICATION

by

Anjum Gupta

March 2011

Dissertation Supervisor: Craig Martell

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704–0188*

1. REPORT DATE *(DD–MM–YYYY)* 03-03-2011	**2. REPORT TYPE** Dissertation	**3. DATES COVERED** *(From — To)* 2009-01-01—2011-03-25

4. TITLE AND SUBTITLE New Framework for Cross-Domain Document Classification	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Anjum Gupta	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Department of the Navy	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. **IRB Protocol Number: n/a**

14. ABSTRACT

Automatic text document classification is a fundamental problem in machine learning. Given the dynamic nature and the exponential growth of the World Wide Web, one needs the ability to classify not only a massive number of documents, but also documents that belong to wide variety of domains. Some examples of the domains are e-mails, blogs, Wikipedia articles, news articles, newsgroups, online chats, etc. It is the difference in the writing style that differentiates these domains. Text documents are usually classified using supervised learning algorithms that require large set of pre-labeled data. This requirement, of labeled data, poses a challenge in classifying documents that belong to different domains. Our goal is to classify text documents in the testing domain without requiring any labeled documents from the same domain. Our research develops specialized cross-domain learning algorithms based the distributions over words obtained from a collection of text documents by topic models such as Latent Dirichlet Allocation (LDA). Our major contributions include (1) empirically showing that conventional supervised learning algorithms fail to generalize their learned models across different domains and (2) development of novel and specialized cross-domain classification algorithms that show an appreciable improvement over conventional methods used for cross-domain classification that is consistent for different datasets. Our research addresses many real-world needs. Since massive number of new types of text documents is generated daily, it is crucial to have the ability to transfer learned information from one domain to another domain. Cross-domain classification lets us leverage information learned from one domain for use in the classification of documents in a new domain.

15. SUBJECT TERMS

Cross Domain Classification, Text Mining, Machine Learning, Genre Shift, Document Classification

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	b. ABSTRACT	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)*
Unclassified	Unclassified	Unclassified	UU	177	

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

NEW FRAMEWORK FOR CROSS-DOMAIN DOCUMENT CLASSIFICATION

Anjum Gupta
Civilian, Spawar Systems Center, Pacific
San Diego, CA
B.S., University of California, San Diego, 2001
M.S., University of California, San Diego, 2005

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2011

Author: Anjum Gupta

Approved by: Dr. Craig Martell, Dissertation Advisor — Dr. Craig Martell, Dissertation Committee Chair

Dr. Ralucca Gera, Advisor — Dr. Andrew I. Schein, Advisor

Dr. Dennis Volpano, Advisor — Dr. Joel Young, Advisor

Approved by: Dr. Peter J. Denning
Chair, Department of Computer Science

Approved by: Dr. Douglas Moses
Vice Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Automatic text document classification is a fundamental problem in machine learning. Given the dynamic nature and the exponential growth of the World Wide Web, one needs the ability to classify not only a massive number of documents, but also documents that belong to wide variety of domains. Some examples of the domains are e-mails, blogs, Wikipedia articles, news articles, newsgroups, online chats, etc. It is the difference in the writing style that differentiates these domains. Text documents are usually classified using supervised learning algorithms that require large set of pre-labeled data. This requirement, of labeled data, poses a challenge in classifying documents that belong to different domains. Our goal is to classify text documents in the testing domain without requiring any labeled documents from the same domain. Our research develops specialized cross-domain learning algorithms based the distributions over words obtained from a collection of text documents by topic models such as Latent Dirichlet Allocation (LDA). Our major contributions include (1) empirically showing that conventional supervised learning algorithms fail to generalize their learned models across different domains and (2) development of novel and specialized cross-domain classification algorithms that show an appreciable improvement over conventional methods used for cross-domain classification that is consistent for different datasets. Our research addresses many real-world needs. Since massive number of new types of text documents is generated daily, it is crucial to have the ability to transfer learned information from one domain to another domain. Cross-domain classification lets us leverage information learned from one domain for use in the classification of documents in a new domain.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgements

I would like to thank my committee members and friends for their help and guidance throughout my studies. Specifically, I would like to thank Dr. Dennis Volpano and Dr. Ralucca Gera for always making sure that I always kept my higher level goals of the dissertation in focus and did not get lost in the details; Craig Martell, Joel Young and Andrew Schein for their guidance with more technical details and comments on the draft version of the dissertation. I would like to thank some of my friends and graduate students from U.C. San Diego for their helpful discussions on machine learning and topic models, specifically, Shibin Parameswaran, Dr. Ali Irturk and Dr. Pew Putthividhya. I would like to extend special thanks to my committee chair, Dr. Craig Martell, for serving as an ideal mentor and guide throughout the process. He was always available to guide and help with any part of the process of Ph.D. His patience, guidance and support was essential in the successful completion of this dissertation.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1: Introduction

Automatic text document classification is a fundamental problem in machine learning. Text document classification is as old as writing itself; in fact, libraries dating back to 2000 BC in Syria were archiving tens of thousands of clay tablets [1]. With the advent of paper making, a new era of written documents started. Institutions such as the library of Alexandria organized hundreds of thousands of scrolls in their archives [2]. The next major explosion in text documents came about with the invention of the printing press; the text documents became easily available and widely distributed. However, as the number of text documents increased, the need for robust classification became more crucial. A few standards of text organization were implemented for small and large libraries such as Dewey Decimal System in 1876 [3] and library of Congress system in 1897 [4].

We are recently experiencing a third, much larger, growth in the number of text documents with the advent of the Internet and the development of the World Wide Web. It is currently inconceivable to manually classify this large number of text documents that are generated daily. Yahoo Inc. made some of the earliest attempts to neatly organize all the information, available on the World Wide Web, in a directory structure. They filled the much needed gap in the Web organization, thereby helping them to become one of the first and most successful Internet companies. However, Yahoo's directory based organization system soon became inadequate and outdated due to the rapid and exponential growth in the number of the text documents. Therefore, new and more intelligent ways of searching and organizing text documents needed to be introduced. Google Inc. arose as one of the most successful companies for searching information in an ever growing Web of text documents.

Given the dynamic nature of the Web, the solution to the problem of text document classification needs to address the exponential growth of the Internet and wide variety of text documents. This wide variety of text documents is a result of documents being generated in various domains. These domains can be e-mails, blogs, wiki articles, news articles, twitter posts, message forums, online chats, speech transcripts, etc. In achieving this goal, one of the most important challenges is the problem of learning topics in text documents that belong to different domains. These domains represent information in different ways, each serving a particular purpose. Often a classification scheme that works well in one domain does not work as well in another. Given the nature of current text document classification algorithms, we hypothesize that an algorithm that is trained to classify e-mails well may not be able to classify news articles with the same or similar accuracy. We confirm this hypothesis with our experimental results in this thesis. This drop in the accuracy while going from one domain to another is what we refer to as cross-domain classification problem. This dissertation focuses on the problem of cross-domain classification. That is, we use classification information from one domain and apply that learned information to classify text documents from a different domain.

For ease of understanding, we illustrate some of the terms used in this dissertation such as **domain, topic and category** with an example on Wikipedia and New York Times (NYT) newspaper. The articles written on Wikipedia covering different topics constitute **a domain** and the articles written in the NYT newspapers by professional journalists constitutes **another domain**. These two domains may share the subject matter e.g., both may contain articles on political elections. However, each domain may also have its own style of writing while covering the particular topic. The difference in the writing style is due to the differences in the editing process, caliber of the writers, purpose of the articles etc. It is this difference in the writing style that differentiates these two domains such as Wikipedia and NYT. In our context, **topic** could have two meanings. It could describe the category of the document or describe an output of a topic model. We will generally use the term **category** for the former. So an article discussing the culinary arts of Japan will belong to cooking category and Japan category. Two different domains, e.g., Wikipedia and NYT, may both have articles on the category cooking.

Text documents can be classified either by using supervised learning or unsupervised learning. Supervised learning method classifies the documents by using a set of pre-labeled data. A third class of learning algorithm, referred to as semi-supervised learning, is often used when there is lack of labeled data. Semi-supervised learning leverages both supervised and unsupervised learning techniques. Each of the classification algorithms has its own advantages and disadvantages. In the following two paragraphs, we introduce supervised and unsupervised learning algorithms in more detail.

In supervised learning, an algorithm is presented with a set of correctly “labeled” data that it can use to learn the distribution of the data into different classes. Supervised learning is called “supervised” because the algorithm is given external input, a supervision, about what is wrong and what is right, i.e., it is being supervised by the labeled data. Consider training an algorithm to recognize human faces in images. With supervised learning, the algorithm will be provided with multiple images and informed whether each image is a face or not. The supervised algorithm then learns the differentiating characteristics between face and non-face images. Once these characteristics are learned, the algorithm can be used to classify new and unlabeled images into face and non-face categories. Common supervised learning algorithms include artificial neural networks (ANN) [5], support vector machines (SVMs) [6], k-nearest neighbors (KNN) [7] and naive Bayes (NB) [8].

In unsupervised learning, labeled data is not available. Instead of putting items into fixed categories, the algorithm tries to discover patterns of clusters. Using the previous example, the algorithm is given the set of images of faces and non-faces but without the information of which image is a face or not a face. Ideally, the algorithm will find two distinct clusters: a cluster containing all the face images and another cluster with all the non-face images. However, it is more likely that the algorithm will end up finding many different clusters each capturing some aspect of an image. As one can see, outcome of unsupervised algorithm can be unpredictable; therefore it can reveal patterns that were previously unknown. For example, an unsupervised learning algorithm can reveal patterns in the gene expression, in the subjects of a certain genetic disorder, which could not be detected by manual inspection of the data. These algorithms are also often used for applications such as e-discovery used to locate electronic data for legal cases. Common unsupervised algorithms include k-means [9], expectation-maximization algorithm for mixture of Gaussians (EM) [10], latent Dirichlet allocation topic (LDA) model [11]. Outcome of an unsupervised algorithm is often a probability distribution of the given data points. In our example, each image, face or not face, will constitute a data point in this distribution. The models learned by probabilistic unsupervised algorithms are also called generative models because they provide a method to generate new data points by revealing the underlying graphical model. However, it is important to note that not all generative models are generated by probabilistic unsupervised methods.

Both supervised and unsupervised algorithms present their own advantages and disadvantages based on a particular application. As with the image classification example, the unsupervised learning algorithm, in most cases, will not detect patterns in the data according to perceived labels assigned by humans. Since there is no external input directing the unsupervised algorithm to learn the differences between pre-determined classes, unsupervised algorithms often learn other subtle patterns that may or may not be useful. In summary, unsupervised learning algorithms can reveal interesting and unexpected patterns. Supervised learning algorithms can learn the differences between pre-determined classes, however, they present their own disadvantages by requiring a large number of labeled data points. In many cases, obtaining a large dataset that has been carefully labeled is not feasible. Supervised learning algorithm forces itself to learn any pattern given and generalizes it to classify the new data. If the data was not carefully labeled or the number of labeled data points was very small, the performance of the supervised learning algorithm on new and unlabeled data is often very poor [12], [13]. In summary, supervised learning algorithms offer a way to classify information in pre-determined categories given a carefully labeled large number of data points.

We will illustrate the use of learning algorithms for our cross-domain classification task with an example. Suppose that we want to classify articles from NYT into one of the pre-determined categories. In this case, NYT will be our **testing domain**. In order to classify NYT articles, we will use labeled articles from Wikipedia to learn a **classifying model**. In this case, Wikipedia will be our **training domain**. **A classifying model** is a function that takes an unlabeled article as its input and returns a category label as the output.

Text document classification often requires one to classify documents in specific, pre-determined, categories. At the same time, we lack the ability to generate large amount of labeled data for documents especially with the new topics and new domains. The desire to classify documents in pre-determined categories poses problems for unsupervised learning, while the lack of labeled data is problematic for supervised learning. It is sometimes the case, as shown in this dissertation, that supervised algorithms taught to classify documents in one domain cannot classify similar documents in different domains. We show that when a conventional supervised algorithm, e.g., SVM, is trained to classify news articles from New York Times (NYT) newspaper, it classifies NYT articles with good accuracy. However, when that learned classifier is used to classify Wikipedia articles, the accuracy of classification drops significantly. In order for a supervised algorithm to work well for multiple domains, it has to be trained again and again for each new domain that will be classified. Each time it needs a new labeled set of documents from the training domain. Our algorithms, developed as part of this research, are able to learn the classifying model from one domain (training domain) and use it on a different domain (testing domain) with appreciable improvement in the accuracy when compared with conventional algorithms.

Our goal is to classify text documents in a new (testing) domain by using labeled set of documents from a different domain (training). In other words, we want to be able to classify text documents in a new domain without requiring any set of labeled documents from this domain.

This is what we refer to as the cross-domain classification. Our research develops specialized cross-domain learning algorithms to accomplish this goal. Major contributions of this dissertation include:

- Define the cross-domain problem and establish new framework to tackle the problem. We also present our framework's relation to the previous work done in the related field;
- Gather evidence that suggests that cross-domain classification is more challenging problem than single domain classification problem. We present empirical results showing a significant drop in classification accuracy when different domains are used for testing and training;
- Analysis of the cross-domain classification problem by tabulating the reasons of this considerable drop in the classification accuracy;
- Novel and specialized cross-domain classification algorithms that show an appreciable improvement, over conventional methods used for the same task, that is consistent for different datasets;
- Development and presentation of a new dataset that can be useful for many researchers working in the field of cross-domain classification.

To understand the challenges of cross-domain classification task, we first start by performing experiments using Wikipedia as our only training domain. This research provides us a tremendous insight about the cross-domain classification task. While working with Wikipedia, we only develop algorithms that specifically use Wikipedia as the training set to classify text documents in any other domain. These initial algorithms are customized for Wikipedia and they utilize the unique properties of Wikipedia articles such as the presence of external and internal links. We learn from these experiments and then extend our work to be more general, that is, our work consists of general set of algorithms that are not tied down to any particular training domain.

In our algorithms, we utilize both supervised and unsupervised algorithms to accomplish the cross-domain classification task. We use some of the common supervised algorithms such as k-nearest neighbors, support vector machines and naïve Bayes. The unsupervised algorithms used in our methods are primarily based on topic models such as Latent Dirichlet Allocation (LDA) and probabilistic Latent Semantic Analysis (pLSA). We go over two different ways of using

LDA for classification, (1) using the vectors describing the distribution of topics in documents (γ vectors) and (2) using the topic as distribution of words (β vectors). We describe the methods and distance metrics for our methods and give empirical results.

Topic models are generative models for a collection of text documents. They assume that documents may contain a mixture of topics, where topics are defined as distributions over words. There are many topic models that developed to model text document collections, each making different assumptions about the relationships between topics, words, documents and their underlying distributions. Most topic models operate in unsupervised learning framework i.e. they look to extract generative distribution of the collection instead of focusing on dividing documents into distinct classes. There are, however, enough variations of the topic models that try to incorporate the external labels provided to the algorithm in shaping the underlying distribution. In this way, these algorithms work in a more semi-supervised manner. Our newly developed algorithms, in this dissertation, do not develop new generative models. There are many variations of the topic models developed or modified for different purposes. Our algorithms concentrate on utilizing the information extracted from a topic model in a way that is useful for cross-domain classification. We provide both theoretical and empirical justification for the robustness of our algorithms.

For developing and verifying the effectiveness of our cross-domain classification algorithms, we use text documents from three different domains for our experiments: (1) Wikipedia, (2) New York Times newspaper and (3) newsgroups. All three domains are used as target and testing domains at different times to show the consistency and robustness of our algorithms in cross-domain classification.

Cross-domain classification has many real world applications. Given the dynamic nature of the Web, massive number of text documents, belonging to new and wide variety of domains, is generated daily. In this environment, it is essential to be able to transfer learned information from one domain to another domain. If one wants to classify documents consisting of text chat transcripts, conventional algorithms will require a large number of labeled chat transcripts, before being to classify them. Such a labeled data may be very difficult to obtain or may not even exist at all. Our algorithms, on the other hand, will be able to use the information learned from a different domain, such as Wikipedia and will be able to classify the chat transcripts without requiring any labeled chat data. In short, cross-domain classification lets us leverage information learned from one domain for use in the classification of documents in a new or a different domain.

This thesis is organized as follows: Chapter 2 discusses the related work in the field of cross-domain classification. Chapter 3 outlines some machine learning algorithms utilized in this research. Chapter 4 gives the details of the data sets used in our research. Chapter 5 presents our experiments with Wikipedia as the chosen training domain. It concentrates on techniques of parsing and using different sections of the Wikipedia articles to learn a document classifier. Chapter 6 presents the empirical results of cross-domain classification using conventional methods. It shows a considerable drop in the accuracy when different domains are used for training and testing. Chapter 7 describes our algorithms for cross-domain classification along with the experimental results and the analysis of these results. We run Wilcoxon rank sum statistical tests to show the statistical significance of our results presented in Chapter 7. These results are shown in Appendix A of this dissertation. We conclude with future work in the area of cross-domain classification in Chapter 8.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Related Work

Our research touches on two main research areas: machine learning and text document classification. Our goal is to accomplish cross-domain text document classification using machine learning algorithms. This goal leads us to explore a wide array of research areas including experiments with Wikipedia data, theoretical distance measures of probability distributions and various topic models for text documents. We gained substantial insights from the related work discussed in this chapter. This chapter discusses the related work that includes: (1) classification of text documents using Wikipedia, (2) topic models for supervised learning, (3) using unsupervised topic models for classification, (4) methods for cross-domain knowledge transfer and (5) topic models for cross-domain classification. Some of the common machine learning algorithms such as Latent Dirichlet Allocation (LDA), Support Vector Machines (SVMs), and Naive Bayes (NB) will be introduced in Chapter 3.

2.1 Classification of Text Documents Using Wikipedia

Wikipedia, in the last few years, has emerged as an invaluable resource of text documents. Wikipedia offers a thorough organizational structure of the articles in various categories. Each article belongs to one or more categories, although there may be articles without a category. The article, itself, is also organized into different sections such as introduction, references, notes etc. Wikipedia has been analyzed in various publications [14, 15, 16, 17, 18]. While some of these papers have concentrated on the evolution of Wikipedia and social aspects of the Wikipedia community [17, 18], we are more interested in researching the organizational structure of its articles. Given the structure and the size of Wikipedia, there is also a tendency to use it for text document classification. There are many different ways to use Wikipedia in aid of text document classification. Among these different ways, one of the prominent ways is using Wikipedia as a source of *semantic background knowledge.* As a *semantic background knowledge* source, Wikipedia is used to augment another text data [15, 19]. Our goal is to exploit the organizational structure of Wikipedia and its articles for text document classification. The following paragraphs will introduce these related works in more detail.

Wang et al. attempted to improve the accuracy of text document classification by augmenting their training data with Wikipedia content [19]. The authors first created the feature vectors, which are the word counts in the documents using the bag-of-words model. The authors then incorporated word relations: synonymy, polysemy, and hyponymy, into this feature vector. This is done by simply adding the counts of the related words in the feature vector, however, the authors' use of Wikipedia is limited, since their approach concentrated only extracting the three different types of relationships among words. They extracted the word relations using the hyperlinks between the Wikipedia articles and did not use any other part of the Wikipedia data. The authors have also published a similar paper presenting a way of using Wikipedia to develop a thesaurus [20].

Gabrilovich et al. had an approach that is more elaborate in its use of Wikipedia text [21]. The authors' approach not only concentrated on explicitly defined lexical relationships among the words, but also used the document text itself to augment the feature vectors. In their experiments, they showed a modest but consistent improvement in the classification accuracy of various datasets including 20-newsgroups [22], where the accuracy increased from 85.4% to 86.2%.

Schonhofen presented a way of using Wikipedia categories in the classification of text documents [23]. The author did not utilize the Wikipedia article text in his classification algorithm; he only used the category labels and the titles of the articles. He experimented by classifying Wikipedia articles as well as the 20-newsgroup articles. Their 20-newsgroup classification did show a modest improvement when it is performed using the Wikipedia information.

Wang et al. improved on the results of co-clustering algorithm [24] by using the Wikipedia text [25] to augment their training data. The authors also used 20-newsgroups data and split this data into two domains. They used some of the newsgroups as first domain and some other similar newsgroups to represent the corresponding second domain. As an example from the research article, the first domain contained newsgroups such as recreation.autos and recreation.baseball, while the second domain contained recreation.motorcycles and recreation.hockey.

Medelyan et al. have written a survey article of research on Wikipedia for text mining purposes [26]. The authors focused this survey on the research that extracts and makes use of the concepts, relations, facts and descriptions found in Wikipedia. They organized the work into four broad categories: applying Wikipedia to natural language processing, using it to facilitate information retrieval, information extraction, and as a resource for ontology building. This survey is a great resource for researchers who are interested in using Wikipedia for text mining.

Using data from Wikipedia to enhance an already labeled text data has been researched by many as we have shown in this section. However, using different sections within a Wikipedia article for the purpose of text mining has not been given as much attention. In our research we fill this gap in the literature. In our Wikipedia research, we parse and analyze different parts of a typical Wikipedia article, which we refer to as sections of the article. Chapter 5 discusses the properties of these individual sections. We present our experimental results in cross-domain classification, where Wikipedia text is used to enhance the classification of documents in a different domain.

2.2 Topic Models for Supervised Learning

Topic models are generally unsupervised probabilistic generative models based on a Bayesian network and are often used for modeling a document collection. Different topic models are developed by modifying and/or extending the underlying Bayesian networks. For example, the LDA model extends the probabilistic Latent Semantic Analysis (pLSA) model by introducing Dirichlet priors over the topic distributions. As one can imagine, many topic models have been developed over the years by modifying other topic models in different ways. Some of these modifications are done to develop topic models for supervised learning. However, a common theme is shared by all topic models. Heinrich presented the underlying commonality of topic models elegantly in his article titled “A Generic Approach to Topic Models” [27].

Even though most topic models remain unsupervised, in this section, we will discuss some of the topic models that use the document labels. The label of a document is inserted in the Bayesian network as an *observed* node and influences the probabilities and distribution of the words in topics. This is the case with the models in [28, 29, 30, 31, 32, 33] and many others. Some of these papers such as [32, 33] do not address the text document classification problem in general but use the model to tackle problems of segmentation and recognition of the objects in images and video.

Andrzejewski et al. incorporated supervision into the LDA model [31]. This is done by restricting a set of the words from a given set of topics. For example, topic set, {1,2}, may not contain the set of words such as {british, uk, wales, london}. Authors named this model as LDA-topic-in-subset (LDA-TIS). More specifically, LDA-TIS constrains a set of words to be only assigned to specific subset of topics. Authors have also published a previous paper to find bugs in programming code, where they partition the topics into two sets: (1) *usage topics* which can appear in all documents, and (2) *bug topics* which can only appear in a special subset of documents [34].

The authors added the TIS information into LDA via a modification of the Gibbs sampling equations, which are shown below:

$$q_{nk} = \left(\frac{\sum_d \sum_{m \neq n} 1\left(w_m^d = t\right) 1\left(z_m^d = k\right) + \gamma}{\sum_{d,m,t} 1\left(w_m^d = t\right) 1\left(z_m^d = k\right) + T\gamma} \right) \left(\sum_{m \neq n} 1\left(z_m^d = k\right) + \alpha \right) \tag{2.1}$$

and then defining p as follows by adding a parameter η to control the strength of the constraint:

$$p\left(z_n = k \mid Z_{\neg n}, W\right) \propto q_{nk}\left(\eta\, \delta\left(k \in C^{(n)}\right) + (1 - \eta)\right) \tag{2.2}$$

where $C^{(n)}$ is the set of possible values for the latent topic z_n for word w_n. The indicator function $\delta\left(k \in C^{(n)}\right)$ takes a value of 1 if $k \in C^{(n)}$ and 0 otherwise. For example, if we wish to restrict z_n to a single value, this can be accomplished by setting the set $C^{(n)}$ to topic 5. Likewise, we can set $C^{(n)}$ to a subset of topics, e.g., $\{1, 2, 3\}$, or to the whole subset $\{1, 2, \ldots, K\}$ in which case the modified sampling reduces to the standard collapsed Gibbs sampling. The formation above gives a flexible way to insert prior knowledge into the inference of latent topics. We can set $C^{(n)}$ individually for every word w_n in the corpus. This allows us to force two occurrences of the same word in a document to be explained by different topics. This effect indeed would be impossible to achieve using standard LDA. The authors performed two experiments which are explained in the following paragraphs in more detail.

The first experiment is performed using a corpus of 9000 yeast-related abstracts. The authors restricted the topics for a set of seed words, translation, trna, anticodon, ribosome, to be topic 0, for all occurrences of the seed words. Their goal was to discover if LDA-TIS can guide the topic discovery to be more related to the user-seeded concepts. The authors compared the topics learned from standard LDA with LDA-TIS and found that LDA-TIS is able to group all terms relevant to the seeded words to topic 0, while standard LDA ends up splitting the relevant terms between three different topics.

Their second experiment was similar to the first, but authors performed this experiment using the Reuters newswire corpus [35]. The difference between the two experiments was that the seeded terms $\{$britain, british, uk, u.k., wales, scot-land, london$\}$ are now allowed to be in a subset of topics (topics 1,2,3) instead of being restricted to only one topic. In addition, all the other location-related terms are not allowed to be in the first three topics. While LDA-TIS uncovered the first three topics to be related to the location "United Kingdom", the topics are also split nicely into business (topic 1), cricket (topic 2), and soccer (topic 3). In contrast, the standard LDA model was not able to discover such interesting topic patterns. Even though the LDA-TIS has promising results for this particular application, it could not repeat the same success with other locations related seed terms such as China, United States and Germany. In summary, the LDA-TIS cannot be very successful if the users' target concepts are not prevalent in the text corpus.

Blei et al. have also proposed a supervision modification of their LDA model called, Supervised LDA (sLDA) [28]. The authors performed a small modification on the LDA model by adding a *response variable* to the model that depends on the document and the topic distribution of that document. The sLDA model can be written as

$$\begin{aligned} & p\left(x_{1:N}, y|\alpha, \beta_{1:k}, \eta, \sigma^2\right) \\ & \quad = \int_\pi p\left(\pi|\alpha\right) \sum_{z_{1:N}} \left(\prod_{n=1}^{N} p\left(z_n|\pi\right) p\left(x_n|z_n, \beta_{1:k}\right) \right) p\left(y|z_{1:n}, \eta, \sigma^2\right) d\pi \end{aligned} \tag{2.3}$$

where the variables in the Equation 2.3 are taken from the LDA model as described in the Background chapter. Under sLDA, one generates the documents the same way as she would under a normal LDA. Once the topic proportions (Z_n) are generated, you draw a response variable y. y is a real-value random variable and $y|z_{1:N}, \eta, \sigma^2 \sim N\left(\eta^T \bar{z}, \sigma^2\right)$ where $\bar{z} := (1/N) \sum_{n=1}^{N} z_n$ and η is an observed, learned, parameter for the response variable.

Their experiments used the star ratings from movie reviews data [36] and number of votes each link received from Digg.com's data [28] as the response variables. Since the distribution of the response variable in the experimental datasets was not normal, they attempted to achieve normality by taking the log of these numbers. They evaluated the performance of the model by computing R^2, which they called predictive R^2, evaluated as $pR^2 := 1 - \left(\sum (y-\hat{y})^2\right) / \left(\sum (y-\bar{y})^2\right)$. They compared their results of pR^2 with the L_1 regularized least-squares regression and showed 8% to 9% improvement. As can be seen, the sLDA proposes a rather simple, normally distributed, continuous response variable that may not be suitable for many real world classification tasks.

Shan et al. proposed some changes to the sLDA model, primarily in the distribution of the response variable [30]. The authors developed two new models, one by modifying the response variable in sLDA and another by modifying the response variable in their own topic model, Latent Dirichlet conditional naïve Bayes model, [37]. The authors made the response variable more suitable for classification labels, which are rarely distributed normally and are almost never continuous or real valued. The modification to the LDA with their response variable was called discriminative-LDA (DLDA).

Here we will briefly discuss their models and compare it with sLDA, which was introduced in the previous paragraph. The DLDA model is written as below.

$$\begin{aligned} & p\left(x_{1:N}, y|\alpha, \beta_{1:k}, \eta_{1:c-1}\right) \\ & \quad = \int_{\pi} p\left(\pi|\alpha\right) \sum_{z_{1:N}} \left(\prod_{n=1}^{N} p\left(z_n|\pi\right) p\left(x_n|z_n, \beta_{1:k}\right) \right) p\left(y|z_{1:N}, \eta 1 : c-1\right) d\pi \end{aligned} \tag{2.4}$$

where the variables in the Equation 2.4 are taken from the LDA model as described in the Background chapter. The only difference when compared with sLDA is that in sLDA (eq. 2.3) the response variable y depends on a scalar η, σ squared and $\bar{z}$. The response variable y in their model is given by a multi-class logistic regression as shown below.

$$y \sim LR\left(\frac{\exp(\eta_h^T \bar{z})}{1 + \sum_{h=1}^{c-1} \exp(\eta_h^T \bar{z})} \right) \tag{2.5}$$

where $\bar{z}$ is an average of $z_{1:N}$ over all observed words, where each z_n is a k-dimensional unit vector with only the i^{th} entry being 1 if it denotes the i^{th} component. The categorical response variable y can be considered as a sample generated from the Discrete distribution $(p_1, \dots, p_{c-1}, 1 - \sum_{h=1}^{c-1} p_h)$ where $p_h = \frac{\exp(\eta_h^T \bar{z})}{1+\sum_{h=1}^{c-1} \exp(\eta_h^T \bar{z})}$. In two-class classification, y is 0 or 1 generated from Bernoulli$\left(\frac{1}{1+\exp(-\eta^T \bar{z})}\right)$, i.e. the model needs only one η in the two-class case.

The authors illustrated the versatility of their response variables by assigning labels to different type of classification tasks. They used nine different datasets from UCI datasets and showed that their results were competitive to conventional classification algorithms such as naive Bayes and SVMs but in most cases they did not perform as well as them.

2.3 Using Unsupervised Topic Models for Classification

This section differs from the previous section titled "Topic Models for Supervised Learning." This section explores innovative ways of using unsupervised topic models, such as LDA and pLSA, for classification purposes. The previous section, on the other hand, explored some new topic models, themselves, that learn in supervised setting. Surprisingly, the field of using the output of an unsupervised topic model, such as LDA, for classification has not received as much attention as the development of new supervised topic models. Our research, in this dissertation, specifically uses LDA and pLSA for classification of the text documents across domains. We show encouraging experimental results to strengthen the idea that research in using unsupervised topic models for classification deserves more attention than just the research on new topic models. This section will present some papers that use LDA, which is an unsupervised topic model, for classification.

Blei et al., while introducing Latent Dirichlet Allocation (LDA) model, showed the effectiveness of their model by comparing a perplexity measure against pLSA [11]. Perplexity measure can be interpreted as the likelihood of the previously unseen documents. The exact formulation of perplexity measure for M documents from the test set D_{test} is shown in the equation below:

$$perplexity(D_{test}) = \exp\{-\frac{\sum_{d=1}^{M} \log p(\mathbf{w_d})}{\sum_{d=1}^{M} N_d}\} \tag{2.6}$$

where N_d is the number of words and $p(\mathbf{w_d})$ is the word probability vector for document d. Authors showed that LDA model outperformed the pLSA model by assigning more likelihood to the held out documents that belonged to the same set as the one used for training the model. The authors concluded that this was due to the fact that the LDA model uses Dirichlet priors over the topic distribution instead of learning the topic distribution directly from the documents only. This makes LDA less biased towards the training set and results in a more generalizable model. In their second experiment, they did use LDA for classification to show its usability for classifying documents. Their experiments were extended in another paper by Li et al. that studied using LDA for classification [38].

Li et al. laid out empirical results of text classification using LDA [38]. Their approach was the same as Blei et al., however, their paper focused on the empirical results of using LDA for text document classification. The authors used the topic distribution vectors, gamma parameters, as the feature vectors for the documents. Gamma represents the topic distribution over a document; it can also be seen as a lower dimensional representation of a document in terms of the topics learned using the LDA. They used SVM as the classifier for these gamma feature vectors and Reuter's dataset for their experiments. Their results showed that tf.idf formulation for the feature vector does better than the LDA gamma vectors. LDA based feature set performed better only if less than 10% of the training set is used.

2.4 Methods for Cross-domain Knowledge Transfer

There are recent papers that try to define and tackle the problem of cross-domain classification. The definition of the cross-domain classification, however, varies from paper to paper. It is important to clarify these differences in the definition of *domains* as used by different researchers. In our case, the distinction between two domains is due to the difference in the underlying writing style. Examples of different domains in our research are news articles, e-mails and Wikipedia articles etc. Our definition of domains closely resembles the concept often described by the word genre.

Dai et al. defined the domains as two similar topics but from the same source of text [24]. Although this definition is significantly different from ours, it is still an interesting research paper due to the fact that they also used documents from one set to classify documents from another set. The two domains created by the authors did not differ in terms of their genre of the text, but they differed in terms of the subject of the text. They used 20-newsgroups, SRAA [39], and Reuter's datasets. As an example, after splitting the 20-newsgroups dataset into two domains, one set contained documents on subject *recreation.hockey* and another set contained documents on *recreation.baseball.* While splitting the Reuters dataset [35] in different domains, they selected the documents in *orgs* class as one domain and documents in *people* class as the second domain. As can be observed, these different subjects that are split across different domains, share a common, broader underlying topic.

Another similarity of this paper to our research is that they also compared and showed an improvement in classification accuracy over more conventional machine learning methods such as naive Bayes and SVMs etc. Their algorithm used co-clustering algorithms presented by Dhillon et al. [40]. The co-clustering algorithm, as described in Dhillon's paper, works on a *word count matrix*, where each row represents a document and each column represents the words. Each entry in the matrix, thus, represents the counts of the words in that particular document. Co-clustering algorithms groups these rows and columns together simultaneously into pre-determined number of clusters. Assuming that the row values are from a random variable X and column values are a random variable Y, Co-clustering algorithm seeks to find the following mapping, that is grouping X values into k clusters and Y values into l clusters:

$$C_X : \{x_1, x_2, ..., x_m\} \rightarrow \{\hat{x}_1, \hat{x}_2, ..., \hat{x}_k\}$$
$$C_Y : \{y_1, y_2, ..., y_n\} \rightarrow \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_l\}$$

Once we define $\hat{X}$ as $C_X(X)$, and $\hat{Y}$ as $C_Y(Y)$, the goal of co-clustering algorithm becomes finding the mapping that minimizes the mutual information loss between the (X, Y) and $(\hat{X}, \hat{Y})$.

Dai et al. extended Dhillon's co-clustering algorithm by adding the minimization from the labeled clusters obtained from the documents in the labeled domain [24]. Following is the cost function used by the authors to perform the co-clustering:

$$I(D_0; W) - I(\hat{D}_0; \hat{W}) + \lambda \cdot (I(C; W) - I(C; \hat{W})) \tag{2.7}$$

where I is the mutual information function. D_0 is the unlabeled domain documents, W is the word counts, $\hat{D}_0$ and $\hat{W}$ are the co-clustering of these documents and words, and C is the labeled clusters obtained from the labeled domain documents. λ is simply a weighting parameter that determines the weight of cost function part that depends on the labeled set of documents i.e. $I(C; W) - I(C; \hat{W})$. In the paper, they equated minimization of the above expression with the minimization of the KL divergence ($D(\cdot \,||\, \cdot)$) between the distributions as follows:

$$\begin{aligned} &I(D_0; W) - I(\hat{D}_0; W) + \lambda \cdot \Big(I(C; W) - I(C; \hat{W})\Big) \\ &= D\left(f(D_0, W)||\hat{f}(D_0, W)\right) + \lambda \cdot D\left(g(C, W)||\hat{g}(C, W)\right) \end{aligned} \tag{2.8}$$

where $\hat{f}$ and $\hat{g}$ are the joint probability distributions after the co-clustering. Their method showed a significant improvement in cross-domain classification accuracy for both SRAA and 20-newsgroups dataset but did not show a significant accuracy improvement for Reuters dataset.

Wang et al. also used the same domain definition as authors in [24], however, the Reuters dataset is not used in their paper [19]. Similar to the paper by Dai et al., the 20-newsgroups data is split into two *domains* by using some newsgroups as the first domain and some similar newsgroups as the second domain. The methods in their paper augmented the feature vectors using the Wikipedia based features [19, 41]. The empirical results showed an increase in accuracy for all pairs of domains. They did not discuss the number of features that were added to achieve this increase in the accuracy.

Swarup et al. addressed the cross-domain classification problem and motivated this problem from a similar point of view as ours [42]. The premise was that since humans are capable to transferring psychological or neurological concepts from one domain to another, a machine learning algorithm should be able to do the same. However, this paper lacked any formal constructs to their approach and any robust empirical results. This paper was motivated by a conceptual cross-domain learning framework based on human learning. The *structural representations* used for cross-domain learning are made from multiple neural networks and corresponding *genomes* to weight the neural networks differently in the presence of different domains. Although this paper did not contain any formal results or proofs, it presented a new direction for tackling the cross-domain learning by developing structures that can mimic so-called *learning-to-learn* or accomplish *knowledge transfer* across domains.

2.5 Topic Models for Cross-Domain Classification

Xue et al. introduced a model titled *Topic-bridged pLSA* that uses a modified probabilistic latent semantic analysis (pLSA) algorithm to generate topic models for cross domain text classification [43]. Authors applied this extension of the model to *bridge knowledge* across domains. This is an interesting extension of pLSA; specifically for cross-domain classification. Their definition and experiments used the same datasets as [19] and [24] and also used the same definition of domains. The authors' model, however, was not dependent on this definition of the domains and therefore can be applied to any two sets of documents, as long as only one of the sets is labeled.

The authors' model extended the basic pLSA model in two ways. Firstly, they split the likelihood method in two different expressions, one for the likelihood of the labeled set (d_l) and one for the likelihood of the unlabeled set (d_u). This way, the user can weigh the model differently to fit labeled or unlabeled data. The weight of the two sets of documents is determined by a weighting parameters, λ. The combined likelihood was obtained from the following equation:

$$\begin{aligned} L = \sum_w \Bigg[\lambda \sum_{d_l} n\left(w, d_l\right) \log \sum_z \Pr\left(d_l|z\right) \Pr\left(z|w\right) \\ + \left(1-\lambda\right) \sum_{d_u} n\left(w, d_u\right) \log \sum_z \Pr\left(d_u|z\right) \Pr\left(z|w\right) \Bigg] \end{aligned} \tag{2.9}$$

Secondly, they incorporated the label information from the labeled set of documents. This is accomplished by adding penalties for mismatch between the topics assigned and the document labels. These penalties include assigning different topics to documents under the same label and assigning same topics for documents under different labels. These penalties are written out in the following equations:

$$f_s(d_l^i, d_l^j) = \log \sum_z \Pr(d_l^i|z) \Pr(d_l^j|z) \tag{2.10}$$

$$f_d(d_l^i, d_l^j) = \log \sum_{z_i \neq z_j} \Pr(d_l^i|z_i) \Pr(d_l^j|z_j) \tag{2.11}$$

where $Pr(d_l^i|z)Pr(d_l^j|z)$ represents the probability that two documents d_l^i and d_l^j generated by same topic z. The combined likelihood of the model then becomes:

$$L_c = L + \beta_1 \sum f_s(d_l^i, d_l^j) + \beta_2 \sum f_d(d_l^i, d_l^j) \tag{2.12}$$

where β_1 and β_2 are again the weighting parameters for the two types of penalties. The experimental results did show an improvement over the algorithms such as SVM and NB. The authors compared the classification accuracy obtained from classifying documents where the training and test set belonged to two different domains. Such a classification is particularly challenging task for SVM and NB. Therefore an improvement in the classification accuracy over SVM and NB classifiers may not be very difficult to achieve.

Zhai et al. approached cross-domain learning problem from yet another point of view [44]. The authors tackled the problem of finding *common themes* among different domains. Marx et al. and Sarawagi et al. have also discussed a similar problem in their papers, [45, 46], respectively. The problem of finding common themes is an interesting problem in its own right, but it is also useful to us since the results of their paper can be extended to other problems such as the cross-domain classification problem, as we define it in this dissertation.

Here we go over the formulation presented in [44] in more detail; even though their goals are different, the formulation resembles our approach for cross-domain classification to a certain degree. The authors modified a unigram mixture model to a new generative model that helps in finding the common themes across different collections. Unigram mixture model, defined in Equation 2.13, assumes that all the words in the set of documents belong to a mixture of topics. To generate a document, **w**, under a unigram mixture model, one first picks a topic, z, from the mixture of topics and then iteratively chooses a word from the distribution associated with that topic.

$$p_d(\mathbf{w}) = \sum_z p(z) \prod_{n=1}^{N} p(w_n|z) \tag{2.13}$$

In their paper, Zhai et al. first extended the model by adding a common background theme to their collection. This is represented by θ_B in the following equation:

$$p_d(w) = \lambda_B \cdot p(w|\theta_B) + (1 - \lambda_B) \sum_{j=1}^{k} [\pi_{d,j} \cdot p(w|\theta_j)] \tag{2.14}$$

where θ_B is assumed to be the theme that represents noise or general words. The authors used the word *theme* to mean what would be known as a *topic* under the topic model vocabulary. In their model, all the other words are picked from j different themes. This formulation can be seen as an extension of the unigram model that encourages all the general words to be put into a common background theme, represented by θ_B. λ represents the weight of background versus specific themes.

The authors further extended this model and split the theme to better fit the framework of multi-domains. Each theme was split into a theme specific to a given domain and a theme common to all domains. With this extension they obtain the following formulation for unigram mixture models:

$$p_d(w|C_i) = (1 - \lambda_B) \cdot \sum_{j=1}^{k} [\pi_{d,j} (\lambda_C \cdot p(w|\theta_j) + (1 - \lambda_C) \cdot p(w|\theta_{j,i}))] + \lambda_B \cdot p(w|\theta_B) \tag{2.15}$$

In their experiments, the authors clustered news stories that contained information from the Iraq war and the Afghanistan war. As expected from their model, they obtained themes that were split into different groups based on their prevalence across domains i.e. some themes were common to all domains while others were specific to a single domain. In particular, they showed that some of their themes contained words that are common to all news stories, some contained words that were common to only war stories, and some themes were specific to only Iraq war or Afghanistan war. The authors were successful in extracting common themes across different domains, which is a goal that is closely related to our own research.

2.6 Conclusion

This chapter discussed the related work that included (1) classification of text documents using Wikipedia, (2) topic models for supervised learning, (3) using unsupervised topic models for classification, (4) methods for cross-domain knowledge transfer and (5) topic models for cross-domain classification. First section went over the ways of using Wikipedia as a tool to aid in text document classification. Second and third sections discussed role of topic models in classification from two points of view. Firstly (in Section (2)), we discussed topic models that incorporate the labeled data to learn a generative model, thus these topic models work in a supervised learning framework. Secondly (in Section (3)), we discussed the ways of using unsupervised topic models for classification purposes. Section (4) and (5) concentrated on the problem of cross-domain document classification, where Section (4) discussed some of the general approaches to solve the problem and Section (5) presented ways of using topic models to solve the problem. As we can see, there has been wide variety of work done in the field of cross-domain classification, most of it, however, tackles a slightly different version of the problem of cross-domain classification than ours. The previous work related to cross-domain classification either defines the word *domain* differently or does not apply to document classification. We introduce new methods and datasets to classify documents in a domain without requiring any labeled document from that particular domain and only using labeled documents from another domain. Our work, in this sense, does not extend any one or two specific work done in the past but extends the field of cross-domain classification as a whole.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3: Background

This chapter introduces some of the general machine learning concepts that are used in this dissertation. Some of these concepts may already be familiar to the readers proficient in the field of machine learning such as support vector machines (SVMs) and Latent Dirichlet Allocation (LDA) topic model, while some other concepts discussed in this chapter are a little more obscure such as Bregman divergences and their relationship to exponential family of distributions. In this chapter, we will briefly outline some common classification algorithms, along with a discussion on EM-algorithm and some common topic models. We will then introduce the concept of exponential family of distribution and their relationship with Bregman divergences. We will also take a closer look at Dirichlet distribution (a member of exponential family) and KL-divergence (a member of Bregman divergences).

3.1 Classification Algorithms

Classification algorithms use supervised learning techniques to classify data into pre-determined categories. Each data point used by these algorithms is presented in the form of a multi-dimensional numerical vector, called a *feature vector*. For example, a text document can be made into a d-dimensional feature vector where i^{th} entry represents the count of the i^{th} word in the vocabulary. These can be seen in Figure 3.1. The classification algorithms use labeled data as a training set to learn the classification model and then apply this learned model to classify unlabeled data from the testing set. In this manner, classification algorithms can be seen as a mathematical function that maps data feature vectors to set of class labels, as written below:

$$X \in \mathbb{R}^d$$
$$Y \in \{1, 2, \ldots, k\}$$
$$f : X \to Y$$

where X is a d-dimensional real valued feature vector. Y is a set of classification labels of integers 1 to k. f is the classifier function that maps X to Y. In this section, we will go over some of the common classification algorithms and their corresponding classifier functions, f.

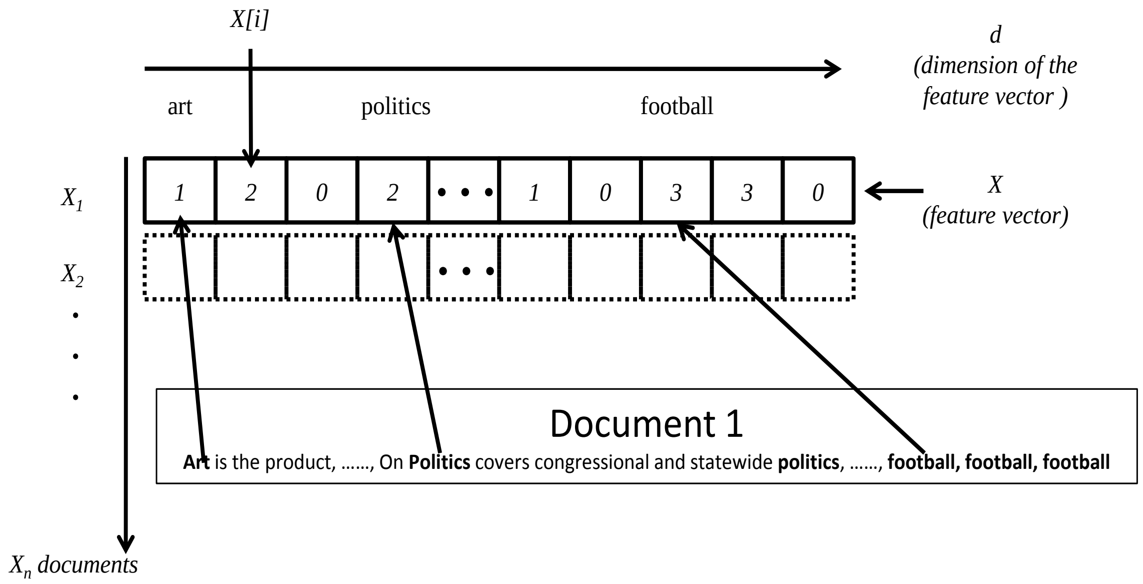

X[i]
d
(dimension of the feature vector)
art
politics
football
X₁
X₂
1 2 0 2 • • • 1 0 3 3 0
X
(feature vector)
Document 1
Art is the product, ……, On Politics covers congressional and statewide politics, ……, football, football, football
Xₙ documents

Figure 3.1: A text document can be made into a d-dimensional feature vector where i^{th} entry represents the count of the i^{th} word in the vocabulary.

3.1.1 Naïve Bayes

Naïve Bayes is one of the simplest and widely used classifier, especially for text documents. It computes the probability of a document belonging to a specific class given the counts of the words in the document. Naïve Bayes is named naïve so since it "naïvely" assumes the independence among the word counts in a document. It treats the document as "bag of words," by not incorporating any contextual or semantic information in the feature vector it uses. The impact of some of the assumptions commonly made in a typical naïve Bayes classifier have been a topic of debate and many improvements have been proposed in the literature from time to time [47, 8]. However, even with these assumptions, the classifier generally performs very well in practice [48]. The naïve assumption that all features are independent simplifies the posterior probability equations. Under this assumption, one can compute the posterior probability by using only class prior probabilities and probability for each of the features independently. The details of the formulations used in this classifier are described in the following paragraph.

Let's assume that there are D documents in our training set. Each of the document, d, in the collection can be represented by a feature vector, w, containing the counts of each word in the vocabulary. The dimension or the length, v, of this feature vector is the size of the entire vocabulary and the entry $w[i]$ represents the number of times i^{th} word appears in this document. With this information, we can compute the prior probabilities of each feature (or word) given a class. This probability can be computed by simply counting the number of times that word occurs in the documents of that class from the training set. Given the independent assumption among the words, the probability of a document D given the class C is:

$$p(D|C) = \prod_i p(w_i|C) \tag{3.1}$$

where $p(w_i|C)$ is the probability of the i^th word given the class C. Using the Bayes Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.2}$$

we can compute the probability of class C given the document D, $P(C|D)$, as follows:

$$P(C|D) = \frac{P(D|C)P(C)}{P(D)} \tag{3.3}$$

Using the above formulation, given a feature vector $\mathbf{x} \in \mathbb{R}^v$, containing word counts and a set of j different categories $Y = \{1, \ldots, j\}$, the classifier assigns category to the document x that maximizes the probability as shown below:

$$\mathrm{f}(x) = \operatorname*{argmax}_j P(Y_j|x) = P(x|Y_j)P(Y_j) \tag{3.4}$$

f is the classifier function that is being used to map $\mathbf{x} \in \mathbb{R}^v$ onto Y.

3.1.2 K-Nearest Neighbors

The k-nearest neighbor is also a supervised learning algorithm. It is one of the simpler and widely used machine learning algorithms that is similar to naïve Bayes. The object is to classify a given d-dimensional point $\mathbf{x}$ into a class $j \in Y$. The algorithm is explained in detail in the next paragraph.

Let's assume that the algorithm is given a training set of n documents represented as d-dimensional feature vectors that belong to c different classes. To classify an unlabeled document, represented as vector $\mathbf{x}$, one first computes $\mathbf{x}$'s distance to each one of points in the training set. She then selects the k-nearest neighbors of $\mathbf{x}$ in the training set. The label assigned to $\mathbf{x}$ is the label of majority of its k-closest neighbors in the training set. If $k = 1$, $\mathbf{x}$ is assigned the same class as the class of its nearest neighbor in the training set. To compute the distances from the test point $\mathbf{x}$ to the points in the training set, any arbitrary distance measure such as Euclidean distance, Cosine similarity measure and KL-distance measure etc., can be used. The choice of the parameter k can also depend on the size of the training set, number of classes and can be adjusted based on cross-validation results. In our dissertation, unless otherwise specified, we choose the k to be 31.

The classifier function, f, can be obtained for k-nearest neighbor algorithm with k = 1 as follows. Given a feature vector $\mathbf{x} \in \mathbb{R}^d$, containing word counts and a set of j different categories $Y = \{1, \ldots, j\}$, and a distance metric s, the classifier assigns category to the document $\mathbf{x}$ that is equal to the category of its k closest neighbors, as shown below:

$$b = min_j\{s(m_j, x)\}$$
$$f(\mathbf{x}) = label(b)$$

where b is the nearest neighbor of $\mathbf{x}$ in the training set of documents. Function f is the classifier function to map $\mathbf{x} \in \mathbb{R}^d$ onto Y.

3.1.3 Support Vector Machines

Support Vector Machines (SVMs) were developed by Vapnik et al. [49] in 1992. SVMs are a type of linear classifiers similar to a single layered perceptron, however, SVMs find a separating hyperplane that has the maximum margin between the two classes of data points. Since their introduction, SVMs have proven to be very effective in wide array of classification tasks, especially in text classification [50]. A brief formulation of the cost function that SVMs minimize is described in the following paragraph.

Given a training set of instance-labeled pairs $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, l$ where $\mathbf{x}_i \in R^d$ and $\mathbf{y} \in \{1, -1\}^l$, the support vector machines (SVM) [51, 6] find the hyperplane, with normal vector $\mathbf{w}$, by optimizing the following function:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned} \tag{3.5}$$

This formulation results in the margin of the resulting hyperplane to be $\sigma = 1/\|\mathbf{w}\|$. The first constraint requires all training set data to be on the "right" side of the hyperplane except a few points that can be on the “wrong” side of the hyperplane within some slack value denoted by ξ. If the i^{th} training example lies on the“wrong” side of the hyperplane, we get the corresponding $\xi_i \geq 1$. Because the term $C\sum_{i=1}^{n}\xi_i$ in the cost function that is to be minimized, C denotes a parameter that weighs the training data point errors for the final solution. In other words, C allows trading off training error vs. model complexity. The optimal value of this parameter is up to the user and is often chosen based on the results of cross-validation or by some other model selection strategy. In our experiments, unless otherwise stated, C is chosen to be equal to 1 for all cases.

Once the separating hyperplane $\mathbf{w}$ and threshold b is determined, we can use the following classifier function, f:

$$f(\mathbf{x}) = sgn\{\mathbf{w} \cdot \mathbf{x} + b\} \tag{3.6}$$

f is used to map $\mathbf{x} \in \mathbb{R}^d$ onto Y.

3.2 EM Algorithm and Topic Models

This section introduces Expectation-Maximization (EM) algorithm and two common topic models. EM is a common soft-clustering algorithm, where the parameters of the underlying distribution are determined such that the expectation (likelihood) of the observed is maximized. The underlying distribution is often assumed to be a mixture of multiple distributions e.g., mixture of Gaussians, mixture of multinomials etc. Topic models are mixture models, where documents are assumed to be generated from a mixture of distributions. Most topic models use the EM algorithm to find the parameters of the unobservable underlying distributions. There are various different topic models that have been proposed in the literature such as probabilistic Latent Semantic Analysis (pLSA) and Latent Dirichlet Allocation (LDA) etc. These topic models differ in their underlying assumptions about the type of distributions, the role of prior probabilities etc. In topic models, a topic is defined as a distribution over the words in the vocabulary. Following subsections will introduce these three concepts in more detail.

3.2.1 Expectation-Maximization Algorithm

Expectation-Maximization (EM) algorithm is often the algorithm of choice for finding parameters of a distribution that maximizes the likelihood of the observed data. Let X be a set of multidimensional vectors representing the observed data, and let θ be the parameters of the distribution of X. The goal of the EM algorithm is to find the θ that maximizes the likelihood of X. For the sake of convenience, we choose to maximize *log likelihood function* which is defined as follows:

$$L(\theta) = \ln P(\mathbf{X}|\theta) \tag{3.7}$$

In some trivial cases, the estimation of the data that maximizes this likelihood has close-formed solution. For example for an observed data that is assumed to be distributed according to single normal distribution having two parameters, μ and σ^2, the optimal value of μ is simply the average of X, and the optimal value of the σ^2 is the variance of X. However, it is often the case where the assumed underlying distribution is not as simple as single Gaussian but consists of a mixture of distributions. To find parameters of such a mixture of distributions, we introduce hidden variables to make the maximum likelihood estimation of θ tractable. We represent these hidden variables by $z \in Z$.

The joint distribution of X and z is

$$P(\mathbf{X}, z|\theta) = \prod_i P(\mathbf{x}_i|z, \theta)P(z_i|\theta) \tag{3.8}$$

The likelihood expression becomes:

$$P(\mathbf{X}|\theta) = \prod_i \sum_z P(\mathbf{x}_i|z_i, \theta)P(z_i|\theta) \tag{3.9}$$

Since we are interested in maximizing the log-likelihood, we take the log of 3.9 as follows:

$$L(\theta) = \log(P(\mathbf{X}|\theta) = \sum_i \log \sum_z P(\mathbf{x_i}|z_i, \theta)P(z, \theta) \tag{3.10}$$

This is not an easy equation to maximize (given the log of sums), therefore we introduce a distribution on $z_i's$, $q(z_i)$, as follows:

$$L(\theta) = \sum_i \log \sum_z q(z_i) \frac{P(\mathbf{x_i}|z_i, \theta)|P(z_i, \theta)}{q(z_i)} \tag{3.11}$$

$$L(\theta) \geq \sum_i \sum_z q(z_i) \log \frac{P(\mathbf{x_i}, z_i|\theta)}{q(z_i)} \tag{3.12}$$

the final inequality in the previous is reached using the following equation along with Jensen's inequality.

$$\sum_{z_i} q(z_i)\frac{P(\mathbf{x_i}, z_i|\theta)}{q(z_i)} = E\left[\frac{P(\mathbf{x_i}, z_i|\theta)}{q(z_i)}\right] \tag{3.13}$$

We apply the **Jensen's inequality** that states for a concave function f:

$$f(E(x)) \geq E(f(x)) \tag{3.14}$$

Since $\log$ is a concave function, we obtain this lower bound on $L(\theta)$.

We further show that $q(z_i)$ can be made equal to $L(\theta)$ for a fixed θ_n by setting

$$q(z_i) = P(z_i|x_i, \theta_n) \tag{3.15}$$

After showing that $q(z_i)$ is a lower bound on $L(\theta)$ and is equal to $L(\theta)$ at θ_n, we maximize $q(z_i)$ with respect to θ

$$\theta_{(n+1)} = \arg\max_{\theta} \sum_i \sum_z q(z_i)\frac{P(\mathbf{x_i}, z_i|\theta)}{q(z_i)} \tag{3.16}$$

3.15 and 3.16 are called the *E-step* and *M-step* of the EM algorithm respectively. EM algorithm maximizes the likelihood, $L(\theta)$, by repeating the E-step and M-step alternatively, and thus iteratively, obtaining next best parameter θ_{n+1} from θ_n.

EM algorithm forms the basis of many algorithms that require optimizing parameters of a distribution with hidden variables. The next two sections will introduce two topic models that also use EM algorithm to find the parameters of the underlying distribution described by a Bayesian network having hidden variables.

3.2.2 Probabilistic Latent Semantic Analysis

Probabilistic latent semantic analysis (pLSA) was proposed in 1999 by Hoffman et al. [52]. pLSA is a mixture model that assigns multiple topics to a single document. Each document is assumed to be generated from multiple topics. To generate each word in a document under pLSA, one first picks a topic from the set of topics and then generates the word according to the multinomial distribution as described by that topic. The probability of a topic is also dependent on the document itself and is written as $p(z \mid d)$, where z is the topic and d is the document. This formulation results in the following equation for the joint distribution of the document and

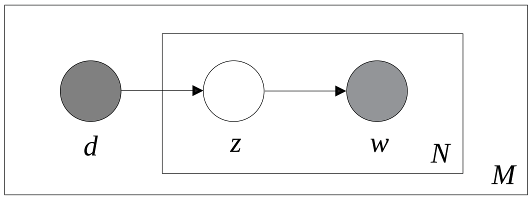


Figure 3.2: pLSA represented as a Bayes network using the plate notation

the words.

$$p(d, w) = p(d) \sum_{z} p(w_n \mid z) p(z \mid d) \tag{3.17}$$

where $z \in Z$ is a multinomial distribution, in other words z is a particular topic. In pLSA, each document has a topic distribution over Z associated with it. Probability of w_i, the i^{th} word, is computed by adding up the probabilities of that word in different topics, weighed by the probability of each topic in the document. Thus to generate a new document under pLSA, we need a distribution of the topics, which are associated with old documents. Therefore, pLSA will generate a new document that is similar to one of the given documents in the training set. In this way, pLSA can be seen as a pseudo-generative model i.e. it can only generate a new document based on a document from the training set. Figure 3.2 shows a plate notation diagram for the pLSA model.

3.2.3 Latent Dirichlet Allocation

Latent Dirichlet Allocation(LDA) is similar to the pLSA as discussed earlier. It is also a generative model for words in documents. In LDA, the model learns k topics, where k can be any integer. Since a topic is defined as a distribution over the words, the k topics, learned by LDA, are k different multinomial distributions of the words in the vocabulary. The vocabulary is a set of all the words used in the training set of documents. LDA does tend to learn clustering of words that form semantic themes (words describing the same concept will be assigned to the same topic). A common measure to compare topic models is the likelihood of the held out document set, known as perplexity measure. In the case of LDA, Blei et al. show that the perplexity measure is higher in a model learned using LDA compared to pLSA [11]. They hypothesize that this increase in the perplexity may be due to the fact that LDA has a Bayesian prior over the word distribution over topics. Lack of this prior in pLSA makes it over-learn the training set, thus reducing the perplexity of the new set in the model given by pLSA.

Given a document as a collection of words, LDA distributes them over k different topics and represents a document as a proportion of these topics. LDA model can be described by two parameters, namely, α and β. α is the parameter for the Dirichlet distribution that is used to generate the topic distributions for individual documents, β is a set of multinomial distributions over the words for individual topics. Given the model parameters $\{\beta, \alpha\}$, to generate a document with N words $\mathbf{W} = \{w_1, w_2, \dots, w_N\}$ under LDA, we take the following generative process:

- sample topic proportion $\theta \sim \text{Dirichlet}(\alpha)$,
- For each word $w_n \in \{w_1, \dots, w_N\}$,
  - sample a topic, z_n from θ,
  - sample the word, w_n from the set of multinomial distributions β for the topic z_n

The Equation 3.18 gives the expression for the probability of $\mathbf{w}$, where $\mathbf{w}$ is a word vector. Figure 3.3 shows the LDA model in the plate notation of the Bayes networks.

$$p(w \mid \alpha, \beta) = \int p(\theta \mid \alpha) \left(\prod_{n=1}^{N} \sum_{z_n} p(z_n \mid \theta) p(w_n \mid z_n, \beta) \right) d\theta \tag{3.18}$$

Equation 3.19 provides a more detailed version of Equation 3.18 by expanding the probability of θ, $p(\theta \mid \alpha)$, as obtained by the Dirichlet probability density function.

$$p(\mathbf{w}|\alpha, \beta) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta \tag{3.19}$$

Even though this is an unsupervised model, it can be used for classification purposes as it provides a different view of the documents as a distribution over different topics. One can use these k topics to compare different collection of documents. A document, in LDA, is represented as a distribution over these topics, one can generate such a representation for a set of documents, where some of the documents in the set are labeled and some are not labeled. The labeled set of documents can then be used to assign labels to the unlabeled set by training a classifier, such as SVM, on the topic distribution representations of all the documents.

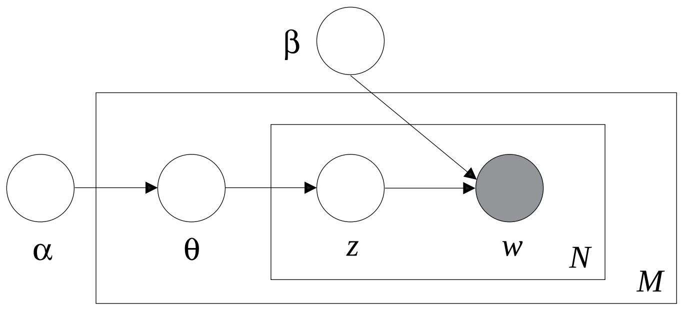


Figure 3.3: LDA represented as a Bayes network using the plate notation

In the next section, we will introduce the concept of exponential family of distribution, which is essential in formulating some of the distance metrics that we use in our classification algorithms.

3.3 Exponential Family of Distributions

In this section, we review the exponential family which is a set of distributions. Exponential families make up an important class of probability distributions. In addition to having some nice algebraic properties, they appear to be the natural distributions to consider. These families include both discrete as well as continuous distributions.

Definition: $P_{(\psi,\theta)}$ is an exponential family distribution, if it can be written in the following form. Let Γ be a convex set in R^d. $\theta \in \Gamma$ is the natural parameter.

$$P_{(\psi,\theta)}\mathbf{x} = \exp(\phi(\mathbf{x}) \cdot \theta - \psi(\theta) - \lambda(\mathbf{x})) \tag{3.20}$$

where $\psi(\theta)$ is a normalization constant which is differentiable on $int(\Gamma)$, and it is called *log-partition function*. $\phi(\mathbf{x})$ is called as *sufficient statistics* [53].

As an example, the Poisson distribution belongs to the exponential family. It is normally written as,

$$P(\mathbf{x}|\mu) = \frac{\mu^x e^{-\mu}}{x!}$$

where μ is the parameter of the Poisson distribution.

Table 3.1: Table showing some common exponential family distributions with their natural parameter and log-partition function.

$P_{\psi,\theta}(x)$	$\phi(x)$	θ	$\psi(\theta)$	Distribution
$p^x(1-p)^{(1-x)}$	$\langle x, 1-x\rangle$	$\langle \log p, \log(1-p)\rangle$	$\log\left(e^{\theta_1}+e^{\theta_2}\right)$	Binomial
$\frac{\lambda^x e^{-\lambda}}{x!}$	x	$\log\lambda$	e^{θ}	Poisson
$\frac{1}{\sqrt{(2\pi\sigma^2)}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$	$\langle x, x^2\rangle$	$\langle\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}\rangle$	$\frac{\mu^2}{2\sigma^2}+\frac{1}{2}\log(2\pi\sigma^2)$	Gaussian

The Poisson distribution can be reduced to the exponential family form as follows:

$$\begin{aligned} P(\mathbf{x}|\mu) &= \frac{\mu^x e^{-\mu}}{x!} \\ &= \frac{1}{x!}e^{x\log\mu}e^{-\mu} \\ &= e^{-\log(x!)}e^{x\log\mu}e^{-\mu} \\ &= e^{x\log\mu-\mu-\log(x!)} \end{aligned}$$

Following the exponential family form, we get

$$\begin{aligned} \theta &= \log(\mu) \\ \psi(\theta) &= e^{\theta} \\ \lambda(x) &= \log(x!) \end{aligned} \tag{3.21}$$

Table 3.1 lists some of these common exponential family distributions such as Gaussian, Binomial, Beta, Multinomial, and Dirichlet distributions, etc. and their characteristic parameters. Exponential family of distributions has a number of convenient properties. Here we list four of these important properties of exponential family of distributions. We will discuss these properties in detail as they relate to our research.

- The first important property of exponential family of distributions is the existence of conjugate priors. *The conjugate prior* of a distribution is another distribution over its parameters. In the case of exponential family, *the conjugate prior* is also a member of exponential family. Furthermore, given any member of the exponential family according to the Equation 3.20, the *conjugate prior* distribution can be expressed in the following form:

$$p(\theta|\alpha,\beta) = m(\alpha,\beta)\exp(\langle\theta,\alpha\rangle - \beta\psi(\theta)) \tag{3.22}$$

where α and β are hyperparameters of the conjugate prior. Importantly, the function $\psi(\cdot)$ is the same between the exponential family member and its conjugate prior.

- The second important property of the exponential family of distributions is bijection between exponential family members and Bregman divergences. This property means that each exponential family member, described by a convex log-partition function $\psi(\theta)$, has a corresponding Bregman divergence associated with it described by a convex function, ϕ. Furthermore, the two convex functions ψ and ϕ are Legendre conjugates of each other.

- The third important property of the exponential family of distributions is the *one-to-one* mapping between the *canonical parameters*, θ, and the so-called *mean parameters* which we denote by μ. For each canonical parameter $\theta \in \Theta$, there exists a mean parameter $\mu \in \mathcal{M}$, where $\mathcal{M}$ can be defined as:

$$\mathcal{M} := \left\{ \mu \in \mathbb{R}^d : \mu = \int \phi(x) p(x;\theta) dx \quad \forall \theta \in \Theta \right\} \tag{3.23}$$

  Furthermore, Θ and $\mathcal{M}$ are dual spaces in the sense of Legendre duality. In Legendre duality, we know that two spaces Θ and $\mathcal{M}$ are dual of each other if for each $\theta \in \Theta, \nabla\psi(\theta) = \mu \in \mathcal{M}$. This duality yields the following relationship between the log-partition function $\psi(\theta)$ and the sufficient statistics function of $\phi(x)$:

$$\nabla\psi(\theta) = \mathbb{E}(\phi(x)) = \mu \tag{3.24}$$

$$\nabla\phi(\mu) = \theta \tag{3.25}$$

- The fourth important property of the exponential family of distributions is that the exponential families arise naturally when we look for a maximum entropy distribution consistent with given constraints on the expected values. For example, for a non-negative random variable with an expected value of $-1/\lambda$, the maximum entropy distribution is the *exponential* distribution and for any random variable with a known mean and variance, the maximum entropy distribution is the normal distribution. We will now define the maximum entropy solution a bit more formally. The entropy of a random variable X is defined as:

$$H(X) = -\sum_X p(x) \log p(x)$$

When we find a probability distribution $p^*(x)$ that maximizes $H(x)$ while satisfying the following constraints on the expected values, we get a unique $p^*(x)$ which belongs to an exponential family of distributions [54].

$$\begin{aligned} \sum_X p(x) f_i(x) &= \alpha_i \\ \sum_X p(x) &= 1 \end{aligned}$$

where exponential family is defined previously as Equation 3.20.

3.3.1 Dirichlet Distribution

The Dirichlet distribution is a member of exponential family of distributions. It is a multivariate distribution over positive real numbers. Let $\alpha_1 \dots \alpha_k$ be parameters of Dirichlet distributions such that $\alpha_i > 0$ for $i = 1 \dots K$. A vector $(x_1, \dots, x_k)$, where $\mathbf{x}_i > 0$ for $i = 1 \dots K$ and $\sum_{i=1}^{K} = 1$ will be distributed according to Dirichlet distribution denoted as $(x_1, \dots, x_k) \sim D(\alpha_1, \dots, \alpha_k)$. Samples taken from a K dimensional Dirichlet distributions lie on a $K - 1$ simplex. The probability of vector $x_1 \dots x_k$ under Dirichlet distribution can be written as

$$P(x_1 \dots x_K) = \mathrm{Dir}(x_1 \dots x_k | \alpha) \tag{3.26}$$

$$= \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \qquad x_i > 0,\ \sum_i x_i = 1;\ \alpha_i > 0 \tag{3.27}$$

where α_i is the i^{th} element of the parameter vector α. The normalization constant $B(\alpha)$ is defined as follows

$$B(\alpha) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)} \tag{3.28}$$

The Γ function is the *gamma* function. It can be thought of as a factorial function extended to real numbers. The *gamma* function is defined as follows

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt \tag{3.29}$$

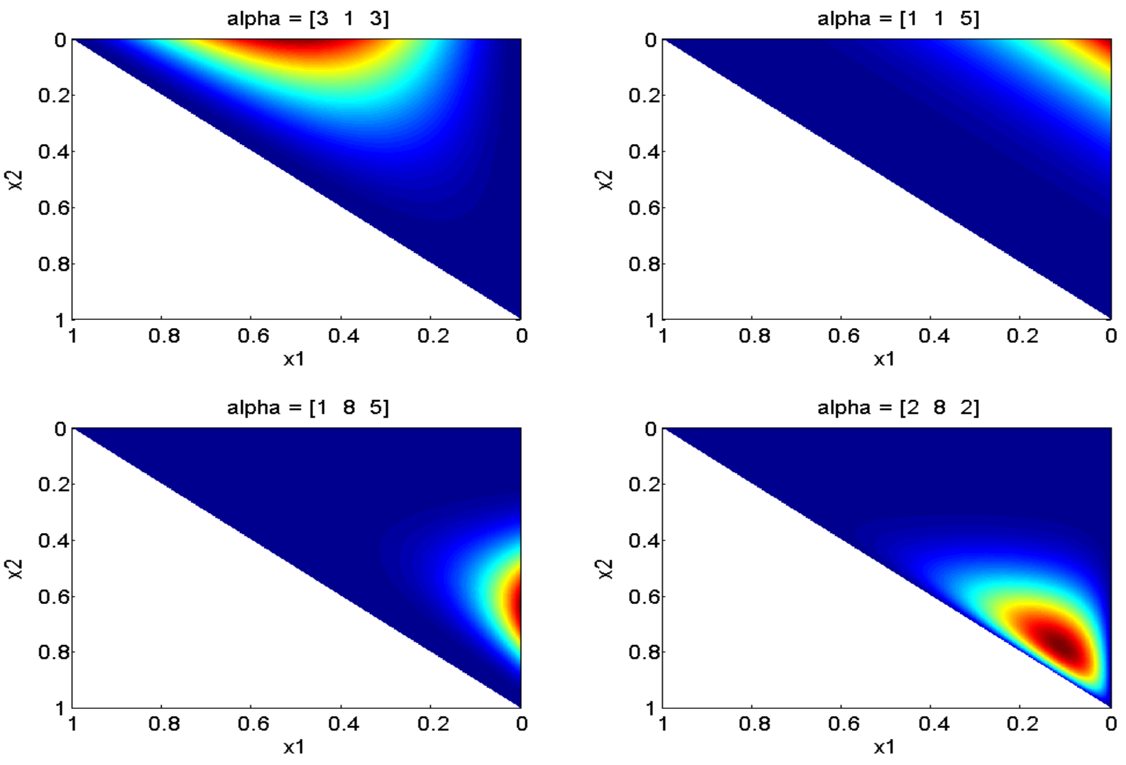


Figure 3.4: The contours of the Dirichlet distribution for different values of α.

The expected value and the variance of Dirichlet distribution is defined as follows:

$$E[x_i] = \frac{\alpha_i}{\alpha_0} \tag{3.30}$$

$$Var[x_i] = \frac{\alpha_i(\alpha_0 - \alpha_i)}{\alpha_0^2(\alpha_0 + 1)} = \frac{E[x_i](1 - E[x_i])}{\alpha_0 + 1} \tag{3.31}$$

where, $\alpha_0 = \sum_{i=1}^{K} \alpha_i$. From this, we can see that the variance of the Dirichlet distribution is inversely proportional to the α_0. Figure 3.4 shows the contours of the Dirichlet distribution for different values of α.

Relation Between Multinomial and Dirichlet Distribution

Dirichlet distribution is among the most important distributions because Dirichlet distribution is the conjugate prior for the parameters $\beta_1 \ldots \beta_K$ of a discrete multinomial distribution [55].

This property of Dirichlet distributions being the conjugate prior for multinomial distributions can be used to find the posterior parameter of a multinomial distribution given a prior and observed data. Specifically, if the prior over parameter of a multinomial distribution is given by α, then in the presence of observed data $x, \ldots, x_k$, we get the following relationships:

$$\begin{aligned} \beta &\sim Dir(\alpha) \\ x_1, \ldots, x_k | \beta &\sim Multi(\beta) \\ \beta | \mathbf{x} &\sim Dir(\alpha + \mathbf{x}) \end{aligned}$$

where $\mathbf{x}$ is the observed value vector, $x_1, \ldots, x_k$, and β is a parameter vector of the multinomial distribution [56].

Standard exponential family form of Dirichlet Distribution

It is easier to identify the natural and expectation parameter of Dirichlet distribution by writing it in its standard form. The Dirichlet distribution can be written in exponential family form as

$$\mathrm{Dir}(p_1 \ldots p_K | \mathbf{u}) = \exp\left(\begin{bmatrix} u_1 - 1 \\ \vdots \\ u_K - 1 \end{bmatrix}^{\mathbf{T}} \begin{bmatrix} \log p_1 \\ \vdots \\ \log p_K \end{bmatrix} + \log \mathbf{\Gamma}(\mathbf{U}) - \sum_{\mathbf{i=1}}^{\mathbf{K}} \mathbf{\Gamma}(\mathbf{u_i}) \right) \tag{3.32}$$

where $U = \sum_{i=1}^{K} u_i$. The expectation under P of the natural statistic vector is

$$\left\langle \begin{bmatrix} \log p_1 \\ \vdots \\ \log p_K \end{bmatrix} \right\rangle_P = \begin{bmatrix} \psi(u_1) - \psi(U) \\ \vdots \\ \psi(u_K) - \psi(U) \end{bmatrix} \tag{3.33}$$

where the digamma function $\psi(\cdot)$ is as defined previously.

Relation Between Gamma and Dirichlet Distribution

Dirichlet distributions also have relationship with other methods of exponential family of distributions such as *Beta* and *Gamma* distributions [57]. Gamma distribution can be used to generate samples from a given Dirichlet distribution. Gamma distribution has two parameters, k and θ, which are also known as *shape* and *scale* parameters respectively. The probability density function is defined as follows:

$$f(x; k, \theta) = x^{k-1} \frac{e^{\frac{-x}{\theta}}}{\theta^k \Gamma(k)} \qquad \text{for } x \geq 0 \text{ and } k, \theta > 0 \tag{3.34}$$

We obtain the following relationship between Dirichlet and Gamma distributions. If y_i's are independently distributed according to Gamma distribution with parameters α_i and θ for $i = 1, \ldots, k$, that is, $y_i \sim Gamma(\alpha_i, \theta)$ independently then

$$V = \sum_{i=1}^{k} y_i \sim Gamma(\alpha_0, \theta) \qquad \text{where } \alpha_0 = \sum_{i=1}^{k} \alpha_i \tag{3.35}$$

then

$$X = (x_i, \ldots, x_k) = \left(\frac{y_i}{V}, \ldots, \frac{y_k}{V}\right) \sim Dir(\alpha_i, \ldots, \alpha_k) \tag{3.36}$$

This property of Gamma and Dirichlet distributions is often used to collect samples from a given Dirichlet distribution.

3.4 Bregman Divergences

As we mentioned in the previous section, some common families of probability distributions – such as Gaussian, Binomial, and Poisson – are *exponential families*. This formalism has turned out to be very powerful in statistics and machine learning. This framework is general enough to include many distributions of interest (such as the distributions which factor over a specified undirected graph) while at the same time being specific enough that it implies all sorts of special properties. Furthermore each exponential family has a natural distance measure associated with it. In the case of spherical Gaussians, it is perhaps obvious that this distance measure is squared Euclidean distance, because the density at any given point is determined by its squared Euclidean distance from the mean.

As another example, in the multinomial distribution, it can be checked that the density of a point depends on its KL-divergence from the mean. Therefore, KL divergence is the natural distance measure of the multinomial. Notice that it is not a *metric*, i.e., it is not symmetric and does not satisfy the triangle inequality. However, as we will see, it is well-behaved in some ways and has a lot in common with squared Euclidean distance.

The various distance measures underlying different exponential families are collectively known as the *Bregman divergences* [58, 59]. We give the formal definition of these divergences, which does not follow the intuition about exponential families but rather associates each divergence with a specific convex function.

Definition: Let $\phi : S \rightarrow R$ be a strictly convex function which is defined on a convex domain

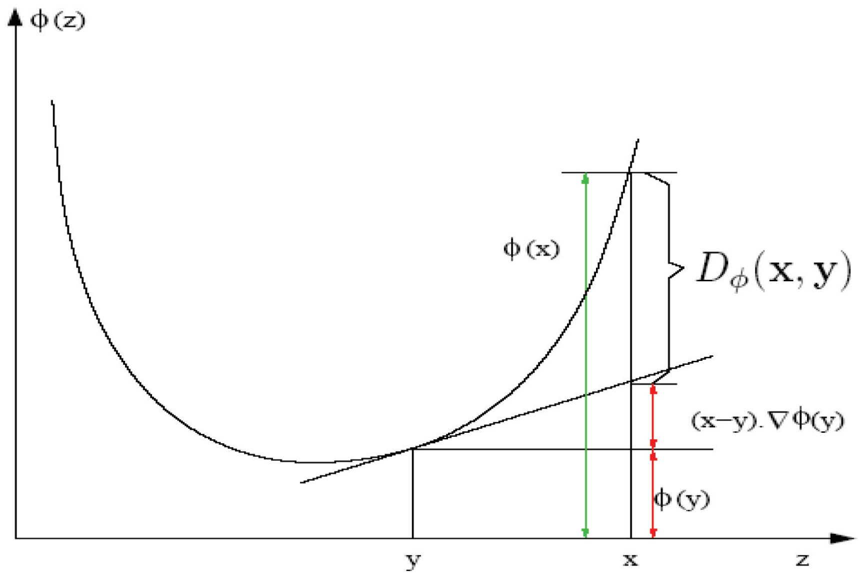


Figure 3.5: Bregman distance between points $\mathbf{x}$ and $\mathbf{y}$ gives us the first order approximation error when $\phi(\mathbf{x})$ is estimated using point $\mathbf{y}$.

$S \subset R^d$ and is differentiable on the interior of S. The Bregman distance $D_\phi : S \times \text{int}(S) \to [0, \infty)$ is then defined by

$$D_\phi(\text{x}, \text{y}) = \phi(\text{x}) - \phi(\text{y}) - \nabla\phi(\text{y}) \cdot (\text{x} - \text{y}) \tag{3.37}$$

A pictorial representation of Bregman distance between two points, $\mathbf{x}$ and $\mathbf{y}$, is shown in Figure 3.5. As mentioned earlier, some of the common distances, that are often used, belong to the family of Bregman distances. For example, choosing $\phi = \frac{1}{2}\|\mathbf{x}\|^2$ gives $D_\phi(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2$, which is the squared Euclidean distance. The derivation of this distance measure is shown below.

$$\begin{aligned} D_\phi(\mathbf{x}, \mathbf{y}) &= \frac{1}{2}\|\mathbf{x}\|^2 - \frac{1}{2}\|\mathbf{y}\|^2 - \mathbf{y} \cdot (\mathbf{x} - \mathbf{y}). \\ &= \frac{1}{2}\|\mathbf{x}\|^2 - \frac{1}{2}\|\mathbf{y}\|^2 - \mathbf{y} \cdot \mathbf{x} + \|\mathbf{y}\|^2 \\ &= \frac{1}{2}\|\mathbf{x}\|^2 + \frac{1}{2}\|\mathbf{y}\|^2 - \mathbf{y} \cdot \mathbf{x} \\ &= \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2 \end{aligned} \tag{3.38}$$

Similarly, $\phi(\mathbf{x}) = \sum_{i=1}^{d} x_i \log x_i$ gives

$$D_\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} x_i \log \frac{x_i}{y_i} - \sum_i x_i + \sum_i y_i, \tag{3.39}$$

which is a generalization of KL-divergence. It can be easily seen that this generalization reduces to the regular definition of KL-divergence when $\mathbf{x}$ and $\mathbf{y}$ are probability measures and therefore sum to one. The next subsection will discuss KL-divergence in a little more detail.

3.4.1 KL-Divergence: A Closer Look

KL-divergence is regarded as the "distance" between two distributions. For two distributions $p(x)$ and $q(x)$, the KL-divergence between them is defined as:

$$D_{KL}(p\|q) = \sum_X p(x) \log \frac{p(x)}{q(x)} \tag{3.40}$$

Notice that the KL-divergence between p and q is not symmetric and does not obey the triangle inequality. As we saw earlier, KL-divergence is a Bregman divergence. So it is always positive and equals to 0 only if $p = q$. Support for the KL distance as the true distance between two distributions comes from the concepts of information/coding theory. In coding theory, the entropy of a random variable X can be thought of as the average number of bits to represent it. $D_{KL}(p\|q)$ can be thought of as the number of bits that will be wasted by encoding events of p by using code that is optimized for q.

3.5 Bregman Divergences as Natural Distances for Exponential Distributions

In this section, we will describe the relationship between Bregman divergences and Exponential families of distributions more formally. We mentioned earlier that each exponential family has a natural distance measure associated with it, and that distance is a Bregman distance. The relationship clearly brings out how Bregman divergences are in fact the natural distance measures for exponential families. In essence, the probability density at any point x given by an exponential family distribution is directly proportional to some Bregman distance of x from the mean, μ. In other words, the density of an exponential family distribution, $P_{\psi,\theta}(x)$ can be written as

$$P_{\psi,\theta}(\mathbf{x}) \propto e^{-D_\phi(\mathbf{x},\mu)} \tag{3.41}$$

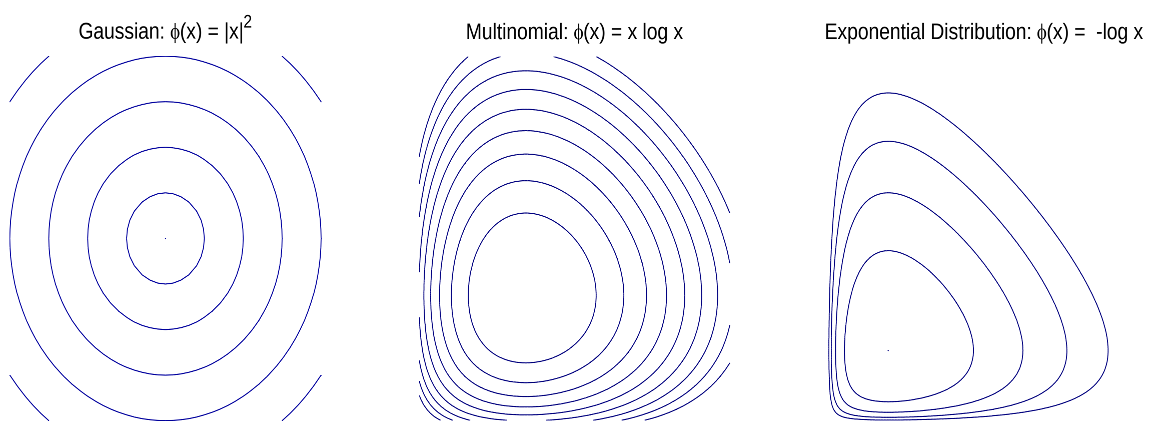


Figure 3.6: Distance contours of some common Bregman distances that also denote the shape of equal density contours for the corresponding exponential family.

where $D_\phi(\cdot,\cdot)$ is a Bregman divergence function. The relationship between functions ϕ and ψ is discussed further ahead in the section. Figure 3.6 shows some contours of a few common Bregman distances that also denote the shape of equal density contours for the corresponding exponential family.

We will now give a formal description of this relationship. A detailed analysis on this topic has been done in the "Clustering with Bregman Divergences" paper [59]. There is a bijection between exponential distributions and Bregman distances.

3.5.1 Bijection Relation

The bijection between exponential family and Bregman divergences is given as follows. The bijection theorem states that any exponential family distribution can also be expressed in the following form where μ is the expectation parameter, ϕ is a strictly convex function and $D_\phi(\cdot,\cdot)$ is the Bregman distance function.

$$P_{\psi,\theta}(\mathbf{x}) = f_\phi(x)e^{-D_\phi(\mathbf{x},\mu)} \tag{3.42}$$

f_ϕ is defined as

$$f_\phi(x) = \exp(\phi(\mathbf{x}) - \lambda(\mathbf{x})) \tag{3.43}$$

Every convex function has an associated convex function, know as its Legendre conjugate or simply conjugate [60]. A convex conjugate, f^c, of a convex function, f is defined as

$$f^c(x) = \sup_y \{xy - f(y)\} \tag{3.44}$$

This bijection between exponential distribution $P_{\psi,\theta}(\cdot)$ and Bregman distance $D_\phi(\cdot, \mu)$ is even more profound since convex functions ψ and ϕ are the Legendre Conjugates of each other, and $\mu = \nabla\psi(\theta)$, $\theta = \nabla\phi(\mu)$.

3.5.2 Examples of Bregman Distances as Natural Distances for Exponential Families

Some examples of exponential family distributions associated with commonly used Bregman distance functions are as follows,

(1) Euclidean distance: $\phi(\mathbf{x}) = \frac{\|\mathbf{x}\|^2}{2}$

$$\begin{aligned} D_\phi(\mathbf{x}, \mu) &= \frac{\|\mathbf{x} - \mu\|^2}{2} \\ P_{\psi,\theta}(\mathbf{x}) &= \exp(-\frac{\|\mathbf{x} - \mu\|^2}{2}) f_\phi(\mathbf{x}) \end{aligned}$$

We get the familiar Gaussian distribution. Thus Euclidean distance is the natural distance associated with the Gaussian distribution.

(2) KL-distance: $\phi(\mathbf{x}) = \sum_{i=1}^d x_i \log x_i$ gives

$$\begin{aligned} D_\phi(\mathbf{x}, \mathbf{y}) &= \sum_{i=1}^d x_i \log \frac{x_i}{y_i} \\ P_{\psi,\theta}(\mathbf{x}) &= \prod_{j=1}^d \left(\frac{\mu_j}{\mathbf{x}_j}\right)^{x_j} f_\phi(\mathbf{x}) \end{aligned}$$

Thus KL distance is shown to be the natural distance associated with the multinomial distribution.

3.6 Conclusion

This chapter goes over important background concepts that are utilized in our research. Some of these concepts are well known such as SVMs, KNN classifiers, EM algorithm and topic models such as LDA and PLSA. While some other concepts are not known as well such as Bregman divergences and their relationship to exponential family of distributions. In addition to properly laying out and introducing some of the important concepts of machine learning, this chapter also tries to weave together these concepts by showing the relationships among them. We will refer back to this chapter while describing some of our own algorithms that are developed as part of this dissertation.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4: Datasets for Cross Domain Document Classification

Since the cross-domain document classification is a relatively new area of research, it lacks a common dataset that researchers can use for their experiments. The field of cross-domain document classification needs a dataset that has large number of documents on various topics. Moreover these topics need to be common across multiple domains. For example, a dataset will require hundreds or thousands of documents, in each domain, on topics such "finance" and "music." Therefore, we developed these datasets for our experiments to specifically show the robustness and accuracy of our algorithms for cross-domain classification. We obtained our text documents from three different domains: (1) Wikipedia, (2) New York Times (NYT), and (3) 20-Newsgroups data set. All of these three domains contain a large number of documents on variety of topics. We extracted different sets of data, each consisting of its own group of topics. These topics were carefully chosen to reflect some real-world document categories and to ensure that the topic exists in all three domains. These datasets can also serve as a valuable resource for researchers working in this field of cross-domain document classification. In this section, we will go over the details of these datasets that are created for our research.

4.1 Wikipedia

Wikipedia provides a massive data source for research in the area of document classification. With over 16 GB of text data, over 8 million titles, and over 20 million page-category links [14] [61], extracting useful information from Wikipedia requires carefully developed data structures and optimized algorithms. To download Wikipedia articles, one could write a computer program to download the HTML pages directly from the Web and then to use packages such as "beautiful soup" [62] to strip off all the HTML tags to get the pure text. This type of approach will work well if one only needs to download a few articles or if the title of the article is known in advance. However, for a task that requires building an entire training set by getting thousands of articles under different categories, downloading individual pages using a Web API would have been too slow.

Our dataset was prepared by the original text files of the articles written in Wiki-script. Wikipedia provides text from all the articles at the following URL: *download.wikimedia.org/enwiki/latest/* in a zipped xml file. This link provides various other files such as files containing all the previous versions of every article. Our data of Wikipedia was obtained by downloading the xml file containing the current version of all Wikipedia articles, written in Wiki script, on Oct 30, 2009. The xml file is a large (20 GB) file containing tags for article title and article text. The xml file does not contain any explicit labeling or categorization of the articles. The tags used in the xml file do not contain the category information. The xml structure of the file is very simple, containing only two tags, one for the title of the article and second for the text of the article. All other information about the article, such as its categories etc. had to be inferred from the article text itself.

Each Wikipedia article can be assigned to multiple categories. Categories themselves can be put into different categories, thus creating a "hierarchy" tree of categories. In fact, what is produced by Wikipedia is not a tree but a graph with lots of cycles. An article is assigned to categories by putting a [[Category:]] tag in the article. For example, the tag [[Category:History]] in page x, will place page x under category "History." To see a list of all the pages under a category e.g., "History", in Wikipedia, one can look at a page titled "category:History." However, the contents of these pages, listing the contents of a category, are dynamically generated by using the category tags in various pages. This is similar to how the references are dynamically generated at the end of the page, by using the "ref" tags from the body of the article. We first collect the category-title information by browsing all the articles and collecting the "[[Category:]]" tags. We can then make a graph of the category hierarchy and easily access all the articles under a given category.

To be able to extract articles from a given category, we implemented some intermediate paging and indexing files for faster search through the large xml file. For the text mining related task, we need to obtain a set of articles under a specific category and use the category as the document class label. Since categories can have sub-categories, the pages under a given category may not appear directly under the category but may appear in one of its sub-categories. To collect all the articles under a given category, one can traverse the graph down from a category node to its sub-category nodes and collect all the articles along the way. Since this category hierarchy in Wikipedia is not strictly a *tree* structure, it can have loops. Figure 4.1 shows the hierarchical structure of the categories with the category "History" as the root node [63]. Therefore, there is a risk of ending up in an infinite loop while going from a parent node to its children nodes. The problem can be avoided by simply marking the articles as you collect them so you do not collect any duplicate articles and stop when all the articles under a given node have been collected. In practice, however, as we go down the category hierarchical structure, soon after the depth one, we start to diverge far from the original topic and start coming across cycles in the hierarchical structure. To avoid straying too far from the original category, we go down to depth of one while collecting articles under a given category. So to get a list of all the articles under the category "History," we look at all the articles under that category and its sub-categories. We do not go further deep into the categories as depth of one gives us most of the relevant articles under a category. Even though, in theory, one could go to depth of 3 or 4 in category hierarchy, the relevance of the articles from the original root category greatly decreases after depth 1 or 2. For example, the path, $Law \rightarrow Inheritance \rightarrow Caste \rightarrow Kaji(Nepal)$, shows how the later categories diverge from the topic of the root category, law. Wikipedia category network has over 8000 categories. These categories ranges from very specific topics, such as "TV cartoon characters of 1980s," to very broad such as "Science." We selected the categories on three criteria, (1) the categories were not too general or too specific and contained at least 200 articles in it, (2) the categories represented real-world topics, (3) the categories had a large number of articles in at least one other domain.

4.2 New York Times (NYT)

NYT dataset [64] is corpus containing nearly all the articles from New York Times from January 1987 to June 19 2007. The corpus contains over 1.8 million articles spanning over 20 years period. This consisted of on average 250 articles per day with maximum of 955 articles in a day. There were very few days in this corpus that had zero articles. The NYT corpus is very well organized and professionally created by the NYT staff. Each article is contained in its own xml file. These files are organized into folders according to date, with year, month and day making up different sub-folder levels. The xml file contains on average 50 tag values, that list various attributes of the article such as "date published," "author," "title," and "lead paragraph" etc. The main part of the article is written under the tag "body." The "body" tag itself is comprised of "body.head" and "body.content" sub tags. To prepare the dataset from NYT articles, we are mostly interested in a tag titled, "category" or "topic", that we could use to generate a labeled dataset. The corpus however does not contain a one single tag that can be used to describe the category or topic of the article. After analyzing all the xml tags that could serve as category, we made a list of the following seven potential candidates for the category tag 4.1.

"M" and "S" denote whether the field can have multiple values or not. For example, the entry in

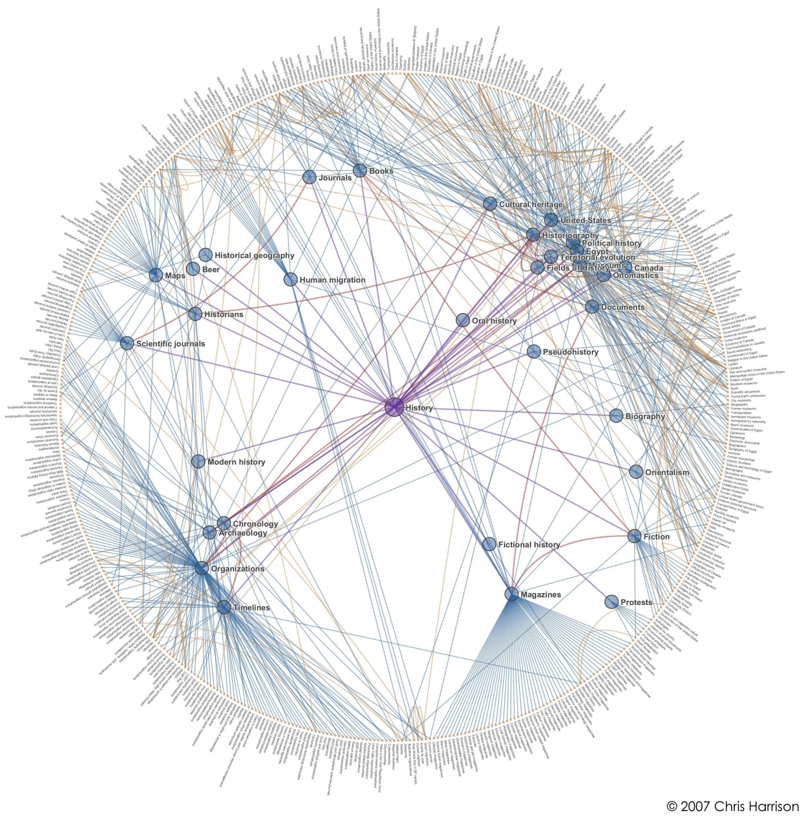


Figure 4.1: Figure showing the hierarchical network of categories in Wikipedia with the category "History" as the root node. Category structure is not strictly a tree and can have loops.

the first line of the Table 4.1 has a value of "M" that means in an article, on "Tom Brokaw," the tag "Biographical Categories" can have multiple values such as "journalism" and "television." Out of these seven possible choices for the category tags, we choose four different fields as labels for the document, namely, (1) Descriptors, (2) General Online Descriptors, (3) Online Descriptors, and (4) Taxonomic Classifiers. An article is assigned all the labels that occur in at least one of these chosen fields. Therefore, to gather the articles that belonged to category "opera," all the articles that had opera in at least one of these tags will be gathered. Table 4.2 shows all the categories in different groups as they are named in Wikipedia and New York Times.

Following five pairs of Tables from 4.3 to 4.12 outline the content of each dataset created from five groups, namely, (1) Arts, (2) Computers, (3) Current Affairs, (4) Science and, (5) Social Sciences. A group consists of two tables, the first table presents the counts of articles from each of the categories in that group, while the second table provides technical information about

Values	Tag Name	Description
M	Biographical Categories (ex: Books and Magazines)	category in which featured individual belongs to
M	Descriptors (ex: Data Processing)	descriptive terms corresponding to subject
S	Feature Page (ex: Education)	name of page article appears in
M	General Online Descriptors (ex: Research, Surfing)	general description
M	Online Descriptors (ex: Computers And the Internet)	descriptive terms corresponding to topics
S	Online Section (ex: Business; Technology)	name of the section placed under nytimes.com
M	Taxonomic Classifiers (ex: Top/News/Technology)	hierarchy of taxonomic descriptors

Table 4.1: List of seven potential candidates for the category tags in the XML files of the NYT dataset. "M" and "S" denote whether the field can have multiple values or not.

the dataset, such as the vocabulary size, average size of the documents etc. These datasets were collected from two domains, Wikipedia and New York Times (NYT) and the tables give information pertaining to both of these domains.

Table 4.3 presents results for the group "Arts," which includes categories such as "Film," "Literature," "Music" etc. Table 4.3 shows that the number of articles obtained from Wikipedia under the category "Film" is 997 and from NYT under the same category is 1618. Table 4.4 gives some more technical details of this dataset. The combined vocabulary size, with only the stop words removed, is 118,778. As one can see that this is a large vocabulary size and can be reduced. We reduce it by removing the rare words from the dataset and the words that appear only in one of the two domains. By removing the words that occur less than six times in the entire dataset, we almost reduce the vocabulary size by 50%. This can be seen by the second entry in the Table 4.4, which is 61,688. We show in our experiments that we do not get any significant decrease in the accuracy by removing all the rare words that occur less than 100 times in the entire dataset. This reduces the dimension by almost 90%. This can be seen from Table 4.4 in the third line that lists the vocabulary size obtained after removing the words occurring less than 100 times to be 10,373. Since by removing words that occur less than 100 times, we do not lose any classification accuracy, we (unless otherwise stated) use this dataset for our experiments. Table 4.4 goes on to show other attributes of the datasets, such as "Minimum number of words in a document" or words that appear less than two times in one of the domains (Words in domain less than two times), etc. The Tables 4.5 to 4.12 give the same details for other groups of categories.

Table 4.2: Groups of Categories

Group Name	**Categories**	**Category Name in NYT**
Arts (6)	Theatre	Theater
	Music	Music
	Opera	Opera
	Film	Motion Pictures
	Television	Television
	Literature	Books and Literature
Current Affairs (6)	Finance	Finances
	Military	Armament, Defense and Military Forces
	Terrorism	Terrorism
	Law	Law and Legislation
	Christianity	Christians and Christianity
	Islam	Islam
Science (6)	Genetics	Genetics and Heredity
	Space	Space
	Anthropology	Archaeology and Anthropology
	Medicine	Medicine and Health
	Chemistry	Chemistry
	Physics	Physics
Technology (5)	Software	Computer Software
	Electronics	Electronics
	Internet	Computers and the Internet
	Telecommunications	Telephones and Telecommunications
	Computer Security	Computer Security
Social (5)	Sociology	Sociology
	Linguistics	Language and Languages
	Economics	Economic Conditions and Trends
	Psychology	Psychology and Psychologists
	Law	Law and Legislation

Category	**Wiki**	**NYT**
Film	997	1618
Literature	1747	1798
Music	555	1618
Opera	3187	1780
Television	874	1702
Theatre	497	1771

Table 4.3: Number of documents in each category under "Arts" group as obtained from two different domains, Wikipedia and NYT

	Wiki	**NYT**
Combined Total Vocabulary Size	118778	118778
Common Vocabulary Size (words occuring at least 6 times)	61688	61688
Common Vocabulary Size (words occuring at least 100 times)	10373	10373
Other attributes of data after removing rare (less than 100 occurances) and unique words		
Minimum number of words in document	8	6
Average number of words in document	401	354
Maximum number of words in document	10851	4762
Documents with less than 50 words	726	752
Doucments with less than 20 words	99	209
Words in domain less than 2 times	6	34

Table 4.4: Some useful attributes of the dataset for the "Arts" group, including the feature vector dimension (size of the vocabulary) and documents sizes etc.

Category	**Wiki**	**NYT**
Computer Security	1069	944
Electronics	1227	866
Internet	864	1495
Software	924	1471
Telecommunications	1903	1624

Table 4.5: Number of documents in each category under "Computers" group as obtained from two different domains, Wikipedia and NYT

	Wiki	**NYT**
Combined Total Vocabulary Size	74733	74733
Common Vocabulary Size (words occuring at least 6 times)	38011	38011
Common Vocabulary Size (words occuring at least 100 times)	6970	6970
Other attributes of data after removing rare (less than 100 occurances) and unique words		
Minimum number of words in document	9	6
Average number of words in document	410	362
Maximum number of words in document	7596	3894
Documents with less than 50 words	555	451
Doucments with less than 20 words	43	75
Words in domain less than 2 times	3	84

Table 4.6: Some useful attributes of the dataset for the "Computers" group, including the feature vector dimension (size of the vocabulary) and documents sizes etc.

Category	**Wiki**	**NYT**
Christianity	900	1832
Finance	561	1645
Islam	1241	1655
Law	1370	1554
Military	636	1144
Terrorism	432	1113

Table 4.7: Number of documents in each category under “Current Affairs” group as obtained from two different domains, Wikipedia and NYT

	Wiki	**NYT**
Combined Total Vocabulary Size	93997	93997
Common Vocabulary Size (words occuring at least 6 times)	47805	47805
Common Vocabulary Size (words occuring at least 100 times)	9129	9129
Other attributes of data after removing rare (less than 100 occurances) and unique words		
Minimum number of words in document	10	7
Average number of words in document	588	394
Maximum number of words in document	9651	4430
Documents with less than 50 words	457	668
Doucments with less than 20 words	56	59
Words in domain less than 2 times	6	47

Table 4.8: Some useful attributes of the dataset for the “Current Affairs” group, including the feature vector dimension (size of the vocabulary) and documents sizes etc.

Category	**Wiki**	**NYT**
Anthropology	1533	1437
Chemistry	2845	1384
Genetics	2089	1753
Medicine	2288	1925
Physics	2719	581
Space	473	1797

Table 4.9: Number of documents in each category under "Science" group as obtained from two different domains, Wikipedia and NYT

	Wiki	**NYT**
Combined Total Vocabulary Size	119734	119734
Common Vocabulary Size (words occuring at least 6 times)	60981	60981
Common Vocabulary Size (words occuring at least 100 times)	11260	11260
Other attributes of data after removing rare (less than 100 occurances) and unique words		
Minimum number of words in document	7	7
Average number of words in document	449	386
Maximum number of words in document	10325	5617
Documents with less than 50 words	1325	723
Doucments with less than 20 words	200	101
Words in domain less than 2 times	8	135

Table 4.10: Some useful attributes of the dataset for the "Science" group, including the feature vector dimension (size of the vocabulary) and documents sizes etc.

Category	**Wiki**	**NYT**
Economics	1004	1944
Language	1901	1962
Law	1331	1946
Psychology	1173	488
Sociology	1985	420

Table 4.11: Number of documents in each category under "Social Sciences" group as obtained from two different domains, Wikipedia and NYT

	Wiki	**NYT**
Combined Total Vocabulary Size	102537	102537
Common Vocabulary Size (words occuring at least 6 times)	50839	50839
Common Vocabulary Size (words occuring at least 100 times)	9392	9392
Other attributes of data after removing rare (less than 100 occurances) and unique words		
Minimum number of words in document	6	12
Average number of words in document	534	416
Maximum number of words in document	13263	4756
Documents with less than 50 words	567	372
Doucments with less than 20 words	54	40
Words in domain less than 2 times	5	77

Table 4.12: Some useful attributes of the dataset for the "Social Sciences" group, including the feature vector dimension (size of the vocabulary) and documents sizes etc.

4.3 Newsgroups Dataset

The dataset containing text from 20 different newsgroups has been one of the most widely used datasets in the field of machine learning, specifically for text document classification. It is a relatively old dataset with one of its earliest uses dating back to 1995 [22]. Table 4.3 shows the topics of the 20 newsgroups used in this dataset. The dataset is made available in three different formats, with minor differences among them [65]. The original dataset contains total of 19,997 posts in 20 different newsgroups with almost 1000 posts from each newsgroup. The other two versions are created from the original dataset by removing some less important or confusing information from the dataset, specifically, (1) entries that were copied and posted in multiple newsgroups (duplicate posts) and (2) some extra header information such as path, follow up, date etc. We use one of these derived datasets that is known as 20news-18828. It contains, as the name suggests, 18,828 total posts and only the "from" and "subject" fields from the header. This dataset does not contain any duplicate posts, i.e., posts that appear in multiple newsgroups in the original dataset.

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

For our cross-domain document classification task, we had to find similar topics in other domains to create the dataset that can be used in our experiments. Table 4.3 shows the original 20 newsgroup topics. Out of the 20 original topics, we found 12 topics that occurred in our other two domains, Wikipedia and NYT. We prepared four different groups of categories, similar to our experiments with Wikipedia and NYT. These four groups of category were named "All, "Politics," "Rec" (for recreation) and, "Sci (for science)." Table 4.13 shows the categories in each one of these groups. The groups "Politics," "Rec" and, "Sci" are subsets of the group named "All."

Table 4.15 and 4.16 present details of the datasets related to category group "All." Group "All" includes 12 categories including "Automobiles," "Baseball," "Christianity" etc. Table 4.15 shows the number of articles obtained from the Wikipedia, NYT and Newsgroups under the category "Automobiles" as 780, 1922 and 988 respectively. Table 4.16 gives some more technical details of this dataset, in particular the size of the vocabulary of the dataset and some statistics on the document sizes. Each dataset is generated by using two domains at a time. It should be noted that this vocabulary size is the one that is obtained after removing the words that occurred less than 100 times in the dataset and the words that appeared only in one of the two domains. The vocabulary thus changes depending on which two domains are used. The Table 4.16 shows that the vocabulary size for the group "All" from the domain pair, Wiki-NYT, is 42,410 and from the domain pair Wiki-Newsgroups is 34,705 and from the domain pair NYT-Newsgroups is 32,357.

Table 4.13: Groups of Categories

Group Name	**Categories**	**Category in NYT**	**Category in Newsgroups**
All (12)	Automobiles	Automobiles	rec.autos
	Baseball	Baseball	rec.sport.baseball
	Christianity	Christians and Christianity	soc.religion.christian
	Cryptography	Computer Security	sci.crypt
	Electronics	Electronics	sci.electronics
	Gun	Gun Control	talk. politics.guns
	Hockey	Hockey, Ice	rec.sport.hockey
	Medicine	Medicine and Health	sci.med
	Middle East	Top\News\World\MiddleEast	talk.politics.mideast
	Motorcycles	Motorcycles, Motor Bikes, Motor Scooters	rec.motorcycles
	Politics	Politics and Government	talk.politics.misc
	Space	Space	sci.space
Politics (3)	Guns	Gun Control	talk. politics.guns
	MiddleEast	Top\News\World\MiddleEast	talk.politics.mideast
	Politics	Politics and Government	talk.politics.misc
Rec (4)	Automobiles	Automobiles	rec.autos
	Baseball	Baseball	rec.sport.baseball
	Hockey	Hockey, Ice	rec.sport.hockey
	Motorcycles	Motorcycles, Motor Bikes, Motor Scooters	rec.motorcycles
Sci (4)	Cryptography	Computer Security	sci.crypt
	Electronics	Electronics	sci.electronics
	Medicine	Medicine and Health	sci.med
	Space	Space	sci.space

Table 4.14: Table showing the categories in each one of the groups used in the cross-domain classification experiments using Wikipedia, NYT and Newsgroups as the three domains. Four different groups of categories were used to generate four different datasets. The groups are “All,” “Politics,” “Rec” and, “Sci.”

Category	Wiki	NYT	Newsgroups
Automobiles	780	1922	988
Baseball	552	1780	994
Christianity	912	1927	996
Cryptography	718	1212	991
Electronics	1454	966	981
Guns	63	957	910
Hockey	139	1814	998
Medicine	2383	1811	988
MiddleEast	849	1508	940
Motorcycles	291	420	992
Politics	1714	1415	775
Space	482	1941	985

Table 4.15: Number of documents in each category under "All" group as obtained from three different domains, Wikipedia, NYT and Newsgroups.

	Wiki-NYT		Wiki-News		News-NYT	
Vocabulary Size	42410		34705		32357	
Min. # of words in document	6	8	6	7	7	7
Ave. # of words in document	604.2	416.9	599.5	155.8	155.5	414.2
Max. # of words in document	11985	5252	11891	6553	6591	5162

Table 4.16: Some useful attributes of the dataset for the "All" group, including the feature vector dimension (size of the vocabulary) and size of the documents etc.

Category	Wiki	NYT	Newsgroups
Guns	63	966	910
MiddleEast	850	1581	940
Politics	1720	1545	775

Table 4.17: Number of documents in each category under "Politics" group as obtained from three different domains, Wikipedia, NYT and Newsgroups.

Tables 4.17, 4.18, 4.19, 4.20, 4.21, 4.22 list similar information for the other three groups of categories used, namely, "Politics," "Rec" and "Sci."

4.4 Conclusion

In this chapter, we summarized the different datasets that we generated for our cross-domain document classifications. Since the filed of cross-domain classification is relatively new, the research community lacks the datasets suitable for this research. The cross-domain document

	Wiki-NYT		Wiki-News		News-NYT	
Vocabulary Size	19013		16231		13473	
Min. # of words in document	10	11	10	8	8	11
Ave. # of words in document	745.5	423.1	737.2	207.9	206.5	416.2
Max. # of words in document	11840	6143	11685	4802	4735	6048

Table 4.18: Some useful attributes of the dataset for the “Politics” group, including the feature vector dimension (size of the vocabulary) and size of the documents etc.

Category	Wiki	NYT	Newsgroups
Automobiles	784	1974	988
Baseball	552	1810	994
Hockey	139	1815	998
Motorcycles	291	424	992

Table 4.19: Number of documents in each category under “Rec” group as obtained from three different domains, Wikipedia, NYT and Newsgroups.

	Wiki-NYT		Wiki-News		News-NYT	
Vocabulary Size	16526		10636		15507	
Min. # of words in document	6	11	6	5	6	11
Ave. # of words in document	484.453	395.585	468.979	111.47	114.238	393.737
Max. # of words in document	7981	4172	7926	6219	6417	4130

Table 4.20: Some useful attributes of the dataset for the “Rec” group, including the feature vector dimension (size of the vocabulary) and size of the documents etc.

Category	Wiki	NYT	Newsgroups
Cryptography	719	1217	991
Electronics	1456	971	981
Medicine	2387	1986	988
Space	482	1952	985

Table 4.21: Number of documents in each category under “Sci” group as obtained from three different domains, Wikipedia, NYT and Newsgroups.

classification requires datasets that contain labeled documents in one domain over some predetermined categories and unlabeled documents on the similar categories in a second domain. With this type of data, one can run experiments by using the first dataset as the training dataset and the second dataset as the testing dataset. In the datasets that are created as part of this dissertation, we use three different domains that all share common categories among them. This makes it possible to run the cross-domain experiments among three different pairs of domains,

	Wiki-NYT		**Wiki-News**		**News-NYT**	
Vocabulary Size	23426		18533		17100	
Min. # of words in document	7	7	7	7	7	7
Ave. # of words in document	480.286	401.966	474.368	143.685	142.835	396.919
Max. # of words in document	8673	6173	8554	6077	6044	6103

Table 4.22: Some useful attributes of the dataset for the "Sci" group, including the feature vector dimension (size of the vocabulary) and size of the documents etc.

and in each pair, each domain can serve as a test or train domain. In addition to this, we create datasets using four or five different category groups, giving an additional sets of data to test consistency and robustness of the cross-domain classification algorithms. Our datasets are extensive and can be used as a benchmark for future research on cross-domain document classification. We also describe the way we created these datasets, since it may be desirable to update the datasets from time to time, given the dynamic nature of data sources such as Wikipedia and the introduction of new domains such as youtube video descriptions, twitter etc. These datasets and the documentation on the process to generate them is one of our important contributions in this dissertation.

CHAPTER 5: Wikipedia Classification

This chapter demonstrates an effective way of parsing Wikipedia articles for text classification purposes. The information in this chapter can be used to serve two purposes: (1) classification of Wikipedia articles, (2) classification of articles from a different domain, such as New York Times (NYT) news. Here we describe methods of data collection for Wikipedia articles and empirically analyze different sections of the Wikipedia articles to be used in document classification. We also show the effectiveness of our methods to classify the given articles to their categories.

Each Wikipedia article consists of standard sections, such as “introduction,” “references,” “links to other Wikipedia articles” etc. In this chapter, we analyze the results of each section independently and propose a method of combining some of these sections to improve the classification accuracy. We use multiple datasets from Wikipedia each containing documents from number of different categories. Our results show a consistent improvement in classification accuracy of the articles when different sections of the articles are used independently, instead of using the entire text of a Wikipedia article all together at one time. Our results not only show an improvement in classification, but also show that using only parts of the articles greatly reduces the size of the training feature vectors. For our experiments, we classify unlabeled Wikipedia and NYT articles into their respective categories. In both cases, only the labeled Wikipedia articles are used for training.

Wikipedia is an abundant source of labeled text documents that is freely available and covers almost all subjects. Wikipedia was launched in 2001 and it has been growing exponentially since then [61]. It now contains over 8 million articles in English language alone. All of these articles in Wikipedia are assigned to various categories, which can be used as the document labels to prepare a training set for a classification algorithm.

Use of Wikipedia, as a supporting dataset, to classify documents in a different domain has received significant attention. However, there are three areas that need to be improved namely (1) availability of benchmark datasets, (2) documentation on how to efficiently gather labeled data from Wikipedia and (3) how to use the Wikipedia articles for text classification purposes. A set of algorithms for effectively exploiting a vast source of labeled documents, such as Wikipedia, for text document classification is highly desirable.

This chapter describes a way to obtain Wikipedia data for text mining purposes and empirically analyzes the inherent structure of Wikipedia articles for the text classification task. As our contribution, we show that our method of parsing and combining the article sections improves classification accuracy of Wikipedia articles as well as articles from a different domain, such as New York Times. For our experiments, we classify unlabeled Wikipedia articles in to their respective categories. We create our dataset by collecting articles under different, but related, categories. 80% of these articles are used for training and then are cross validated on the other 20%. We repeat this process for three different groups of five categories to show the generalization of our results. In our experiments, we use three supervised learning methods, namely, support vector machines (SVMs), k-nearest neighbors (KNNs) and naïve bayes (NB).

This chapter of the thesis is organized as follows. Section 1 outlines the Wikipedia data format and discusses how a good dataset can be collected from Wikipedia for text mining purposes. Section 2 gives details of our dataset and different sections of the Wikipedia articles explored in our experiments. In Section 3, we show and analyze our results of classification using different sections of the Wikipedia article. Section 4 proposes ways of combining different sections of the Wikipedia articles to increase the accuracy of the classification. In Section 5, we show our results for cross domain classification using NYT as our test domain. In cross-domain classification, only the Wikipedia articles are used to classify text from a different domain, namely news stories from NYT. Section 6 outlines the future work and conclusion of this part of our research.

5.1 Gathering Data from Wikipedia

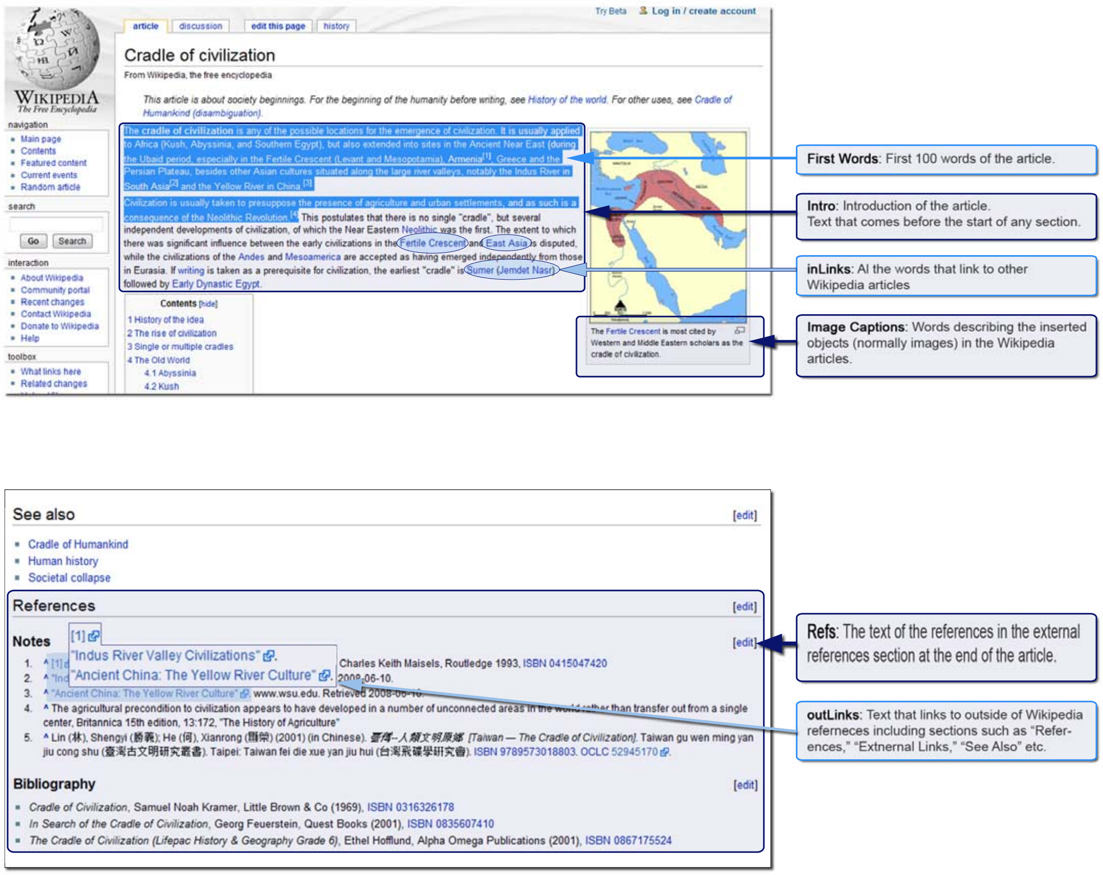
We generate 4 groups of categories for our experiments where each group contains categories that all belonged under the umbrella title of that group. These groups are chosen to make the classification tasks challenging and closer to real life classification challenges. Table 5.1 shows the breakdown of the groups and their categories and the number of articles in each category. For example, group named “Arts” includes 6 categories, which are (1) Theatre, (2) Music, (3) Opera, (4) Film, (5) Television and, (6) Literature. The third columns shows the number of articles in each of these categories such as Theatre category has 497 articles.

Table 5.1: Groups of Categories

Group Name	**Categories**	**Articles**
Arts (6)	Theatre	497
	Music	554
	Opera	3189
	Film	989
	Television	872
	Literature	1744
Current Affairs (6)	Finance	561
	Military	637
	Terrorism	432
	Law	1370
	Christianity	900
	Islam	1241
Science (6)	Genetics	2095
	Space	481
	Anthropology	1533
	Medicine	2292
	Chemistry	2869
	Physics	2718
Technology (5)	Software	926
	Electronics	1228
	Internet	864
	Telecommunications	1905
	Computer Security	1070

5.2 Parsing the Wikipedia Article

Our goal in this part of the research is to exploit the article structure that is common in Wikipedia. Most articles in Wikipedia is composed of a set of pre-determined sections, such as most articles contain an introduction section in the beginning of the article and a reference section at the end of the article. We make a list of nine of these sections that are common across many articles while having some distinctive characteristics. These sections and their descriptions are listed in the table 5.2. For example, section #2 is named "Main Section" which contains the main body of the article, in other words, the entire Wikipedia article excluding sections #4, #7 and #8. Figure 5.1 shows some of the different sections as they occur in a typical Wikipedia articles.



Cradle of civilization

From Wikipedia, the free encyclopedia

This article is about society beginnings. For the beginning of the humanity before writing, see History of the world. For other uses, see Cradle of Humankind (disambiguation).

The cradle of civilization is any of the possible locations for the emergence of civilization. It is usually applied to Africa (Kush, Abyssinia, and Southern Egypt), but also extended into sites in the Ancient Near East (during the Ubaid period, especially in the Fertile Crescent (Levant and Mesopotamia), Armenia[1], Greece and the Persian Plateau, besides other Asian cultures situated along the large river valleys, notably the Indus River in South Asia[2] and the Yellow River in China.[3]

Civilization is usually taken to presuppose the presence of agriculture and urban settlements, and as such is a consequence of the Neolithic Revolution.[4] This postulates that there is no single "cradle", but several independent developments of civilization, of which the Near Eastern Neolithic was the first. The extent to which there was significant influence between the early civilizations in the Fertile Crescent and East Asia is disputed, while the civilizations of the Andes and Mesoamerica are accepted as having emerged independently from those in Eurasia. If writing is taken as a prerequisite for civilization, the earliest "cradle" is Sumer (Jemdet Nasr) followed by Early Dynastic Egypt.

Contents [hide]
1 History of the idea
2 The rise of civilization
3 Single or multiple cradles
4 The Old World
4.1 Abyssinia
4.2 Kush

The Fertile Crescent is most cited by Western and Middle Eastern scholars as the cradle of civilization.

First Words: First 100 words of the article.

Intro: Introduction of the article.
Text that comes before the start of any section.

inLinks: Al the words that link to other Wikipedia articles

Image Captions: Words describing the inserted objects (normally images) in the Wikipedia articles.

See also [edit]
- Cradle of Humankind
- Human history
- Societal collapse

References [edit]

Notes [edit]

[1]
"Indus River Valley Civilizations".
"Ancient China: The Yellow River Culture".

1. ^ [1] Charles Keith Maisels, Routledge 1993, ISBN 0415047420
2. ^ "Ind 2008-06-10.
3. ^ "Ancient China: The Yellow River Culture". www.wsu.edu. Retrieved 2008-06-10.
4. ^ The agricultural precondition to civilization appears to have developed in a number of unconnected areas in the world rather than transfer out from a single center, Britannica 15th edition, 13:172, "The History of Agriculture"
5. ^ Lin (林), Shengyi (勝義); He (何), Xianrong (顯榮) (2001) (in Chinese). *臺灣--人類文明原鄉 [Taiwan — The Cradle of Civilization]*. Taiwan gu wen ming yan jiu cong shu (臺灣古文明研究叢書). Taipei: Taiwan fei die xue yan jiu hui (台灣飛碟學研究會). ISBN 9789573018803. OCLC 52945170.

Bibliography [edit]
- *Cradle of Civilization*, Samuel Noah Kramer, Little Brown & Co (1969), ISBN 0316326178
- *In Search of the Cradle of Civilization*, Georg Feuerstein, Quest Books (2001), ISBN 0835607410
- *The Cradle of Civilization (Lifepac History & Geography Grade 6)*, Ethel Hofflund, Alpha Omega Publications (2001), ISBN 0867175524

Refs: The text of the references in the external references section at the end of the article.

outLinks: Text that links to outside of Wikipedia referneces including sections such as "References," "Extnernal Links," "See Also" etc.

Figure 5.1: A figure showing different sections of an article from Wikipedia. The sections are shown as they are used in our experiments. These sections can be found in most of Wikipedia articles.

To generate our feature vectors, we parse each article into its different sections. After the parsing, feature vectors are generated using the raw counts of the words. In the pre-processing of the data before making the final feature vector, we remove the following: (1) Stop words, (2) any words with digits, (3) words that appear less than three times in the entire dataset. In our experiments, we did not use any feature vector from a document section that had less than 10 characters in it. These small feature vectors for a section, containing less than 10 characters in them, mostly appeared due to the absence of that section from the article. For example, if an article did not have an image, then the feature vector that contained the image section of that

Table 5.2: Description of individual sections used in this chapter

Section #	**Name**	**Description**
1	All Sections	All sections of the Wikipedia articles, excluding the names of the article categories at the end
2	Main Section	The entire Wikipedia article excluding sections (4), (7) and (8)
3	Intro	Introduction of the article. Text that comes before the start of any section
4	inLinks	All the words that link to other Wikipedia articles
5	First Words	First 100 words of the article
6	Blue Words	All the words that are linked either within Wikipedia or to an external website
7	Refs	The text in the external references section at the end of the article
8	Image Captions	Words describing the inserted objects (normally images) in the Wikipedia articles
9	outLinks	Text that linked to an external website, this includes sections such as “References,” “External Links”

article will have 0 characters (less than 10) in it. Among the four groups of the categories, as discussed before, and nine initial sections, we have total of 36 different datasets. Each dataset containing articles from categories listed under one of the four groups and feature vectors that were generated using a particular section of the articles. For example, one dataset would include feature vectors from only the “inLinks” section of the articles under the group “Arts.” This way, we obtain nine different training datasets only for the category group “Arts,” each such dataset can be used for classifying the articles.

In our experiments, we do use all 36 of these datasets and compare the results of each section individually. Next section will discuss our experimental results for individual sections and provide discussions.

5.3 Individual Section Results

Our goal in this section is to evaluate and compare the classification accuracy of different sections of the Wikipedia articles. We use four different groups to make sure that the results are consistent and generalizable across documents of different subject matter. Each one of the datasets was classified using three main classification algorithms, namely support vector machines (SVM), naive bayes (NB) and *k*-nearest neighbor (KNN) with cosine similarity as the "distance" measure. The number of nearest neighbor (*k*) was 31. The accuracy results did not vary much with *k*, as we let *k* vary from 11 to 51. 31 was chosen to be a suitable number for 6 classes.

Table 5.3, shows the accuracy of different sections in terms of percentage on the "current affairs" group. The "current affairs" group contains the categories shown in table 5.1. For other category group results, please refer to the appendix section. Figure 5.2 shows the plot of size of different sections (on x axis) and the accuracy in percentage. This graph is consistent for different groups of categories (see results in the appendix). We see that the additional data in the entire Wikipedia article (AllSections) does not provide any advantage in getting better accuracy when compared with data only in the first few words (FirstWords) and the linked words (BlueWords). This can be seen by noticing that bubble numbers 2 and 3 (FirstWords and Intro) are almost at the same height as the bubbles 12 and 13 (MainSection and AllSections). Furthermore, we see that if you combine sections representing first few words (FirstWords and Intro) with sections representing only the linked words (inLinks and BlueWords), we achieve an accuracy that is higher than using the entire Wikipedia article (AllSections). This is shown in the Figure 5.2 since bubble 10 and 11 are higher than bubbles 12 and 13.

We compare and analyze the accuracy of each section so that sections can be compared against each other in terms of their performance and redundancy. From Figure 5.2, ImageOrFile, Refs and RefsExtLinkSeeAlso, give significantly low accuracy results. Therefore, we eliminate those sections from our analysis and only use 6 of the original 9 sections. The comparison is performed by analyzing (1) which articles each section classified correctly and (2) how often different sections agreed with one another. We want to find the pairs of sections that best complemented each other and thus could be combined to improve the overall classification. We cannot rely on measures like correlation or mutual information blindly since these measures do not take into account the accuracy of the individual section. For example, if two classifiers were random label generators, the mutual information between the two classifiers is likely to

Table 5.3: Classification Accuracy on Individual Sections

ID	Section Name	Average Size (in Words)	KNN	NB	SVM
1	All Sections	349	87.8	71.5	90.9
2	Main Section	320	86.7	71.7	90.2
3	Intro	63	83.1	70.6	90.1
4	inLinks	77	83.7	79.6	89.4
5	First Words	44	83.9	70.9	90.2
6	Blue Words	114	85.1	79.1	89.4
7	Refs	68	70.1	55.7	78.8
8	Image Captions	32	66.8	3.1	74.7
9	outLinks	69	74.9	61.2	82.7

be very low but the accuracy of both classifiers will also be low. Therefore, before using these measures, we need to ensure that each individual section being compared has reasonably high accuracy. While comparing the sections for finding most suitable pairs, we would like to find pairs that showed less correlation or low mutual information while both sections still having high accuracy. To compare two sections, e.g., A and B, or to measure the degree of *redundancy* in these sections, we used correlation coefficient and normalized mutual information (NMI) as described in the paper [66] between the label predictions of each section. The NMI measure gives a value of +1, if X and Y are perfectly correlated (either negatively or positively) and a value of a 0 if X and Y are independent. The formulation of NMI measure and correlation coefficient can be seen in Equation 5.1 and 5.2 respectively.

$$NMI(X,Y) = \frac{\sum_{x,y} p(x,y) \ln \frac{p(x,y)}{p(x)p(y)}}{-\sum_{x,y} p(x,y) \ln p(x,y)} \tag{5.1}$$

$$\rho_{(X,Y)} = \frac{cov(X,Y)}{\sigma_x \sigma_y} \tag{5.2}$$

Following is a sample table for the value of NMI (Table 5.4) and correlation coefficient (Table 5.5) obtained using the given formulas, where X and Y were raw predicted labels. These

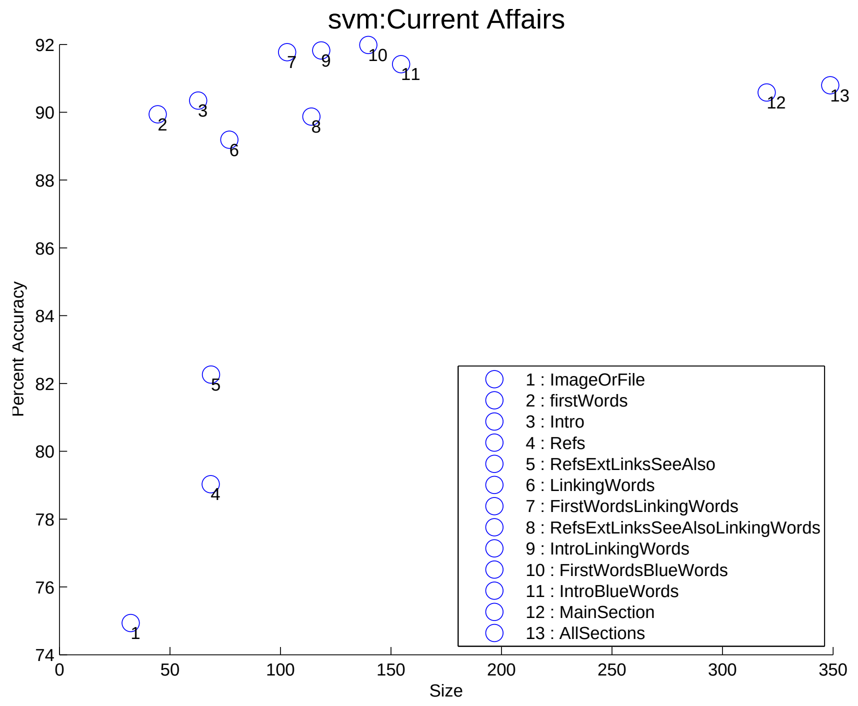


Figure 5.2: Accuracy of different sections plotted against the average size of the feature vector. X-axis has the average number of words in the section and Y-axis shows the average accuracy obtained by using only that section of the Wikipedia for classification. These results are obtained using "current affairs" group of categories as the data and SVM as the classifier.

labels are for the classes under the group "current affairs" and the classifier SVM. For example, the Table 5.4 shows that the normalized mutual information between section ID 3 (Intro) and section ID 6 (Blue Words) is 0.499 and the correlation between these two sections, shown in Table 5.5, is 0.833. In the rest of this chapter, to prevent redundancy, wherever the trend is similar across groups, we will use the "current affairs" group, and SVM classifier to present our results. Table 5.4 and Table 5.5 show that a simple correlation formula also has the similar pattern of independence, however, the numbers show a better spread over the range in NMI. The pairs of section that show a low correlation or NMI measure contain the least amount of redundant information and since the sections compared all had good accuracy, the pairs that show low mutual information should increase the accuracy when combined.

5.4 Combined Section Results

We also combine some of the sections in different ways to create "hybrid" classifiers. We combine the classifiers simply by using both sections together to generate the feature vector. From the table of NMI measures between the sections, we see that the pairs [Intro (3) , inLinks(4)],

Table 5.4: NMI Measure Between Wikipedia Sections

Section ID	**1**	**2**	**3**	**4**	**5**	**6**
1	1.000	0.714	0.578	0.549	0.585	0.561
2	0.714	1.000	0.564	0.535	0.581	0.531
3	0.578	0.564	1.000	0.499	0.611	0.499
4	0.549	0.535	0.499	1.000	0.498	0.632
5	0.585	0.581	0.611	0.498	1.000	0.499
6	0.561	0.531	0.499	0.632	0.499	1.000

Table 5.5: Correlation Measure Between Wikipedia Sections

Section ID	**1**	**2**	**3**	**4**	**5**	**6**
1	1.000	0.938	0.875	0.862	0.882	0.860
2	0.938	1.000	0.878	0.868	0.888	0.857
3	0.875	0.878	1.000	0.840	0.898	0.833
4	0.862	0.868	0.840	1.000	0.841	0.891
5	0.882	0.888	0.898	0.841	1.000	0.835
6	0.860	0.857	0.833	0.891	0.835	1.000

[First Words (5), inLinks(4)], [Intro (3), Blue Words(6)] and [First Words (5), Blue Words(6)] best complement each other. These pairs are consistent among different groups of classes mentioned in this chapter. We run the classification on these four pairs of sections. Following are the results, in Table 5.6, that show the accuracy of these four pairs along with the results obtained by using the entire Wikipedia article i.e. section, “All sections”. From the Table 5.6 and the Figure 5.3, we see that using combinations of sections like First Words and Blue Words (ID 5,6) consistently outperform the accuracy given by the entire combined article (ID 1).

A different way of combining different sections of Wikipedia can be performed by taking the majority vote for the predictions among different sections. In this way, we first individually obtain a classification label from each section. Then we assign the label to the document that is the given by the majority of the sections independently. For example, to classify an article X, we first classify it using only one section at a time and obtain the classification labels. Assuming that two of the sections labeled X as finance and four of the sections labeled X as

Table 5.6: Classification Accuracy (in percentage) for Combination of Sections

ID	**Section Name**	**Average Size (in words)**	**Arts**	**CA**	**Science**	**Tech**
1	All Sections	349	92.2	90.9	88.8	81.9
3,4	Intro-inLinks	118	93.6	91.6	89.6	82.4
5,4	Firsts-inLinks	103	93.2	92.1	89.4	83.2
3,6	Intro-Blues	155	93.2	91.8	89.6	82.6
5,6	Firsts-Blues	140	93.3	91.8	89.7	83.4

law, then the final classification label of X will be law, as it was the label given to it by majority of sections. When we take a majority vote of using the section individually, it significantly improves the performance over using the entire article together or even the section "First Words - Blue Words" alone. Figure 5.4 shows the accuracy results as obtained by three different methods of combining the sections of a Wikipedia article, namely, (1) all sections together by using the entire article, (2) using only the first few words of the article combined with the linked words (sections Firsts-Blues) and, (3) using individual sections independently and then taking a majority vote to determine the final classification. Figure 5.4 shows that using the entire article together at one time gives you the least accuracy when compared to the other methods. However, if one does use the entire text of the article but by taking majority of the individual sections, we get better accuracy than using first words and the linked (blue) words.

After classifying Wikipedia articles using different sections, we also experimented by classifying articles in a different domain, namely NYT, to show the effectiveness of parsing articles in to different sections for cross-domain classification. In our cross-domain classification, only the labeled Wikipedia articles were used to classify the NYT articles. Since the two different domains have different distribution of documents in each category as well as difference in the writing style, we expect such a cross-domain classification accuracy to be significantly lower than in-domain (where test and training set are from the same domain) classification. Even though, the accuracy in the cross-domain experiments, as expected, is lower in absolute terms, we still get an improvement in the classification accuracy while using a parsed Wikipedia article over using the entire article together. Figure 5.5 shows a clear pattern that the individual Wikipedia sections outperform the entire article together or at least do as good as using the entire article even in the case of cross domain classification.

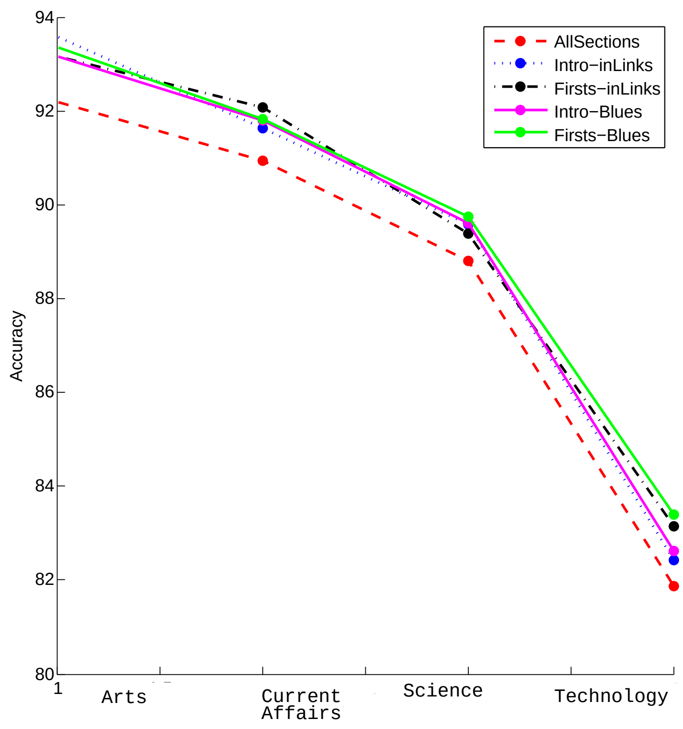


Figure 5.3: Accuracy of different sections for difference groups of categories. The graph consistently shows that Section, "All Sections," does not perform as well as the combination of first few words (section FirstWords or Intro) and links (section BlueWords or inLinks) in the Wikipedia article. In addition to this improvement, one should also note the reduction in the training data since first Words and Blue Words combined consist of less than 50% of the article.

5.5 Conclusion

Our work on Wikipedia shows that one can get better accuracy in classification results by using only the "Blue Words" and the introduction part of the article than by using the entire article of Wikipedia. This reduces the article text (or sparse feature vector length) by almost 70%. We also show that a more effective way of using the entire article text is to first parse it into different sections and then take a majority vote of the different classifiers, created by using different sections individually. We demonstrate that these results are consistent among different groups of classes. We also use Wikipedia to classify a completely unlabeled dataset from a different domain, in this case, new stories from NYT. We see that different parts of the Wikipedia again outperform the entire article taken together while classifying documents in a different domain. The initial approach of using parsed components of Wikipedia articles instead of using the entire article text yields promising results. There are many different ideas to further extend this research topic. We can develop different measures to combine different sections and therefore improve the cross domain accuracy. In this chapter, we use Normalized Mutual Information

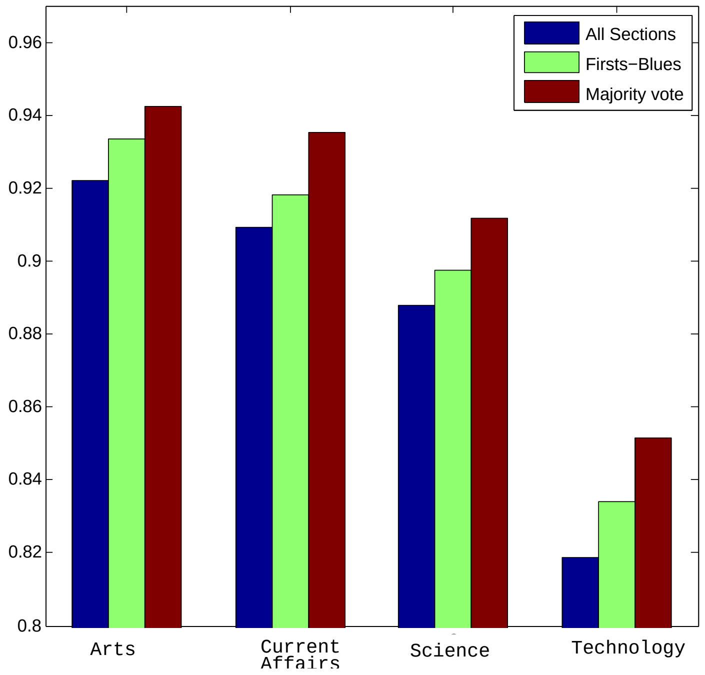


Figure 5.4: Comparison of accuracies of the entire article, only first and blue words, and the majority vote from different sections. We see that using the entire Wikipedia article does not give us the best accuracy. We achieve a significant improvement in the classification accuracy when different sections are used independently and combined by majority vote.

(NMI) and correlation coefficient as a measure to combine different sections. In the future, we would like to experiment with different datasets from variety of domains.

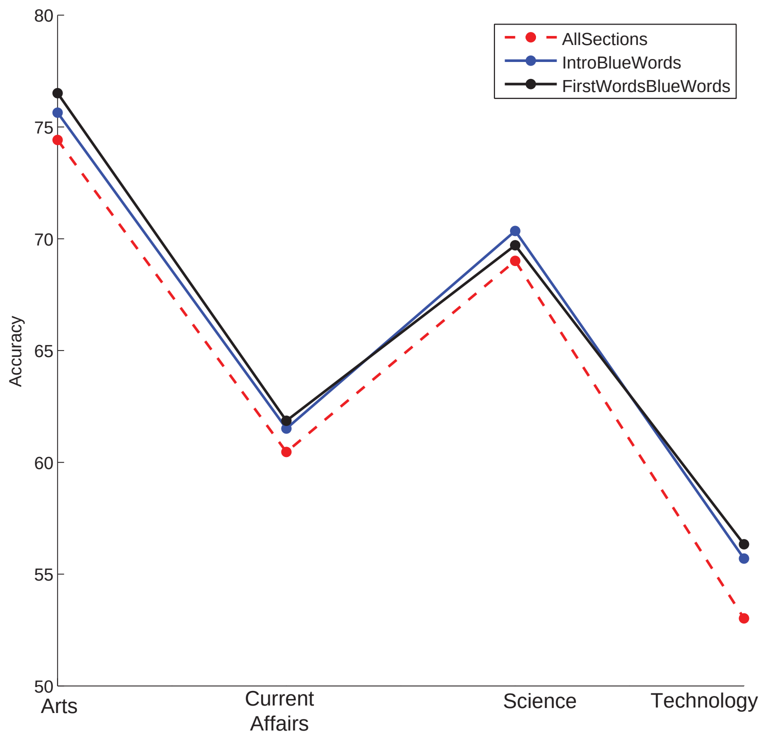


Figure 5.5: Comparison of accuracy for the cross-domain classification. Even though the difference between the accuracy obtained from different section combination is small, it still shows an improvement in the accuracy while using the combination of a few sections (solid lines) over the accuracy obtained from using the entire section at once (dotted red line).

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 6:
Cross-Domain Classification Using Conventional Algorithms

In the previous chapter, we discussed how one can use Wikipedia as a common source of labeled documents to classify unlabeled documents in another domain. We introduced methods that were specifically designed to work with Wikipedia articles as the source of labeled training documents. These Wikipedia-specific algorithms parsed a Wikipedia article into its different sections, such as "Introduction," "References," "External Links," etc. Although we achieved an improvement in the accuracy of the cross-domain classification of NYT articles by using the Wikipedia specific algorithms, our main research goal is to develop general algorithms. These general algorithms can be used to classify articles in domain Y by using labeled articles from domain X, where X and Y can be any two domains.

In this chapter, we will first establish the need for cross-domain classification by showing that conventional algorithms fail to classify documents accurately when training and testing data is used from two different domains. We will show that the average drop in the accuracy of a conventional algorithm going from single domain to multi-domain classification ranges from 22%-30%. In the next chapter, we will introduce algorithms that improve the accuracy compared to the conventional methods for the cross-domain document classification for any two domains.

We choose a set of most commonly used classifiers for text document classification for our empirical studies on the cross-domain document classification. These conventional classifiers are chosen due to their simplicity, popularity in document classification research and robust theoretical foundation. We choose four such classifiers including, (1) K-nearest neighbor with Euclidean distance (KNN-Euclidean), (2) K-nearest neighbor with Cosine distance (KNN-Cosine), (3) Support Vector Machines (SVMs) and (4) Naive Bayes (NB).

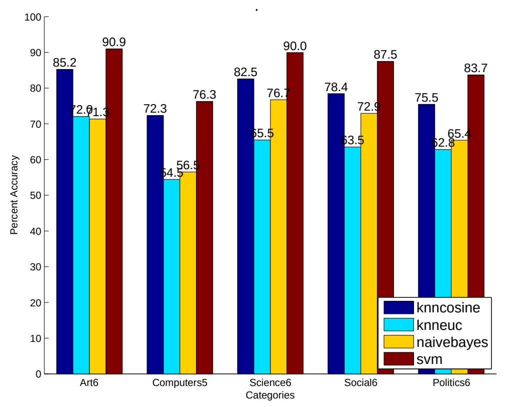
In the first section we show the classification results when we use the entire vocabulary from two domains to generate our counts feature vectors as explained in Chapter 3. In the second section, we reduce the vocabulary by only including the words that appear in both domains and appear at least 100 times in the entire dataset of two domains. The motivation to reduce the vocabulary stems from our observation of the results as introduced in the first section of this chapter and from some previous works on text document classification [67, 68].

In the conclusion of this chapter, we also obtain baseline measurements that can be used to compare our algorithms for the cross-domain classification. These baseline measurements are the best accuracy results that we obtain from the conventional classifiers. Choosing the conventional classifiers and the feature vectors that give us the best cross-domain classification performance is the fairest way to evaluate our algorithms.

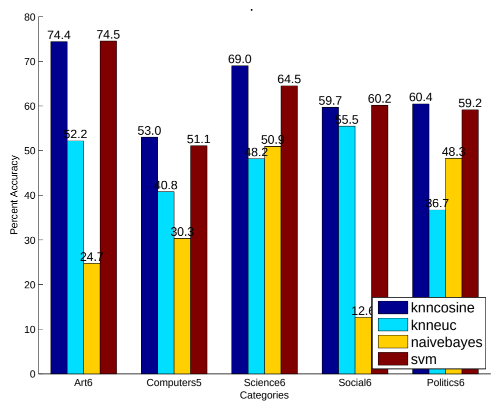
6.1 Classification Using Entire Vocabulary

In this section we present experimental results of document classification on our cross-domain data using conventional classifiers such as KNN, SVMs and NB. The feature vectors, used in these experiments, were the raw word counts from the text after removing the stop words. The entire vocabulary from both domains was used to obtain these counts. In the case of KNN classifier, two common distance measures, namely, *cosine similarity* and *Euclidean distance* measures were used. The chosen K for the KNN classification was 31. This K was chosen after preliminary experiments with the data. Besides only relying on the experiments, the K value of 31 was also chosen because it is a reasonable value for a multi-class classification when the number of classes is 5 or 6. Our algorithms show an improvement over the best conventional algorithms for the cross-domain classification. We use the results of the conventional classifier algorithms that provide the best cross-domain accuracy as our benchmark to represent the result from the conventional algorithms. Figure 6.1 shows the results of the conventional classifiers on the classification of the Wikipedia and NYT data.

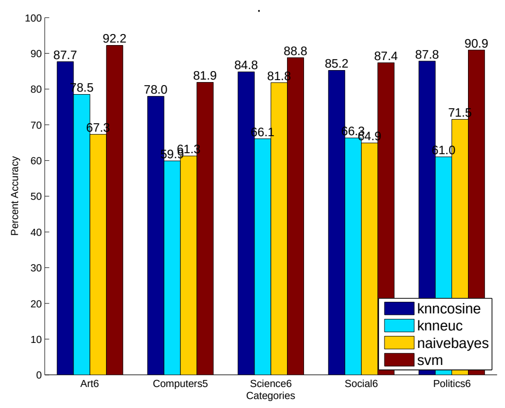
Figure 6.1 contains four sub-figures in two rows. Top row shows the results of classifying NYT articles with using NYT as training set (a) and Wikipedia as training set (b). Bottom row shows the results of classifying Wikipedia articles using Wikipedia as training set (c) and NYT as training set (d). In each figure, the x-axis shows different category groups such as “Arts,” “Computers” etc., while y-axis shows the accuracy in percentage. Figure 6.1 includes a legend listing the four conventional algorithms. It can be seen, by comparing right and left bar graphs, that when training and testing sets were from different domains the classification accuracy drops significantly. This drop in the accuracy varies for different classifiers as well as different group of categories. For example, the average accuracy drop in the classification of NYT articles from using NYT as training set and using Wikipedia as training set is 28% when SVM classifier is used.



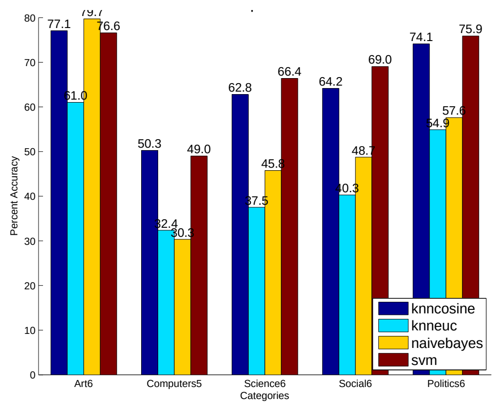
(a) caption: NYT2NYT



(b) caption: Wiki2NYT



(c) caption: Wiki2Wiki



(d) caption: NYT2Wiki

Figure 6.1: Classifier Comparison Using the Entire Vocabulary from Two Domains

Table 6.1 shows the average drop in the accuracy when we classify across domains, that is when we use different domains for test and training sets. The percentage drop in Wiki2NYT column is calculated by comparing the accuracy obtained from the NYT2NYT column. That is the drop in the accuracy in classifying NYT articles when using Wikipedia as the training set (Wiki2NYT) versus when using the NYT as the training set (NYT2NYT). Similarly percent drop in NYT2Wiki column is the drop in accuracy compared with Wiki2Wiki. We see that each one of these classifiers has an average of 25% drop in the accuracy, with Naive Bayes (NB) and KNN-Euclidean performing the worst, and KNN-Cosine and Support Vector Machines (SVMs) performing the best.

As observed from the Figure 6.1 and Table 6.1, we see that NB and KNN-Euclidean classifiers

	Accuracy				% Drop in Accuracy	
Classifier	Wiki2Wiki	NYT2Wiki	NYT2NYT	Wiki2NYT	NYT2Wiki	Wiki2NYT
knncosine	84.7	65.7	78.8	63.3	22.8	19.9
knneuc	66.4	45.2	63.7	46.7	32.1	26.6
naivebayes	69.4	52.4	68.6	33.4	24.1	50.8
svm	88.2	67.4	85.7	61.9	24.0	28.0

Table 6.1: Table showing the average accuracy, over all the category groups, for cross-domain classification as obtained from different classifiers and a corresponding percentage accuracy drop. The percentage drop in Wiki2NYT column is the drop in accuracy compared with NYT2NYT, similarly percent drop in NYT2Wiki is compared with Wiki2Wiki.

perform the worst and also show a more significant accuracy drops. This discrepancy in the classifier performance gives us a hint for the reason of this drop. Both NB and KNN-Euclidean are heavily influenced by words that appear only in training or only in testing set. For example, if NB classifier is given words in the testing data that it never observed in the training data, it results in a division by zero (that is normally handled by some artificial smoothing). In our case, we perform this smoothing by assigning probability of 10^{-5} to words that are not seen in the training data but are seen in the testing data. Euclidean distance between two vectors v_1 and v_2 is also heavily influenced by words that appear only in one of the two vectors.

Considering this pattern in the classifiers and surveying other literature [67, 68, 69, 70], we list some of the factors that may contribute to this drop in the accuracy. Following is our list of some of the reasons that may be responsible for this drop in the cross-domain classification accuracy:

1. Different size of the feature vectors and different length of the articles;
2. Different proportions of the classes in each domain [69];
3. Different vocabulary. Many unique words that only appear in one of the domains;
4. Different distance measures. The points in different domains may be clustered or distributed with different distributions, thus having a different underlying distance measures between them;
5. Different "meaning" of the same topic or words. This is the most abstract but probably one of the most important reasons that explains the inability of the conventional classification methods to generalize across domains.

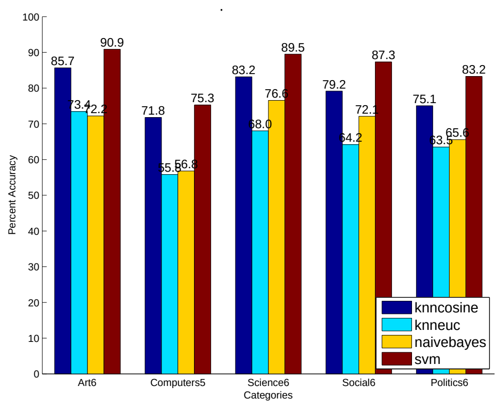
This list also gives us a summary of the issues that need to be addressed in the course of this research. Items (1) and (3) from the list above are supported by the poor performance of NB and KNN-Euclidean and relatively better performance of SVMs and KNN-Cosine. Since KNN-Cosine and SVMs use the inner product of the feature vectors, and KNN-Cosine also is resistant to different sizes of the feature vectors, the loss in the accuracy is relatively smaller in the case of KNN-Cosine and SVMs. Item (1) in the list is also encountered in single domain classification as two documents can vary greatly in size within a single domain. We resolve this issue by simply normalizing the feature vectors.

6.2 Classification Using Intersected Vocabulary Obtained by Removing Rare and Unique Words

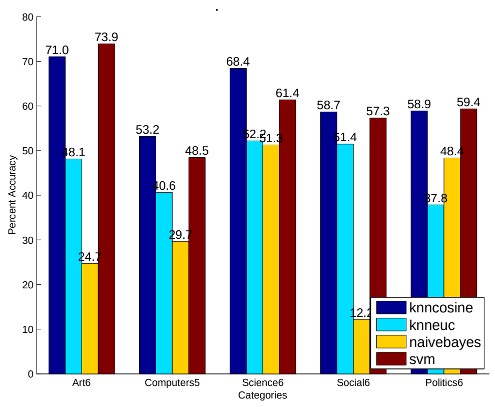
In this section we present cross-domain classification results after removing the words that appear less than 100 times in the two domains (rare words) and words that only occur in one of the two domains (unique words) from the vocabulary. This intersection of the two vocabulary spaces, belonging to two different domains, addresses the item (3) in the list of reasons as introduced in the previous section. We notice a relatively low performance of KNN-Euclidean and NB even when the same domain is used for training and testing e.g., Wiki2Wiki or NYT2NYT. This can be caused by the presence of rare words that often do not contribute to the classification accuracy [67, 68]. Even though, item (3) discusses the presence of only the "unique" words, to get more meaningful intersection of the vocabulary spaces, we also remove the rare words as they have a similar effect to the unique words in the cross-domain classification. So we obtain the intersection of the two vocabulary spaces by removing the words that occur less than 100 times in the two domains and by removing the words that only occur in one of the two domains.

Figure 6.2 shows the drop in the accuracy in percentage when the vocabulary has been reduced by 90% by removing the rare and unique words. Figure 6.2 contains four sub-figures in two rows laid out in the same manner as Figure 6.1. Table 6.2 shows the average drop in the accuracy with the reduced vocabulary, the average is taken across all different category groups of the domain pair Wiki-NYT.

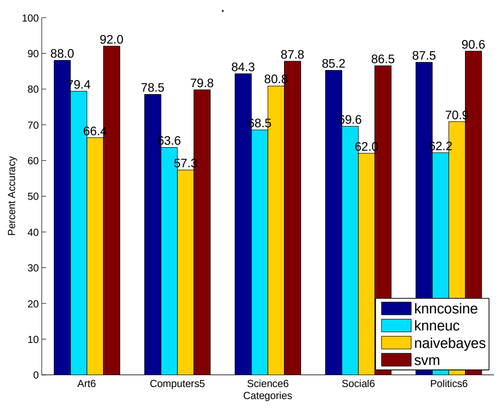
By removing these words, we reduce the total size of the vocabulary by over 90%. As expected we do not get any significant reduction in the performance of the classifiers. Figure 6.2 shows the graphs of the accuracy with the reduced vocabulary. We see an average drop of less than 1% in most cases and in some cases we even see an increase in the accuracy.



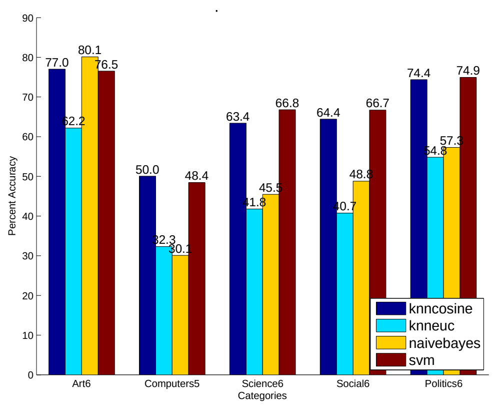
(a) caption: NYT2NYT



(b) caption: Wiki2NYT



(c) caption: Wiki2Wiki



(d) caption: NYT2Wiki

Figure 6.2: Classifier Comparison After Intersecting the Vocabulary Spaces

One, somewhat surprising, result is that when we remove the words that only occur in one of the domains, it even increases the accuracy of the single domain classification. This is an interesting and useful result as it gives an efficient way to reduce the dimension of a text collection by removing the words that do not occur in other similar domains. For example, these results show that we do not lose any accuracy in the New York Times article classification by not using the words that never appear in Wikipedia. Even though Wikipedia and NYT are independent and distinct domains, the most useful words for classification are the words that occur in both domains. This process brought down the average dimension of the feature vectors from 90,000 to 9,000. This reduction in the size of the feature vectors makes running all the experiments possible in realistic amount of time without losing any accuracy in the process. Unless otherwise specified, we do not use any other dimension reduction methods such as principal component

	Accuracy				% Drop in Accuracy	
Classifier	Wiki2Wiki	NYT2Wiki	NYT2NYT	Wiki2NYT	NYT2Wiki	Wiki2NYT
knncosine	84.7	65.8	79.0	62.0	22.6	21.7
knneuc	68.6	46.4	65.0	46.0	32.6	29.1
naivebayes	67.5	52.4	68.7	33.2	22.2	51.3
svm	87.3	66.7	85.4	60.1	24.1	29.7

Table 6.2: The average drop in the accuracy in percentage across different category groups with the intersected vocabulary space between two domains. The vocabulary has been reduced by 90% by removing the rare words and the words that were unique to one of the two domains. Compared with Table 6.1 it shows almost no difference in the accuracies.

	Accuracy				% Drop in Accuracy	
Classifier	New2News	Wiki2News	Wiki2Wiki	News2Wiki	Wiki2News	News2Wiki
knncosine	80.0	61.8	91.8	73.1	22.5	20.5
knneuc	65.4	27.5	75.1	57.7	56.7	21.6
naivebayes	69.4	34.0	78.5	56.8	50.2	27.2
svm	85.8	63.0	94.0	80.0	26.1	14.9

Table 6.3: The average drop in the accuracy in percentage across different category groups with the intersected vocabulary space between two domains, Newsgroups-Wikipedia.

analysis (PCA) etc. in our experiments. For all the experiments in this dissertation we use the reduced vocabulary, obtained by removing the rare words and the unique words. Table 6.3 and Table 6.4 show the average drops in the accuracy for the other two domain pairs, namely, Wikipedia-Newsgroups and Newsgroups-NYT respectively.

From the Table 6.3 and Table 6.4, we show a similar trend in the reduction of the accuracy of approximately 25% and KNN-Cosine and SVM being the best classifiers. Similar graphs to Figure 6.1 showing the drop in the cross-domain classification by conventional methods, for the Wikipedia-Newsgroups pair and NYT-Newsgroups pair are included in the Appendix A.

6.3 Conclusion

This chapter shows the drop in accuracy when two different domains are used as testing and training sets from four conventional classifiers. We also show that removing words that occurred less than 100 times in the two domains does not decrease the accuracy of the classification but reduces the vocabulary size by over 90%. This result is summarized in Figure 6.3. Figure 6.3 shows the results of cross-domain and single domain classification using entire vocabulary and the reduced vocabulary. The results show no change in the accuracy after removing the rare

	Accuracy				% Drop in Accuracy	
Classifier	NYT2NYT	News2NYT	News2News	NYT2News	News2NYT	NYT2News
knncosine	88.5	71.3	85.8	65.9	17.2	23.3
knneuc	78.4	59.7	56.6	35.3	18.7	37.8
naivebayes	75.2	63.7	77.3	50.7	11.5	33.7
svm	92.6	74.8	94.5	63.3	17.9	33.1

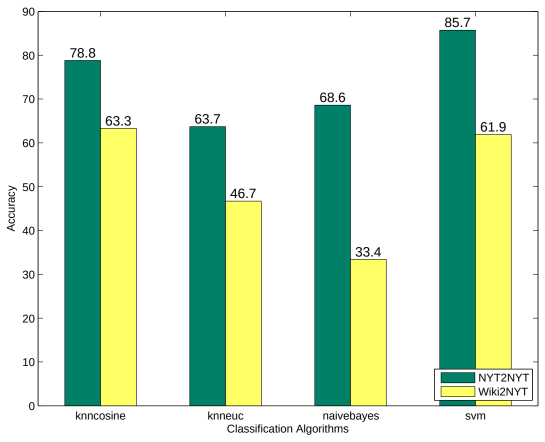
Table 6.4: The average drop in the accuracy in percentage across different category groups with the intersected vocabulary space between two domains, Newsgroups-NYT.

words that reduces the vocabulary by over 90%. The figure also shows a consistent drop in the accuracy by approximately 25% when two different domains are used for testing and training sets.

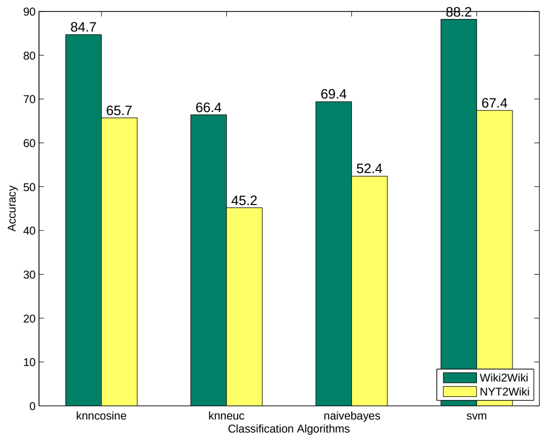
By intersecting the vocabulary of the two domains and removing the rare words, we achieve an almost no drop (less than 1% on average) in the accuracy and in some cases we even see an increase in the accuracy. This increase in the accuracy is not surprising as rare words can often result in a more noisy data and do not contribute to the classification patterns, thus reducing the overall accuracy [67, 68]. On the other hand, we do obtain an interesting result by observing that the classification accuracy increases for single-domain classification after removing the words that do not occur in both domains. This gives us another way to reduce the dimensions of text data for classification i.e., by not including the words that do not occur in another similar domain.

Given the results in Figure 6.3, we use the reduced vocabulary for all our experiments. We thus obtain a baseline measurement for the drop in accuracy by using the intersected vocabulary and primarily comparing our results with KNN-Cosine and SVM classifier. This baseline measurement of the drop in accuracy for NYT-Wiki domain is shown in Figure 6.4.

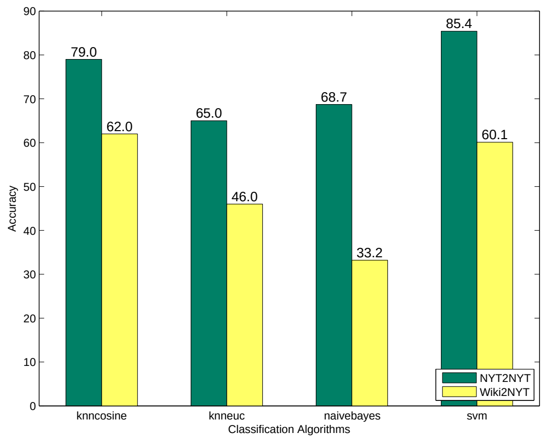
This chapter introduces and then elaborates on the problem of cross-domain document classification. We obtain empirical results showing a significant drop in accuracy for cross-domain classification while using the conventional classifying algorithms. We also summarize and list some of the reasons for this drop. Based on these reasons we show that removing the rare and unique words does not affect the accuracy and in some cases even increases the accuracy by a small amount. In the next chapter, we will introduce more of our algorithms based on topic models to deal with the problem of cross-domain classification.



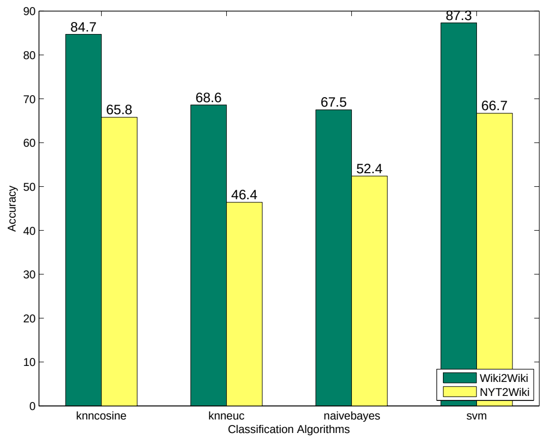
(a) NYT2NYT & Wiki2NYT - Entire Vocabulary



(b) Wiki2Wiki & NYT2Wiki - Entire Vocabulary

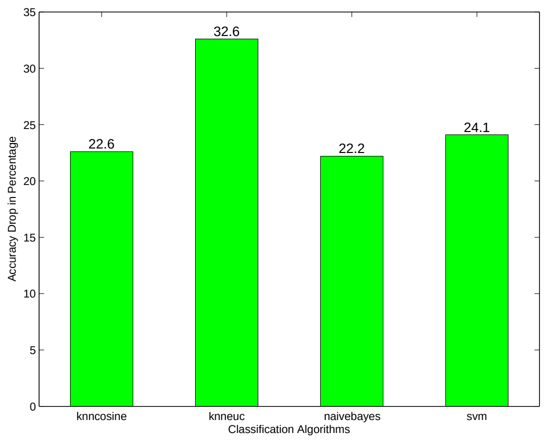


(c) NYT2NYT & Wiki2NYT - Reduced Vocabulary

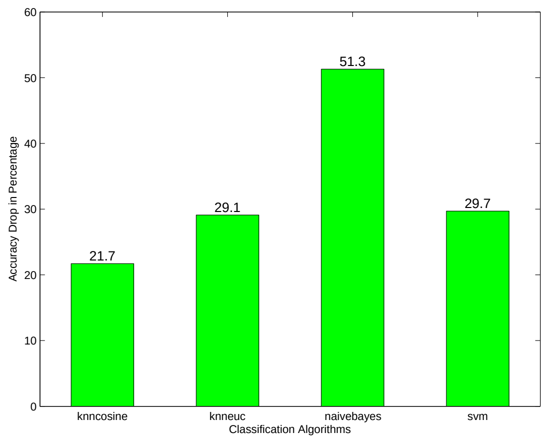


(d) Wiki2Wiki & NYT2Wiki - Reduced Vocabulary

Figure 6.3: The figure summarizes the results of cross-domain (yellow bar) and single domain (green bar) classification using entire vocabulary (top row) and the reduced vocabulary (bottom row). The results show no change in the accuracy after removing the rare words and a consistent drop of approximately 25% when two different domains are used for testing and training sets.



(a) NYT2Wiki Accuracy Drop



(b) Wiki2NYT Accuracy Drop

Figure 6.4: The figure shows the percentage accuracy drop in cross-domain classification from conventional classifiers for the domain pair NYT-Wiki. The NYT2Wiki drop is compared against Wiki2Wiki and Wiki2NYT drop is compared against NYT2NYT.

CHAPTER 7: Novel Cross-Domain Classification Algorithms

In this chapter we explain our cross-domain classification algorithms based on the topic models such as Latent Dirichlet Allocation (LDA). We develop these algorithms and formulations based on theoretical properties of various distributions as well as on our empirical evaluations of the topic models. We present these different algorithms and formulations for the cross-domain document classification and motivate each one with theoretical and empirical observations. Our algorithms, shown in this dissertation, are based on LDA, however, these algorithms can be extended to any other topic model with a few or no modifications.

We first motivate our choice of topic models as the basis of our cross-domain classification algorithms. Topic models, as introduced in Chapter 3, represent a generative model for a collection of document set. This generative model often assumes that the collection of documents covers a number of different topics, where each document in the collection may contain words belonging to one or more of these topics. The topic, in a technical sense, is a distribution over the words in the vocabulary, however, the topic can be assumed to have some semantic meaning by itself. For example, the topic that contains higher probability of words such as "genes," "diversity," "species" may point to a semantic topic of "theory of evolution."

Topic models give us a level of abstraction (or generalization) by extracting common topics from all the documents. Any document can then be represented in terms of these topics instead of specific words in the document. The documents on similar subjects in two different domains must share some common topics between them. A topic model such as LDA is a tool to extract these topics from the documents that are independent of the domain that the documents belong to. We expect that it is this generalization of documents that makes the transfer of learned information from one domain to another domain possible.

In this chapter, we first analyze different components of LDA that can help us use the topic level abstraction in classifying documents. We explain the process of obtaining the generalizable information content from the source domain that is then used for the cross-domain document classification. Then we go over two different ways of using LDA for classification, (1) using the vectors describing the distribution of topics in documents (γ vectors) and (2) using the topic as distribution of words (β vectors). We describe the methods and distance metrics for our methods and give empirical results.

7.1 Abstraction of the Source Domain Using Latent Dirichlet Allocation (LDA)

As previously discussed in more detail, given a set of documents where each document is a collection of words, Latent Dirichlet Allocation (LDA) model distributes the documents over k different topics and represents each document as a proportion of these topics. LDA model is defined by two parameters, namely α and β. α is the parameter for the Dirichlet distribution that is used to generate the topic distributions for individual documents, β is a set of multinomial distributions over the words for individual topics. Given the model parameters $\{\beta, \alpha\}$ to generate a document with N words $\mathbf{w} = \{w_1, w_2, \ldots, w_N\}$ under LDA; we first obtain a topic distribution, $\theta \sim$ Dirichlet(α); then generate word $w_n \in \{w_1, \ldots, w_N\}$ by first sampling a topic $z_n \sim Multi(\theta)$ and $w_n \sim Multi(\beta_{z_n})$, where $Multi$ represents the multinomial distribution function. Topic z_n is a multinomial distribution over V words, where V is the number of words in the entire vocabulary.

Equation 7.1 gives the expression for the probability of $\mathbf{w}$ using the notation of Blei et al. [11], where $\mathbf{w}$ is a word vector, given the model parameters α and β.

$$p(\mathbf{w}|\alpha, \beta) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta \tag{7.1}$$

Before we present the use of LDA for cross-domain classification, we will first go over our process of obtaining distributions (generalizable information content) from the source domain and our terminology for different components of our classification algorithms.

7.1.1 Abstraction of the Source Domain

In order to transfer the learning from one domain to another, we first need to extract generalizable information content of the categories in the source domain. This generalizable information content can also be treated as semantic information content defining the category. For example, if we want to classify documents about finance in the Wikipedia using the set of finance documents in NYT, we need to extract some semantic structure of finance documents. This semantic structure of finance category may be a list of topics that occur commonly under finance. These subtopics of finance can be terms like “lending,” “banks,” “investment,” etc. We choose LDA topic model as our underlying algorithm to extract this generalizable representation of the documents in the source domain. LDA extracts k multinomial distributions that best describe the set of documents presented to it. These k distributions are called topics and are represented by variable β in our framework. In LDA, each document can have words from multiple topics, where different words drawn from different topics. This makes each document contain different proportions of these k topics in it. These topic proportions or topic distributions in a document are represented by gamma (γ) variable in our framework. γ differs from θ (introduced earlier) as in θ represents a draw from a Dirichlet distribution parameterized by α, where α is the learned parameter of an LDA model of the entire document set. Whereas γ is a value of the Dirichlet parameter inferred from the model for a particular document. The γ vectors bring a major shift in our representation of documents from being in terms of words to in terms of topics. The topics obtained from the LDA (β vectors) give us an independent semantic structure and abstract information content of a category.

Figure 7.1 shows an illustration of different parts of our method to extract β and γ vectors from the source domain. Each shaded rectangular area represents a matrix. On the left of the figure, we start with labeled documents from the source domain. The figure uses the labels from the Arts category group as an example. There are total of c different categories. Documents from each category are modeled independently with LDA, thus we obtain k topics from each of the category. This gives us total of $c \times k$ topics (c sets of k topics). Each individual set of k topics is denoted as β_c, where c is the category label and all $k \times c$ topics are labeled as β_{source}, representing the topics obtained from the training set. We then assign the topic proportions to the documents in both target and source domains over these topics using the LDA inference. This is shown in the last two matrices, where the left matrix are the topic proportions obtained for the source domain documents (γ_{source}) and the right matrix is the topic proportions for the documents in the target domain (γ_{target}). Notice that both topic proportions, for training and testing set documents, are obtained using only the topics obtained from the source domain. The target and source domain may not have the same number of documents.

7.1.2 Obtaining Topics from the Source Domain, β Vectors

In order to understand the general nature of a category, we wish to learn the distribution of each category in the source domain. This distribution is obtained using the LDA model that is run independently on each category in the source domain. Let $L_{source} = \{1, 2, 3, \ldots, c\}$ be a set of c category labels in the source (labeled) domain. We learn k topics from the documents from each category independently, thus obtaining c set of k topics, $\beta_{source} = \{\beta_1, \beta_2, \beta_3, \ldots, \beta_c\}$, where each β_i is a set of k multinomial distributions (topics) over the words in the vocabulary.

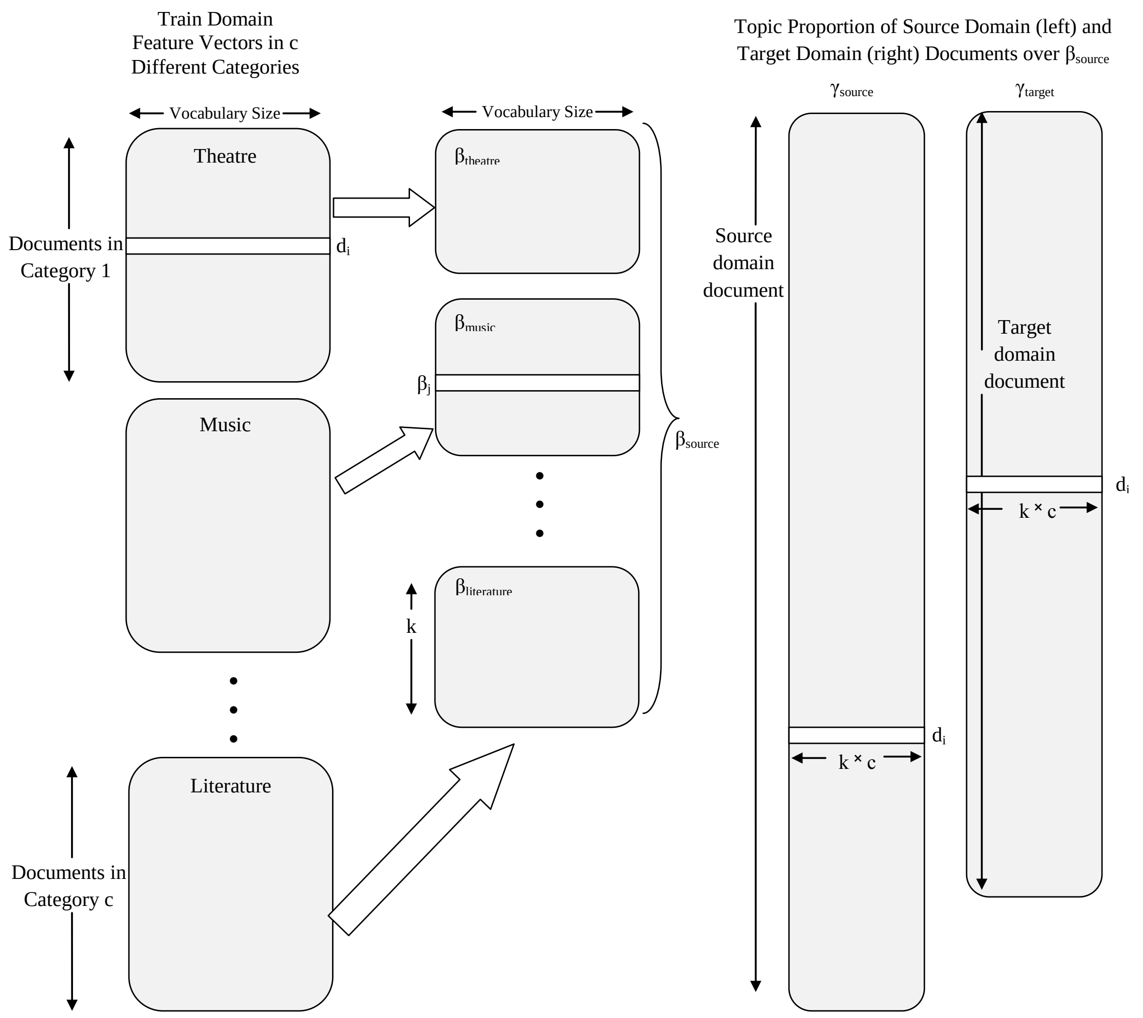


Figure 7.1: Different parts derived from the LDA for our cross-domain document classification framework. We obtain a set of k topics for each of the category in the source domain separately and then obtain the topic distribution of the documents in both target and source domain over all $k \times c$ topics, where c is the number of categories in our source domain.

Figure 7.1 also shows how a set of k topics is obtained from the documents of each category in the source domain. The illustration shows an example of the category group "Art," where the k topics are obtained from each of its six subcategories, namely theater, music, opera, film, television and literature. Given a set of β's, LDA model can also inference the posterior topic distribution in a document. This distribution of topic in a document is represented by γ vector (also shown in the figure). γ vector is a Dirichlet prior of the distribution of topics in a given document.

7.1.3 Obtaining Topic Distributions in a Document, γ Vectors

Once an LDA model has been learned from a set of documents, a β and an α are obtained that model the set of documents. This model can then be used to inference a topic distribution of any document and obtain a posterior distribution of a topics in that document. The γ vectors are the vectors that are Dirichlet parameter for these posterior distributions. A vector γ_i can then also be seen as a topic based representation of a document, d_i. Blei et al. use this topic distribution representation of the documents to classify the documents. We will have more discussion on using γ vectors for classification in the next section.

7.2 Topic Distribution (γ's) for Cross-Domain Document Classification

In this section, we will discuss one of the ways of using topic models for classification, that is by comparing the documents in training and testing set using the topic distribution in documents. The topic distributions in documents are represented by the γ vectors. A γ vector for a document is a $c \times k$ dimensional vector, where c is the number of categories in the training set, and k is the number of topics (β) obtained for each category in the training set individually using LDA model. We concatenate the k topics obtained from each of the categories in the training set and then run the LDA inference on each document from the training set and the testing set using this concatenated set of topics. In this way, we generate the topic proportions over the documents in the source domain and target domain over the topics obtained from the source domain categories. Figure 7.2 shows this model using the Bayes network for LDA model.

In Figure 7.2, the β_i represents the set of k topics and α_i represents the Dirichlet parameter as learned by the LDA model from the i^{th} category in the training set. The combined value of β (β_{train}) is generated by concatenating all the individual β's and a common α (α_{train}) is generated by taking the average of individual α's.

Once we obtain the $c \times k$ topic proportion vectors, one can explore different ways to classify these the documents represented as these topics proportion vectors. Blei et al. use SVM classifier in their paper to use the topic proportion vectors to classify documents. We experiment with a few different classifiers, including KNN with different distance measures, SVM, and classifier based on words assignment for each category. For KNN classifier, we use four different distance measures. The four distance measures chosen, all make a reasonable choice for the distance measures for the topic proportion vectors. In addition to using two commonly used

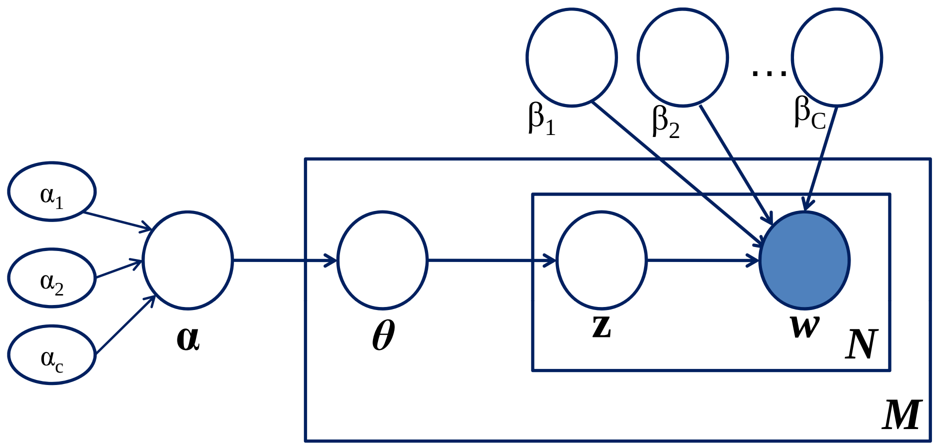


Figure 7.2: A Bayes network showing an LDA model as used to obtain posterior topic distribution (γ) vectors from the target and source domain. Each γ vector is a Dirichlet parameter for the topic distribution over the topics in the set $\beta_s = \{\beta_1, \beta_2, \beta_3, \ldots, \beta_c\}$, where each β_i contains k topics. Topics (β) are combined by concatenating all the topics obtained from the training set categories and α parameter is combined by taking the average of individual αs as obtained by LDA from each training set category.

distance measures in machine learning, namely Euclidean and Cosine distance measures, we use KL distance to compare the distributions that are expressed by the γ vectors. A γ vector is the proportion of the topics in a specific documents. One can think of it as a multinomial distribution over the topics. Therefore, we choose our third distance to be KL distance. However, γ vectors under LDA represent the variational approximation of the Dirichlet parameter α. The KL distance for two d-dimensional Dirichlet distribution as a function of their parameter α, (s and t) is shown in Equation 7.2 [71]. In the Equation 7.2, Γ is known as the gamma function and ψ is known as the digamma function. Although, γ vectors can sometimes be treated as a multinomial distribution especially when the sum of the its elements is large (reducing the variance of resulting Dirichlet distribution), as our fourth measure, we do use a KL distance expression specifically for Dirichlet distribution as a function of its parameter, α.

$$KL(Dir(s)||Dir(t)) = \log \frac{\Gamma(\sum s)}{\Gamma(\sum t)} + \sum_{i=1}^{d} \frac{\log(\Gamma(s_i))}{\log(\Gamma(t_i))} + \sum_{i=1}^{d}((s_i - t_i)(\psi(s_i) - \psi(\sum(s)) \quad (7.2)$$

We use the Equation 7.2 to compute the distance between the two γ vectors as our fourth choice for distance measures. The KL-distance for the Dirichlet distribution has recently been used in a few other classification tasks by Hoffman et al. and Blei et al. [72, 73]. Table 7.1 summarizes the four distance measures that we use and their corresponding reasons why we use these in the classification of documents using the topic proportion (γ) vectors.

Distance Measure	Reason
Euclidean	Most intuitive and widely used distance measure.
Cosine	A distance commonly used in text document classification.
Kullback-Liebler (KL)	A distance between two multinomial distributions.
Dirichlet KL	KL distance derived for Dirichlet distributions. (Equation 7.2)

Table 7.1: List of four distance measures used, and the reason for the choice, to compare the γ vectors.

We present our empirical results by using the KNN classifier with these four distance measures in Figure 7.3. All four distance measures vary in their performance depending on the dataset and category group used. However, the KL distance for multinomial distributions does perform better than others in most cases. The KL-distance for Dirichlet distributions also has similar performance to Cosine and Euclidean measures. Some of the variations in the KL distances may be due to the fact that these distance measures are too sensitive to small values, thus resulting in a log of zero. Some of these problems may be fixed by smoothing the data by using methods that are often used to smooth the data in Naive Bayes classification.

Another important thing to note in these results is that we do not get a higher cross-domain classification accuracy by using only the topic proportion vectors for documents. This result is not surprising as the topic proportion vectors use less than 1% of the dimensions as the original word counts vectors. The fact that the performance is almost as good as a cosine distance measure used on the word counts vector based on entire vocabulary is still an encouraging result.

We also classify the topic proportion vectors using SVMs. The results of SVM classifier for cross-domain document classification are shown in Figure 7.4. The topic proportion vectors as obtained from PLSA or LDA model have a lot of flexibility in terms of topic association as each document can belong to various topics. It is this variation that yields documents that are not neatly separated into classes, even though underlying topics are obtained from different categories. We seek to formulate distance measure that reduces this variation, especially variation among the topics from the same category. In a topic proportion vector the most important element for classification is the weight or the number of words assigned to the topics from different categories. How the weight is distributed among the topics from the same category is not as important. We thus create a vector that is made of the percent of words assigned from each topic set, where each topic set is the set of k topics obtained from a single training category. This, in turn, reduces the dimensions of the feature vector even further as all the elements belonging to the topics of the same category are added up together. The document is classified into the category that most of its words were assigned to. For example, in a topic proportion vector of 120 dimensions, where 20 topics were obtained from 6 different categories in this classifier we will add up the pack of 20 elements and will assign the document to a category that yields the highest total.

Figure 7.5 shows the empirical results of using the word assignment weight of the PLSA and LDA (PLSA Category Weight, LDA Category Weight) topic proportion vectors. The results obtained are compared against the KL-distance KNN classifier on the γ vectors and the Cosine distance on KNN on the feature counts vector. In Figure 7.5, we observe that in most cases, the category weights obtained from the topic proportions outperform both the KL-distance classifier on the γ vectors and the cosine distance classifier on the word counts. Although, the improvement is either not significant in magnitude or consistent across different datasets and category groups. We point out three possible reasons for the lack of significant improvement over the conventional methods when we only use the topic proportion vectors. We tabulate these three reasons as follows:

1. Enormous reduction (more than 98%) in the dimensions of the document. This reduction, although beneficial in some cases, also causes consolidation or loss of important information about the document category.

2. Flexibility in LDA and PLSA model to choose the topic proportion that best fits the document. This flexibility enables the model to provide topic proportion vectors that can vary greatly in their topic assignments, especially when the underlying topics obtained from different categories also show some overlap. We come back to this point in our next section again.

3. Comparisons are still being made among the documents from different domains. Ideally, we should be able to learn some general information regarding the category and compare test documents from a new domain to this general learned model of a category.

Our experiments with the topic proportion give us insights into the nature of topic distributions. In the next section, we explore the nature of the topics and their distribution by using clustering algorithms such as k-means and hierarchical clustering. We then develop algorithms based on using the topics themselves to improve the cross-domain document classification accuracy.

7.3 Clustering of Topics Obtained From the Source Domain

This section describes our analysis of the distribution of the topics obtained from different categories using clustering method. This analysis, although important in its own right, is a small digression from our main point of the chapter, that is describing the use of LDA model for cross domain classification. Readers may choose to skip this section in the interest of continuity as it explains a diagnostic test on the distribution of topic vectors and provides motivation for our classifier formulations derived later in the chapter.

As discussed earlier, we obtain k topics from each of the category in the source domain independently. What we hope to extract are the subtopics of each category. So the k topics from each of the category will have different focus on the words that may point to a subtopic. There may be some overlap, as both categories "music" and "opera" may have "performance" or "singing" as their subtopic, but in general we will see a set unique subtopics for each of the category. In other words, we expect the k topics obtained from each category to belong to its own cluster. We investigate this by clustering the topics using k-means and hierarchical clustering. When we use k-means, we cluster all the topics in c different clusters, where c is the number of categories.

Figure 7.6 shows the clustering analysis of the 300 topics obtained from the "Arts" category group, with 50 topics obtained independently from each of its six subcategories. The top row of the image shows the result of k-means clustering, where k for k-means was chosen to be 6. The left most color bar shows the cluster assignment of each point, where the cluster number is encoded with a unique color shown in the color bar next to it, for example, cluster number 1 is dark blue and cluster number 5 is orange etc. The middle figure shows the cluster numbers sorted in increasing order and thus showing the size of each cluster, we see that dark blue cluster is contains a little less than 50 points, whereas the yellow cluster contains approximately 75 points. We also notice a pathological case of k-means where cluster number 6 (maroon color) contains only one point. The right most figure on the top row shows the actual composition of each cluster obtained. This figure can be matched with the middle figure, where each cluster is overlapped with the actual color coded 300 points. So we see that most of the dark blue points that make up topics 1 to 50 are in cluster 3 (magenta color) with some in the yellow cluster. Ideally, we would want to see exactly one color shade in each cluster and each cluster of about 50 points in size. However, as noted earlier, there is bound to be some overlap between the subtopics of a category and there are some subtopics that represent background or general subtopic that may be common to all categories. In addition to this, we also realize that the k-means algorithm is susceptible to finding a suboptimal clustering, such as producing clusters with only a single point in it. The k-means analysis is encouraging as it does show a good separation among all 6 categories in this high dimensional space. To augment the k-means analysis, we also produce two hierarchical clusterings of the topics using Euclidean and Cosine distance measures. The hierarchical clustering trees are shown in the bottom row of Figure 7.6. Individual points representing topics on the x-axis are labeled using a single character such as "|" or ")" to save space. The labels correspond to a single category, so a vertical bar, "|" is the label for "music", so all the topics obtained from category music are labeled with the vertical bar. What is important to note in the figure is that same labels are clustered together with some patches of mixed area. For example, we see that cluster of "|" on the right of the tree generated using the Cosine measure and on the left of the tree generated using the Euclidean measure. Figure 7.7 also shows the clustering analysis of the Science category group with 100 topics obtained from each of its categories. Science category group contains six subcategories, giving us the total of 600 topics. The Figure 7.7 is laid out in similar manner as the Figure 7.6. We can see 6 clusters in the hierarchical trees. We also see that one of the leaf is separated out far from all the other points, thus giving rise to a singleton cluster in the k-means analysis.

We ran these experiments on all of our data sets with number of topics ranging from 20 to 100 from each category. A few more of these graphs with the domain pairs of Wiki-Newsgroups and NYT-Newsgroups have been included in the Appendix. In each experiment, we do observe the shape of the hierarchical clustering tree and k-means graphs that show some separation among the clusters based on the underlying categories. The presence of these clusters, separated according to the category, is an encouraging result that suggests using topics (β vectors) directly for classification may be useful. In the next section, we will go over in detail about the distance metrics and other formulations for the cross-domain classification using these topic vectors.

7.4 Using Topics (β's) for Cross-Domain Document Classification

Earlier, we saw that using γ vectors, topic distribution vectors, does not give a higher cross-domain classification accuracy than just comparing the word counts of the vectors. This may be due to the fact that while using the γ vectors, we still essentially compare the documents from the source domain to the documents in the target domain, albeit, the documents are represented in terms of topics instead of words. In order to achieve a higher accuracy, we do not need to compare the documents in the target domain directly to the documents in the source domain, but compare the target domain documents to a more generalized knowledge obtained from the source domain. This generalized knowledge, in our case, is given by a set of multinomial distributions that a certain category represents.

This concept of using the generalized distribution of the categories led us to do the clustering analysis on the topic vectors as shown in the previous section. The results of clustering analysis do show some separation among the topics obtained from different categories. These preliminary results support our hypothesis that the LDA topic model run on individual categories produces a set of unique subtopics that extract the distinctive word distribution of the category.

In this section, we will develop formulations for classification and distance metrics that can be applied to these topics obtained using the LDA model. The probability of a document (represented as a word vector) given α and β, $p(\mathbf{w}|\alpha, \beta)$, is computed by integrating the probability of $\mathbf{w}$ given θ and β (Equation 7.3) over all possible values of θ given the Dirichlet parameter α, where θ is a draw from the Dirichlet distribution parameterized by α.

$$p(\mathbf{w}|\theta, \beta) = \prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \tag{7.3}$$

According to the notation used by Blei et al. [11], β_{ij} is the j^{th} element of i^{th} topic and furthermore, each word w_n is unit basis V-dimensional vector that has exactly one component equal to 1 and all other as 0. Since w_n^j equal to 1 only when $j = n$, we reduce the product $\prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j}$ as $\theta_i \beta_{i,n}$, where $\beta_{i,n}$ is the n^{th} element of i^{th} β or $p(w_n|\beta_i)$. In our convention, we use an extra comma in the subscript of β to denote the words within a topic, for example, j^{th} element of i^{th} topic will be written as $\beta_{i,j}$ instead of β_{ij}. Equation 7.4 gives us the short formulation to compute the probability of a word vector given θ.

$$p(\mathbf{w}|\theta, \beta) = \prod_{n=1}^{N} \sum_{i=1}^{K} \theta_i \beta_{i,n} \tag{7.4}$$

Equation 7.4 gives us a simplified way to compare two documents under LDA if we assume that θ is not a random variable but a fixed known value. However, θ, under LDA, is a draw from a Dirichlet distribution parameterized by α and it is thus a random variable. We only know the distribution of θ under the LDA model, not its exact value. In order to evaluate the probability of a word vector given only α and β, we will have to use the Equation 7.1. However, once we obtain the set of k topics from each of the category, various formulations of an effective distance metric are possible that do not require inferencing the value of α or θ (or their variational approximation) using the EM algorithm. We will present these different formulations and give the empirical results in the following subsections.

Figure 7.8 shows how a document from the test set (taken from the target domain) is compared with the topics obtained from the source domain. As the dimensionality and the vocabulary used for the documents is same as that of the topics, they can be compared directly by using any standard distance measure. In next section, we introduce different formulations (based on properties of topic models and distributions) for comparing topics with the documents and we show the corresponding empirical results. We run Wilcoxon rank sum statistical tests to show the statistical significance of our results [74]. These results are shown in appendix A of this dissertation.

7.4.1 Document Topic Comparison Formulations

In this subsection, we will introduce four different formulations for classifying the target domain documents by using the topics from the source domain. As we shall see, these formulations are developed with different assumptions about the underlying model for generating the target domain documents using the source domain topics. We will also give a few different interpretations of these formulations, whenever appropriate.

Formulation 1: Using Single Topics To Generate Target Domain Documents

In our first formulation for a cross domain document classifier based on the source domain topics, we assume that each document is generated from only one topic. This, in a way, is the simplest formulation of the target domain documents in terms of the source domain topics. To use this formulation, where a single topic is chosen to be responsible for generating a given document, we need to develop distance metrics that are appropriate to compare the topic, represented as a multinomial distribution over the entire vocabulary, and the documents, represented as word count vectors. We will concentrate on two main distance measures, namely Kullback Leibler distance (KL distance) and cosine distance measure.

Another way to interpret the assumption, that a single topic from the source domain generates a target domain document, using the LDA framework is by assuming that the value of α is chosen to be significantly less than 1 and is close to 0. The Dirichlet distribution with the α

parameter approaching zero gets concentrated on the corners of the simplex. Samples from such a distribution will contain all the probability mass of the sampled multinomial distribution on a single value in its parameter vector. So the parameter vector of the sampled multinomial distribution has value of 1 in exactly one of the places and 0's in all other places. Even though the constraint of α being close to zero (assigning all the words to one topic) may seem too limiting at first glance, in practice however, α is in fact very small and a 90% of the document is assigned to 5% of the topics.

We derive the likelihood expression for a document under this formulation using the following assumption about the value of α under LDA model.

$$\alpha \ll 1 \tag{7.5}$$

The LDA model, in this case, approaches the mixture of multinomial distributions model, where all the words are assigned to a single topic. We can start with our simplified LDA probability computation when we are given a value of θ as follows:

$$p(\mathbf{w}|\theta_i, \beta) = \prod_{n=1}^{N} \sum_{i=1}^{K} \theta_i \beta_{i,n} \tag{7.6}$$

We know that $\theta_i = 1$ and $\theta_j = 0$ for all $i \neq j$. Using this, we obtain the following reduced expression for the probability of the document.

$$p(\mathbf{w}|\theta, \beta) = \prod_{n=1}^{N} \beta_{i,n} \tag{7.7}$$

In Equation 7.7, we obtain the likelihood expression for the document, w, given a topic under this formulation. We will use KL-distance to compare the target domain documents with the source domain topics. We will lay out the relationship between the likelihood expression and the KL-distance. This relationship will make it easy for us to develop all the distance metric expressions in this section that we use for the cross-domain classification. KL distance between two probability distributions P and Q is given as follows:

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}. \tag{7.8}$$

We have shown earlier in Chapter 3 that KL-distance (also known as KL-divergence) is the natural distance for the multinomial distributions. It is an appropriate distance to compare two distributions under the framework of information theory as it is the difference in the number of bits that one needs to encode a message by using distribution Q, when that message is originally distributed according to P.

In this section, we will show a relationship between the likelihood expression of a document given a topic, β_i and the KL-distance between the document and topic, β_i. We will motivate our choice of KL-distance as a substitute for computing the probability of the word vector as shown earlier. Distribution Q is the maximum likelihood estimate for points that are being generated using the true distribution, P, when the KL-distance between P and Q is minimum [75, 76].

If $\mathbf{w}$ is a set of words in a document and $\mathbf{w_n}$ represents the count of the n^{th} word, and the probability of $\mathbf{w}$ is given as follows:

$$p(\mathbf{w}|\beta) = \prod_{n \in \mathbf{w}} \beta_{i,n}^{\mathbf{w_n}} \tag{7.9}$$

Then we obtain the following relationship between the likelihood and KL-distance:

$$\underset{\beta_i}{\text{argmax}}\ p(\mathbf{w}|\beta_i) = \underset{\beta_i}{\text{argmin}}\ KL(\mathbf{w}\|\beta_i) \tag{7.10}$$

Proof: Taking the log of both sides, we get an expression for the log-likelihood.

$$\log p(\mathbf{w}|\theta, \beta) = \sum_{n \in \mathbf{w}} \mathbf{w_n} \log \beta_{i,n} \tag{7.11}$$

We show that the KL-distance between $\mathbf{w}$ and β, $KL(\mathbf{w}, \beta)$, is minimum for the β that maximizes the above likelihood expression.

$$\begin{aligned} KL(\mathbf{w}, \beta) &= \sum_{\mathbf{w}} \mathbf{w_n} \log \left(\frac{\mathbf{w_n}}{\beta_{i,n}} \right) && (7.12) \\ &= \sum_{n \in \mathbf{w}} \mathbf{w_n} \log \mathbf{w_n} - \mathbf{w_n} \log \beta_{i,n} && (7.13) \end{aligned}$$

From the last equation, we see that the expression $KL(\mathbf{w}, \beta)$ is minimum when $\sum_{n \in \mathbf{w}} \mathbf{w_n} \log \beta_{i,n}$ is maximum.

Using the result obtained in Equation 7.10, we use the KL distance to compare the documents with the topics using the KNN classifier. We compute the nearest neighbor of the documents in the set of all $k \times c$ topics obtained using the LDA as described earlier. The number k for the k-nearest neighbor is chosen to be same as k for the number of topics obtained from each category.

We show an illustration of this process in Figure 7.9. Figure 7.9 shows topics and the document as points distributed on a word simplex. A document with normalized word counts is compared to the topics using KL-distance.

Figure 7.10 shows the results of our cross-domain algorithm that is based on the KL-distance for comparing the documents to the source domain topics. The figure shows three bars (1) blue bar, is the cross-domain classification using conventional KNN with word count vectors using cosine distance measure, (2) green bar, is our algorithm where the target domain documents are compared with the topics obtained from source domain using KNN classifier with KL-distance, (3) red bar, is the single domain classification where both target and source domain are same and classification is done using KNN with word count vectors using the cosine distance measure. We compare the conventional method (blue bar) with our method (green bar) and observe that the classification by using the k-nearest neighbor classifier when the topics are used as the training set (LDA-KL) outperforms the classification by the k-nearest neighbor using the word count vectors (KNN-Cosine). The results are consistent across all three pairs of domains (Wikipedia, NYT and Newsgroups) with each domain serving as a source and target domain.

For the second distance measure, we consider cosine distance measure, which can be described as $1 - \cos(\mathbf{w}, \beta_i)$. The motivation for this distance measure comes from the fact that each topic can itself be treated as a normalized word count vector. Since cosine measure is a commonly used measure to compare two documents, we can use the same measure to compare the document with a topic.

Figure 7.11 compares the cross-domain classification accuracy obtained from the two distance measures, Cosine and KL-distance, that we propose to compare the target domain documents to the topic vectors obtained from source domain categories. The graphs show results of all different category groups explained in Chapter 4, such as “Arts,” “Computers” etc. The blue bar in the figure is the accuracy obtained using the cosine measure and the yellow bar is the accuracy obtained using the KL-distance. Although, both measures have valid reasons to be used, we see that KL-distance measure does outperform the cosine distance measure (yellow bar is higher than blue bar) in most cases. Figure 7.11 shows the results for all three pairs of domains (Wikipedia, NYT and Newsgroups) with each domain serving as a source and target domain.

Formulation 2: Using Likelihoods Computed by LDA and PLSA Models

In our second formulation, we assume that the target domain documents are being generated by the LDA model and the PLSA model given the complete set of β's obtained from a given source domain category. For LDA, this case assumes a more realistic value of α to be less than one, but not close to 0. This results in a distribution that assigns the words in the document to three to five different topics.

When using the value of alpha less than 1, we get the following Equation 7.14 for probability, where a few values of the θ vector are non-zero. This is an equation that is used in the PLSA topic model as it does not have any Dirichlet prior over θ. To compute the value of θ that maximizes the log likelihood in the Equation 7.14, we will have to solve the equation with logarithm of sums.

$$p(\mathbf{w}|\theta_i, \beta) = \prod_{n=1}^{N} \sum_{i=1}^{K} \theta_i \beta_{i,n} \tag{7.14}$$

In case of PLSA, we use the Jensen's inequality and EM algorithm to maximize the log likelihood for each individual set of β's, by using the following E-step and M-step. For LDA, we use the variational approximation of the parameters using the formulations laid out in the paper by Blei et al. [11].

The empirical results for this formulations are shown in Figure 7.12.

The empirical results do not show an improvement in the classification over conventional algorithms. This is not surprising, as using the EM algorithm and PLSA model to adjust θ vector for each set of β introduces too much flexibility in the likelihood expression. In other words, each set of β vectors contains enough combination of topics to generate high likelihoods, thus reducing the effect of individual topics in the set. This result is similar to the result obtained when the entire LDA model is used to classify the documents.

It is also not surprising to see that the LDA results given lower classification than the PLSA results, when the topic distribution of the documents are fitted for each set of β vectors independently. In the PLSA model the EM algorithm is used to obtain the empirical θ based only on the words in the document without Dirichlet prior. The LDA model is more flexible than the PLSA model as it only assigns a probability distribution over the topics. In other words, the more adjustable or flexible the model is to a document, the less classification accuracy it will have, as it is flexible enough to assign a high likelihood to previously unseen documents. This, however, should not be seen as a disadvantage of LDA or PLSA, on the contrary, this flexibility is a key aspect of the topic models. The documents that we are trying to classify here belong to one category group, thus belong to similar categories. This similarity in the categories generate topics that have a considerable overlap, for example, out of 20 topics obtained from Music and Opera categories, 5 to 10 topics from these two categories may be similar. That is to say that a topic that puts a high probability on words such as "performance," "singing" etc. will be present in both categories. If we fit a document belonging to category "Opera" according to the topics of category "Music," the topic model algorithm will be able to find a combination of small subset of topics from Music category that give high likelihood to this document from Opera category. This fine tuning of the weights on different topics for each document impedes the differentiating ability of a set of topics that is obtained from different categories of source domain.

Our next two formulations use the entire set of topics to generate the document, where each topic in the set is weighed equally. We propose two different ways of using the entire set of topics for generating the document, (1) using LDA framework and (2) mixture of multinomial framework.

Formulations 3 and 4: Using All Topics Equally To Generate Target Domain Documents

In our third and fourth formulations for a cross domain document classifier based on the source domain topics, we assume that each document is generated from all the topics obtained from a source domain category, where each topic was equally weighed by the generating topic model. We develop two ways of generating the target domain documents using the source domain topics; (1) using the LDA model, (2) using the mixture of multinomial models. Each formulation produces a slightly different KL-distance based metric calculation. We derive the metrics, give empirical results and show a geometric interpretation of both of these formulations in this section.

Formulation 3: Using All Topics Equally Under LDA Framework

We develop our third formulation based on the LDA framework and use all topics equally to generate the target domain documents. One way to interpret this, in terms of α, is by assuming the value of α is to be significantly larger than 1. The Dirichlet distribution with the α parameter much larger than one yields a uniform distribution over the simplex, making any combination of topics equally likely.

A large value of α makes the document equally distributed among all the topics. On the simplex of topics, a large α puts almost all the probability in the center of the simplex. We derive the likelihood expression for a document under this formulation using the following assumption about the value of α under LDA model.

$$\alpha \gg 1 \tag{7.15}$$

Starting with our original simplified expression for the likelihood under the LDA model, we have

$$p(\mathbf{w}|\theta,\beta) = \prod_{n=1}^{N}\sum_{i=1}^{K}\theta_i\beta_{i,n} \tag{7.16}$$

Since with the assumed value of α to be greater than 1, we know that $\theta_i = \theta = \frac{1}{K}$ for all $i \in \{1,\dots K\}$. Using this we get,

$$p(\mathbf{w}|\theta,\beta) = \prod_{n=1}^{N}\frac{1}{K}\sum_{i=1}^{K}\beta_{i,n} \tag{7.17}$$

Using our word vector notation, where $\mathbf{w}$ is a set of words in a document and $\mathbf{w_n}$ represents the count of the n^{th} word, we get the following expression for the probability of $\mathbf{w}$:

$$p(\mathbf{w}|\theta,\beta) = \prod_{\mathbf{w}}\left(\frac{1}{K}\sum_{i=1}^{K}\beta_{i,n}\right)^{\mathbf{w}_n} \tag{7.18}$$

Taking log of both sides,

$$\log(p(\mathbf{w}|\theta,\beta)) = \sum_{\mathbf{w}}\mathbf{w}_n\log\left(\frac{1}{K}\sum_{i=1}^{K}\beta_{i,n}\right) \tag{7.19}$$

Using the result shown in Equation, 7.10, for a given document, $\mathbf{w}$, we get the following expression,

$$\underset{\beta_i}{\text{argmax}}\ p(\mathbf{w}|\beta_i) = \underset{\beta_i}{\text{argmin}}\ KL\left(\mathbf{w}||\left(\frac{1}{K}\sum_{i=1}^{K}\beta_i\right)\right) \tag{7.20}$$

we can also write this as follows, where $\bar{\beta}$ is mean of β.

$$\underset{\beta_i}{\text{argmax}}\ p(\mathbf{w}|\beta_i) = \underset{\beta_i}{\text{argmin}}\ KL\left(\mathbf{w}||\bar{\beta}\right) \tag{7.21}$$

Given the result in Equation 7.21, the distance metric that computes the distance between a document and a set of source domain topics is the KL-distance between the mean of the topics and the normalized word count vector for the document. This is a centroid based distance used in the hierarchical clustering. Figure 7.13 shows an illustration of this metric on the word simplex. The figure shows the distance from a document to a set of topics, which is computed as the KL distance between the document and μ_β, where μ_β is the mean of all the topics obtained from a single source domain category.

The empirical results for this are shown in Figure 7.14.

Formulation 4: Using All Topics Under Mixture of Multinomial Framework

In formulation 3, in the previous section, we use the LDA framework to generate the documents assuming a large value of α. Under LDA framework, different words in the document can be assigned to different topics. Thus, the likelihood expression results in a product of sums, where the each word is chosen independently, and then the topic is chosen for that word.

In our fourth formulation, we choose the mixture of multinomial framework, where all the words in a document are assigned to the same topic, but the document can be generated using any one of the topics with equal probability. The likelihood expression of the document under this model is given as follows:

$$p(\mathbf{w}|\theta, \beta) = \sum_{i=1}^{K}\theta_i \prod_{n=1}^{N}\beta_{i,n} \tag{7.22}$$

where θ is the vector signifying the weight given to each topic or the probability of that topic as the topic generating the document. Using our assumption of a uniform θ, we obtain the following simplification of this likelihood expression:

$$p(\mathbf{w}|\theta, \beta) = \frac{1}{K} \sum_{i=1}^{K} \prod_{n=1}^{N} \beta_{i,n} \tag{7.23}$$

Taking log of both sides to compute the expression for the log likelihood, we obtain:

$$\log\left(p(\mathbf{w}|\theta, \beta)\right) = C + log\left(\sum_{i=1}^{K} \prod_{n=1}^{N} \beta_{i,n}\right) \tag{7.24}$$

Using the Jensen's inequality, by adding a new distribution $q(\theta)$, using the EM-algorithm framework, we obtain the following result:

$$\begin{aligned}
\log\left(p(\mathbf{w}|\theta, \beta)\right) &= C + \log \sum_{i=1}^{K} q(\theta_i) \frac{\prod_{n=1}^{N} \beta_{i,n}}{q(\theta_i)} && (7.25)\\
&\geq C + \sum_{i=1}^{K} q(\theta_i) \log\left(\frac{\prod_{n=1}^{N} \beta_{i,n}}{q(\theta_i)}\right) && (7.26)\\
&= C - \sum_{i=1}^{K} KL\left(q(\theta)|| \prod_{n=1}^{N} \beta_{i,n}\right) && (7.27)\\
&= C - \sum_{i=1}^{K} q(\theta)\left(\log q(\theta) - \log\left(\prod_{n=1}^{N} \beta_{i,n}\right)\right) && (7.28)
\end{aligned}$$

We obtain the Equation 7.28 by using the definition of KL-divergence as defined in Equation 3.40. In the Equation 7.28, we observe that only one expression has β_i in it and to minimize this expression, we can use our earlier derived result, Equation 7.10 that relates the likelihood with the KL distance. In this case, to maximize the likelihood of a document under a mixture of multinomial model, when all the multinomial distributions (β) are equally weighed (uniform θ vector), we minimize the sum of the KL distances between the document and the multinomial distributions. The multinomial distributions are the topics obtained from the source domain categories. We obtain the following result:

$$\underset{\beta_i}{\text{argmax}}\ p(\mathbf{w}|\beta_i) = \underset{\beta_i}{\text{argmin}}\ \sum_{i=1}^{K} KL(\mathbf{w}\|\beta_i) \tag{7.29}$$

Figure 7.15 shows an illustration of this metric on the word simplex. The figure shows the distance from a document to a set of topics is computed as the sum of KL distances between the document and all the topics obtained from a single source domain category.

The empirical results for this are shown in Figure 7.16.

7.4.2 Using β Vectors for Same Domain Classification

We show that our formulations, (1), (3) and (4) give us much better cross-domain classification when compared to the conventional methods such as K-nearest neighbor using the counts feature vector. In this section, we investigate how these formulations perform for doing the same domain classification. We carry our experiments by splitting the documents in each domain into training and testing set by using 80% of the documents for training and 20% for testing. We do 5 folds cross validation by using this 80-20 split. We train an LDA model on the documents in the training set for each class independently and thus obtaining the 20 β vectors for each class. We then classify 20% of the test set documents using the obtained topic vectors using three of our formulations. We see that a conventional KNN, trained on word counts based feature vectors, does outperform the LDA based KNN formulations in almost every case. This result is expected as a counts based feature vector retain more information about the training set documents and is better suited to train a classifier that is to be used for the documents in the same domain.

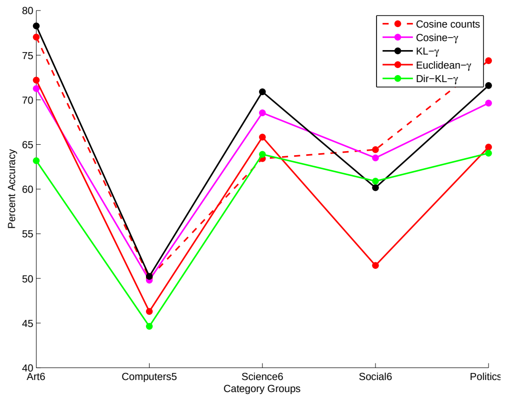
The empirical results for this are shown in Figure 7.17.

7.4.3 Support Vector Machines For Topics

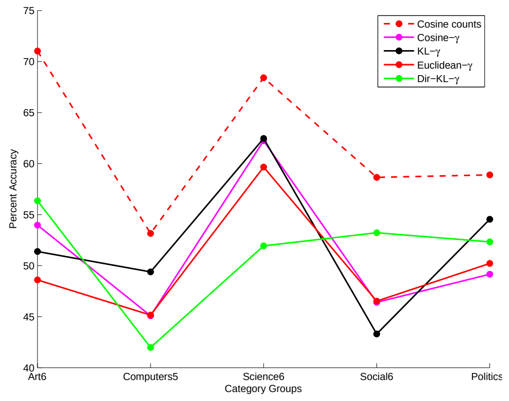
In the previous section, we discussed four different metrics based on different assumptions for the model generating the target domain document. The metrics derived in the previous sections provided a nearest neighbor based classifier and that could also be interpreted using hierarchical clustering analysis. Instead of only using the nearest neighbor based approaches as mentioned in the previous subsections, we can also run a classifier on the topics themselves. Based on the observation that the topics show a separation on the simplex, a classifier such as support vector machines (SVMs) should also be able to find meaningful separating boundaries on the simplex, separating topics obtained from different categories. Figure 7.18 shows the results of using SVM for the cross-domain classification, where the blue bar is the result by conventional SVMs training on the word count vectors and the green bar is the SVM training on the topics obtained using LDA.

The results obtained using the SVM on topics give us better cross-domain accuracy than the SVM classifier results obtained using the conventional word count feature vectors. The separating planes found by the SVM classifier are based on the distribution of the topics on the simplex. Even though, the number of topics is a lot less than the number of documents in the source domain, the separation among these few topics provides us better separation among the categories that can be transferred across different domains. Figure 7.19 shows an illustration of the SVM classifier output on the word simplex.

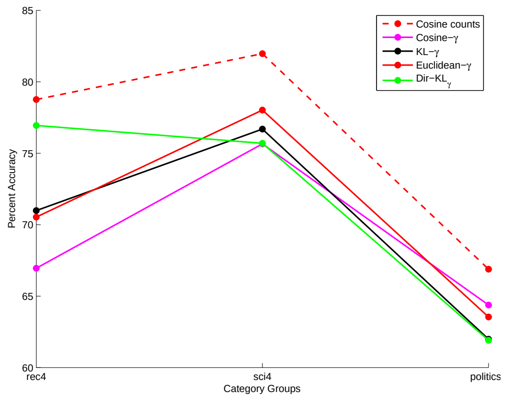
As shown in the previous section, we also can use this classifier using the support vector machines on topics obtained by LDA for a single domain classification. Figure 7.20 shows the results of using the LDA SVM on the same domain and compares it with the conventional SVM on a single domain. We see that a conventional SVM, trained on word counts based feature vectors, does outperform the LDA based SVM. This result is expected as a counts based feature vector retain more information about the training set documents and is better suited to train a classifier that is to be used for the documents in the same domain.



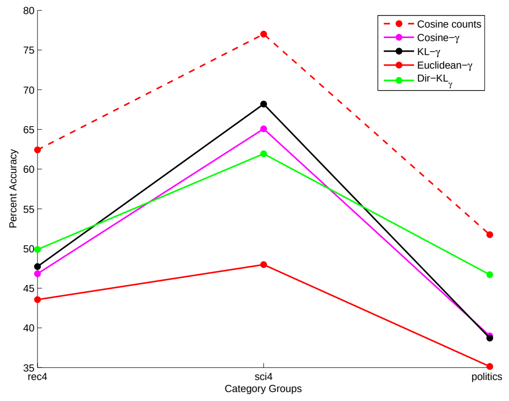
(a) NYT to Wiki Classification



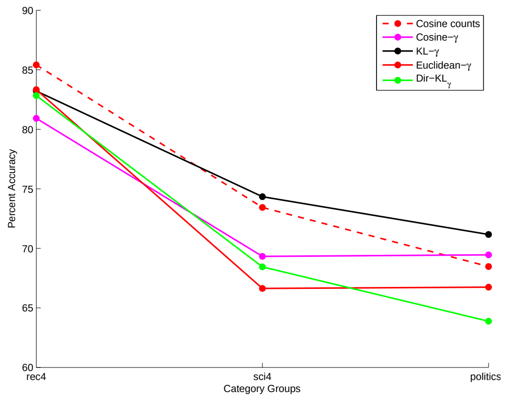
(b) Wiki to NYT Classification



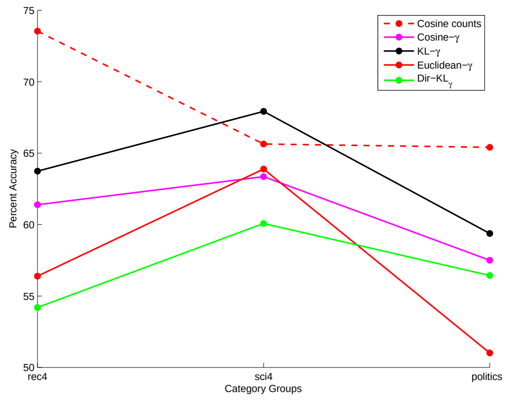
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification

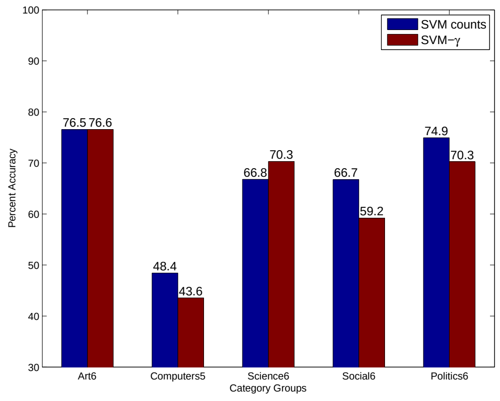


(e) Newsgroups to NYT Classification

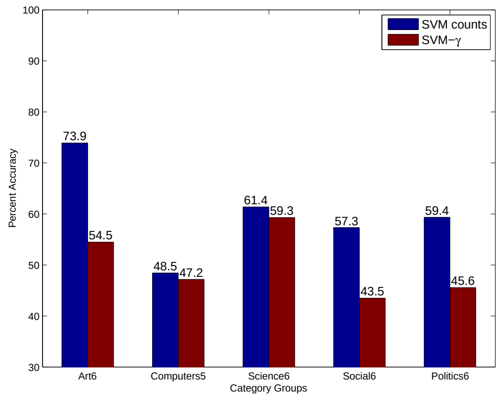


(f) NYT to Newsgroups Classification

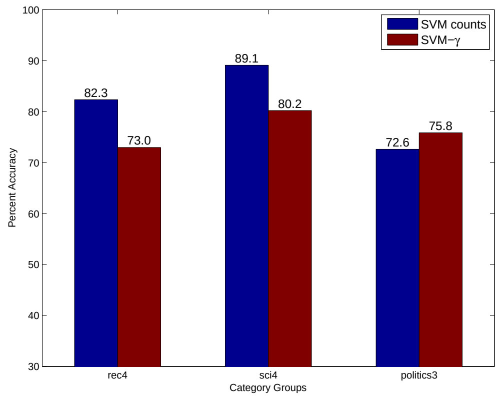
Figure 7.3: Comparison of four distance measures for cross domain document classification using the topic proportion (γ) vectors and KNN classifier with $k = 20$. The dotted red line is using the cross-domain classifier using the conventional cosine distance metric. We see that generally using only the topic proportion vectors for documents do not give us higher accuracy than using the counts. This result is not surprising as the topic proportion vectors use less than 1% of the dimensions as the original word counts vectors. Further, we observe that KL-distance for multinomial distributions performs best most of the times and other three distance measures all vary in performance including the Dir-KL distance (KL distance for Dirichlet distributions).



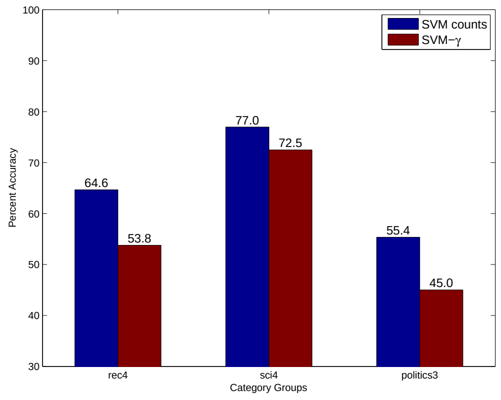
(a) NYT to Wiki Classification



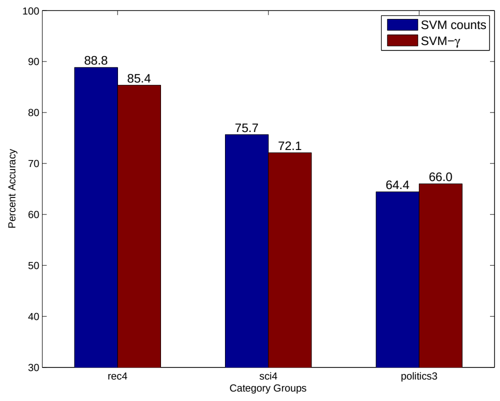
(b) Wiki to NYT Classification



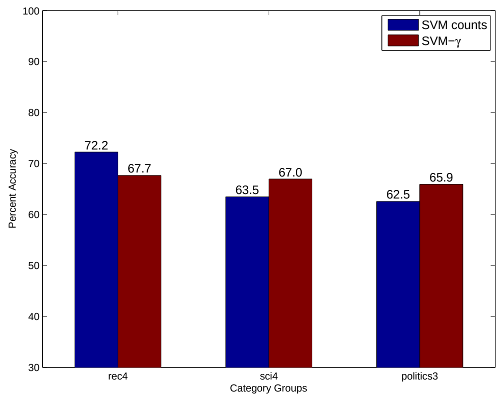
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification

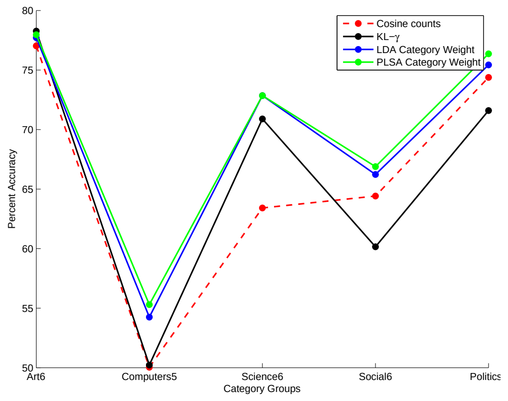


(e) Newsgroups to NYT Classification

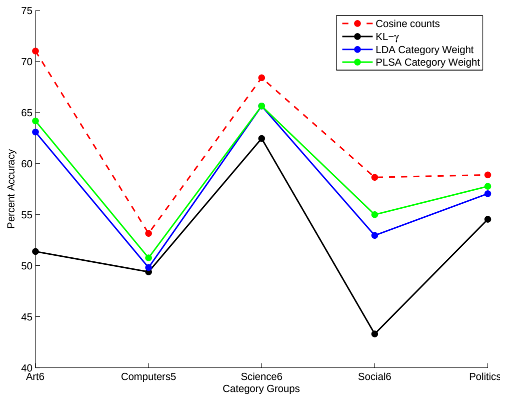


(f) NYT to Newsgroups Classification

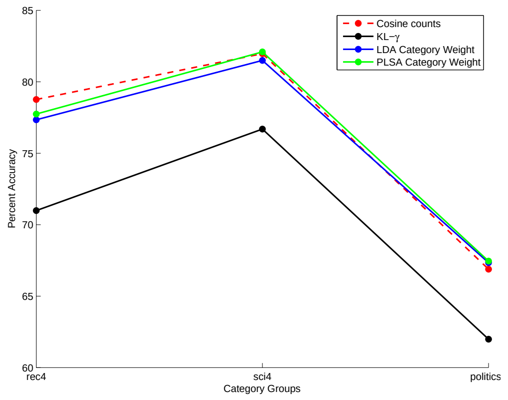
Figure 7.4: Comparison of SVM classifier using the topic proportion (γ) vectors and conventional word count feature vectors. The cross-domain document classification using the word counts is the blue bar and using the topic proportion feature vectors is the red bar. We see that using conventional algorithms such as KNN and SVMs on γ vectors do not give us a higher cross-domain document classification (blue bar is generally higher than red bar).



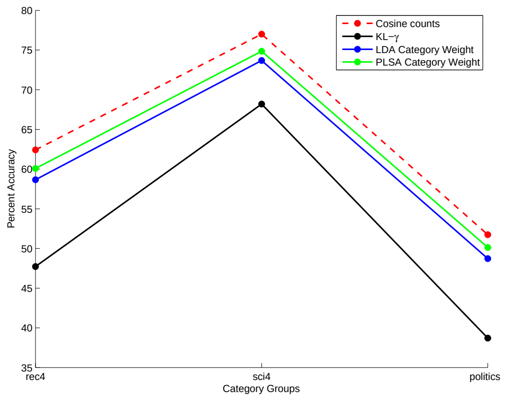
(a) NYT to Wiki Classification



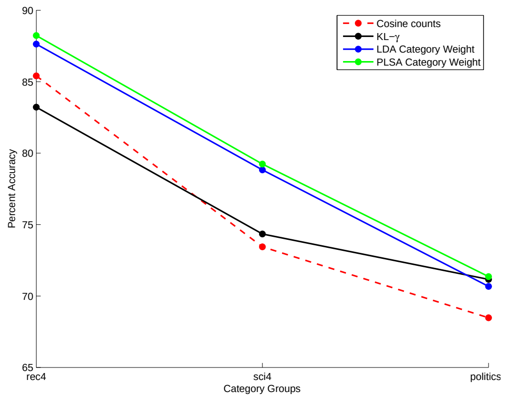
(b) Wiki to NYT Classification



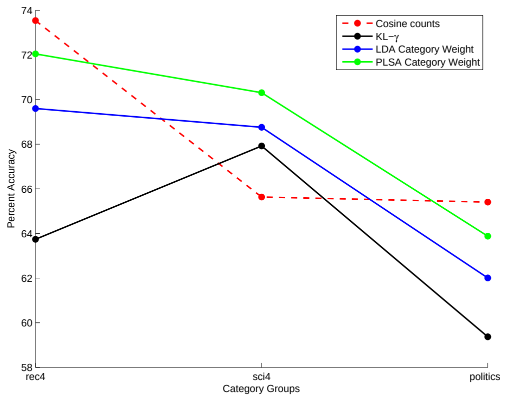
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification

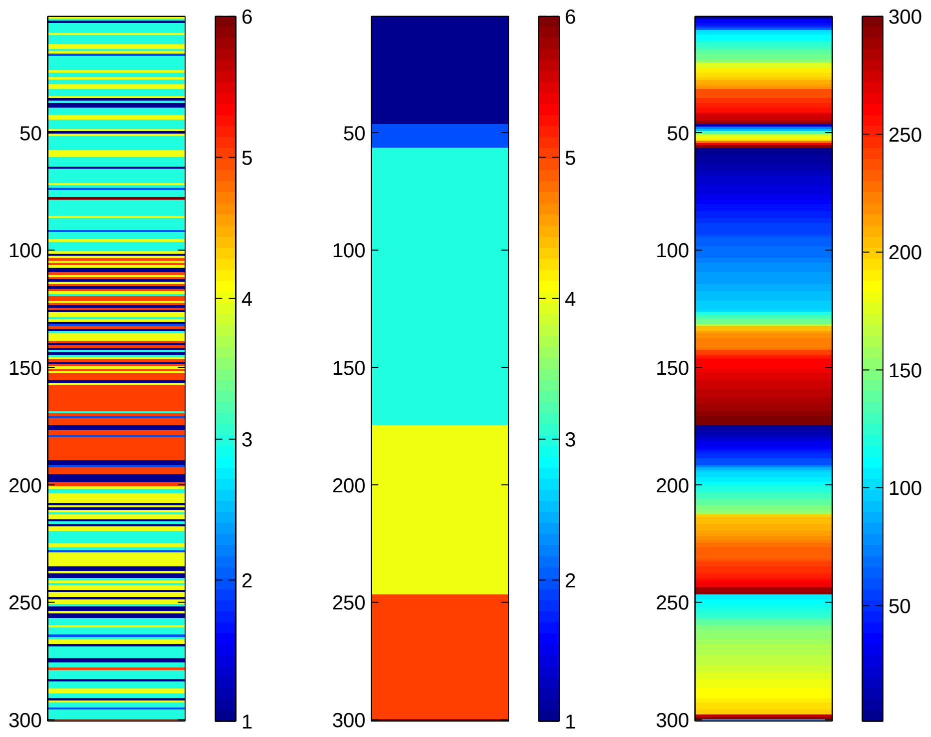


(e) Newsgroups to NYT Classification

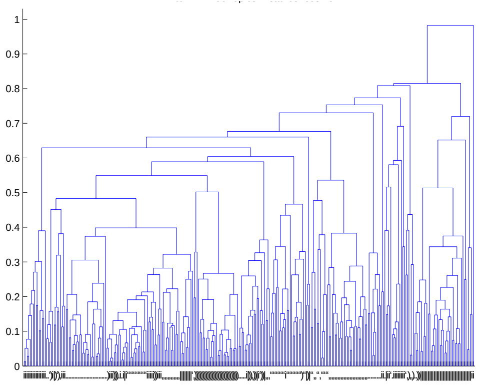


(f) NYT to Newsgroups Classification

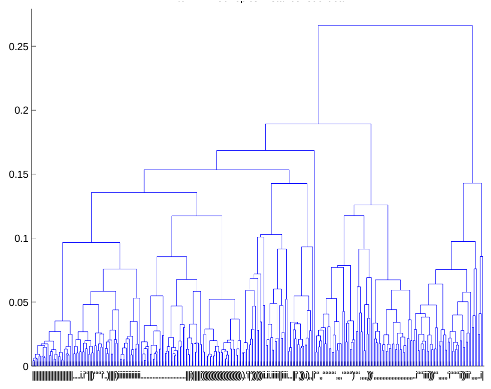
Figure 7.5: Comparison of KL-distance KNN and Cosine distance on word counts KNN classifier with our reduced topic proportion feature vector. Our reduced topic proportion vectors (PLSA Category Weight, LDA Category Weight) are obtained by adding up the weight of all the topics obtained from the same category. We use both PLSA and LDA topic model to obtain the topic proportion vectors. A document is classified to the category that has most words assigned to it. We see that by using words assigned to the topics of a particular category, we do get a higher accuracy. In most cases, we get a higher accuracy than conventional cosine distance measure on the word counts (blue and green line is higher than the red line).



(a) K-means Clustering Analysis of Topics from Arts Category Group

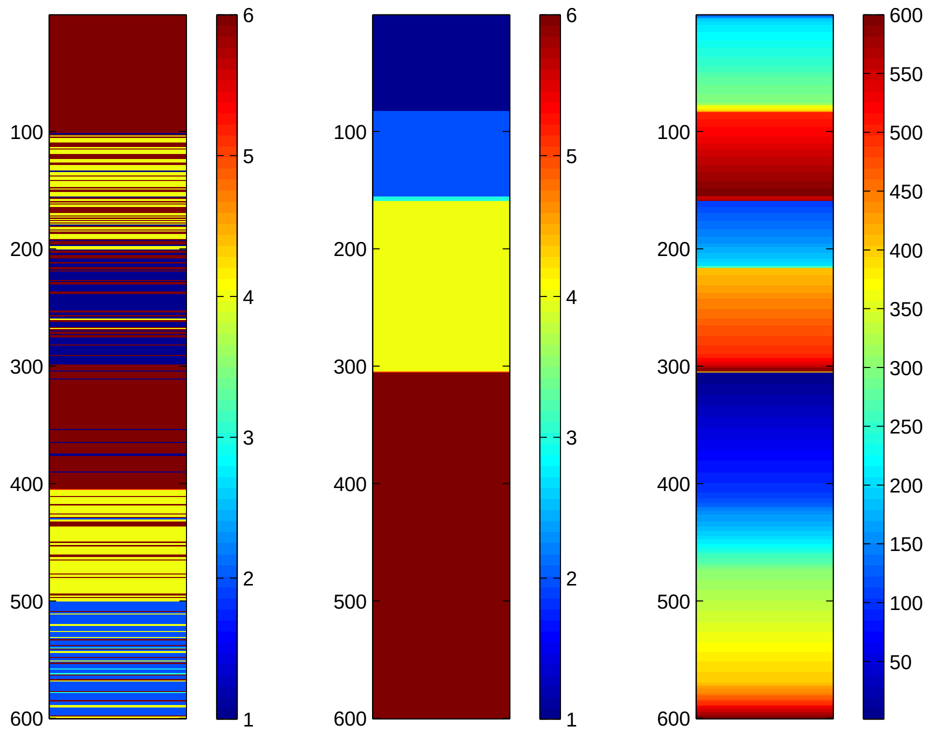


(b) Hierarchical Clustering Tree - Cosine Distance

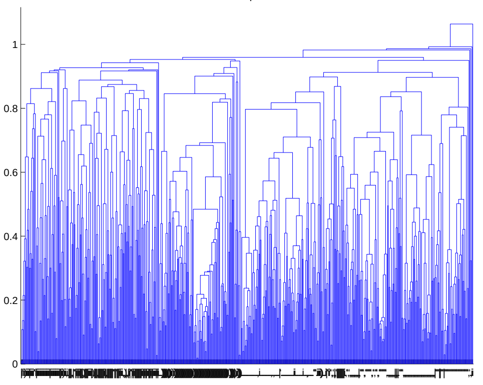


(c) Hierarchical Clustering Tree - Euclidean Distance

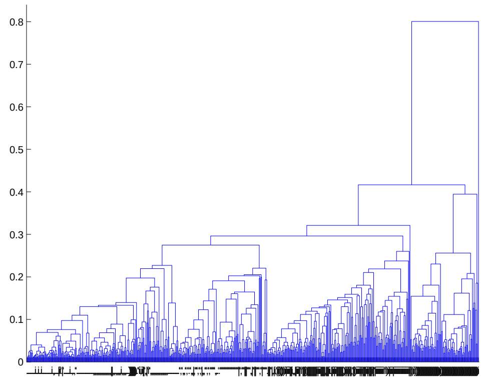
Figure 7.6: Clustering analysis of the 300 topics obtained the "Arts" category group, with 50 topics obtained independently from each of its six subcategories. The left most color bar shows the cluster assignment of each of the 300 topics and the middle figure shows the clusters sorted in increasing order, where each cluster in encoded as a unique color. The right most figure on the top row shows the actual composition of each cluster, where each point (topic) is denoted as a unique color. The hierarchical clustering trees are shown in the bottom row. A single character label corresponds to a single category. It is important to note in the figure is that same labels are clustered together with some patches of mixed area.



(a) K-means Clustering Analysis of Topics from Science Category Group



(b) Hierarchical Clustering Tree - Cosine Distance



(c) Hierarchical Clustering Tree - Euclidean Distance

Figure 7.7: Clustering analysis of the 600 topics obtained the "Science" category group, with 100 topics obtained independently from each of its six subcategories. The figure shows similar plots as described in Figure 7.6.

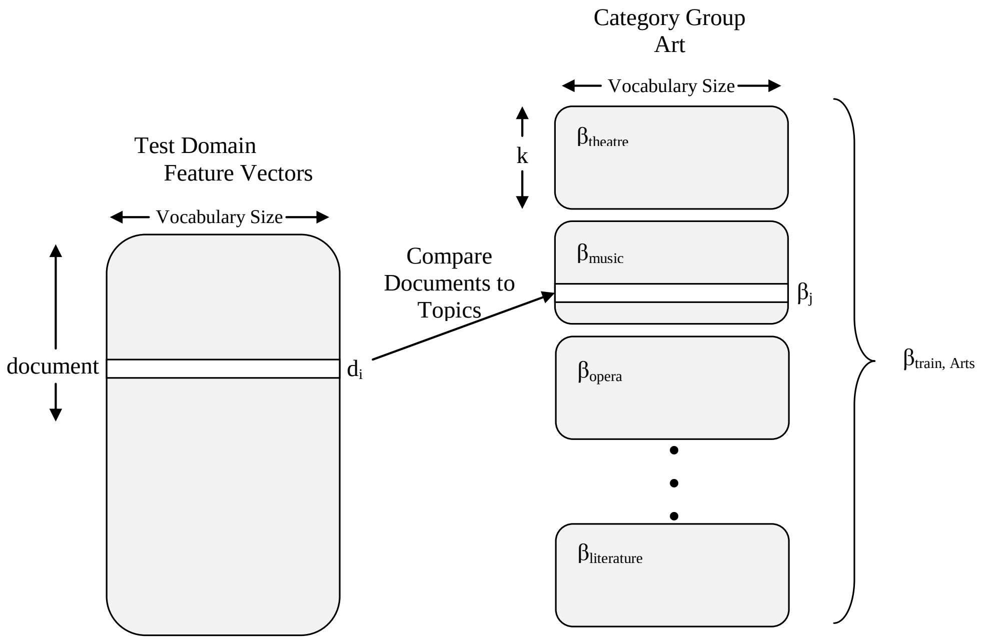


Figure 7.8: An illustration showing the set of β's learned from each category in the source domain independently.

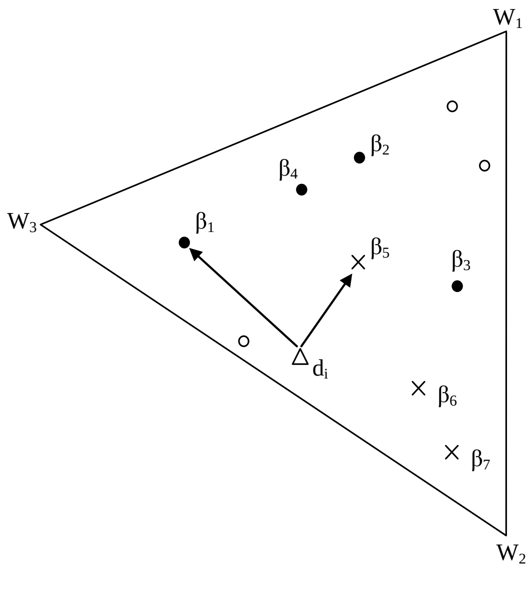
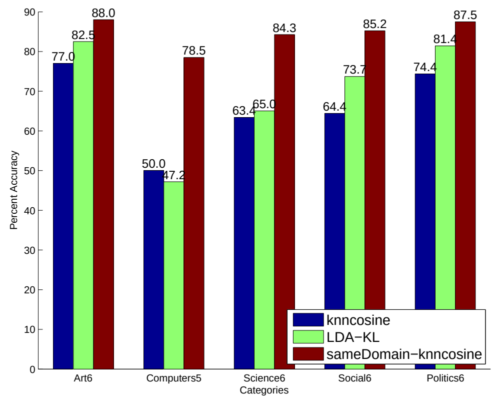
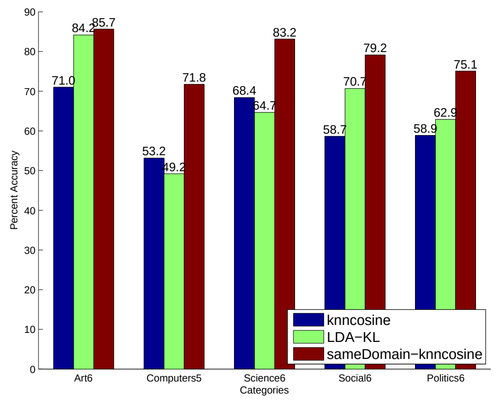


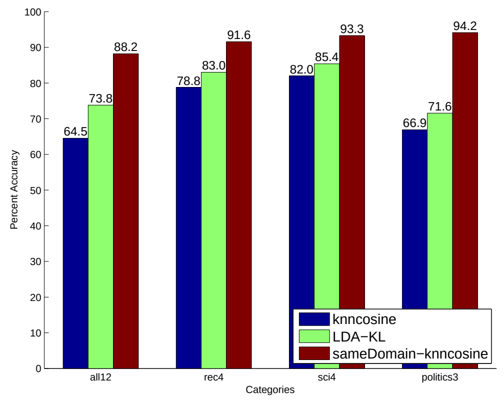
Figure 7.9: An illustration for comparing documents to single topics on a word simplex. KL-distance, being the natural distance measure on a simplex, gives us the best classification accuracy.



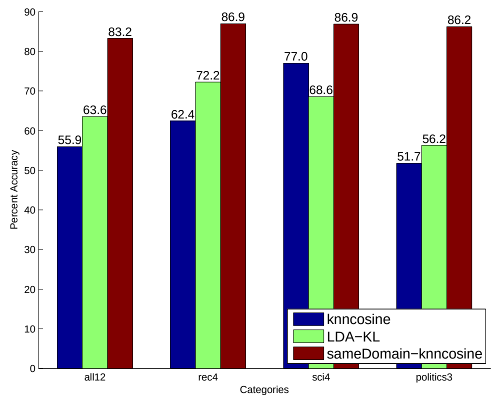
(a) NYT to Wiki Classification



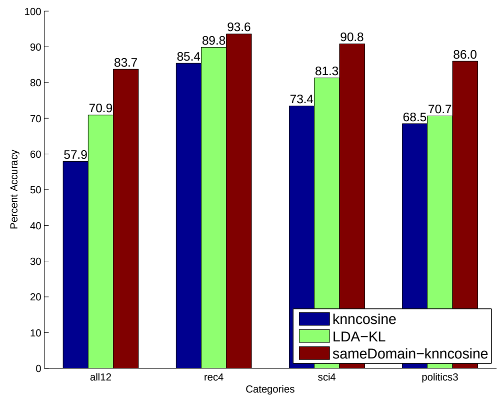
(b) Wiki to NYT Classification



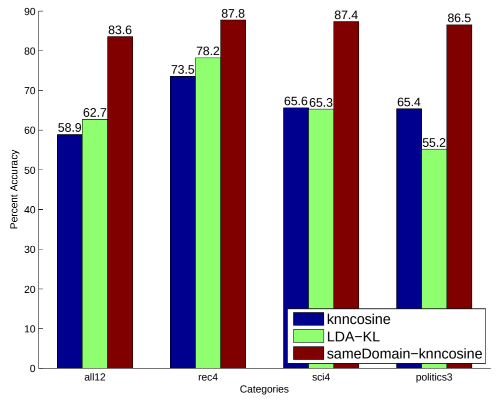
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification

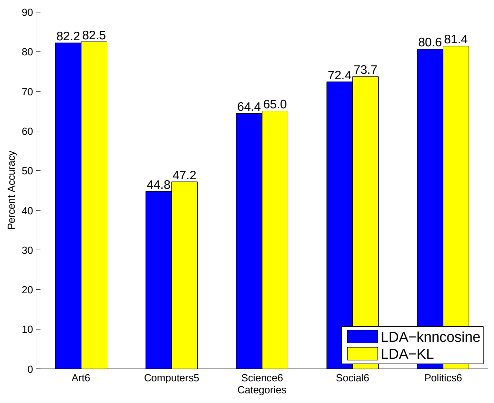


(e) Newsgroups to NYT Classification

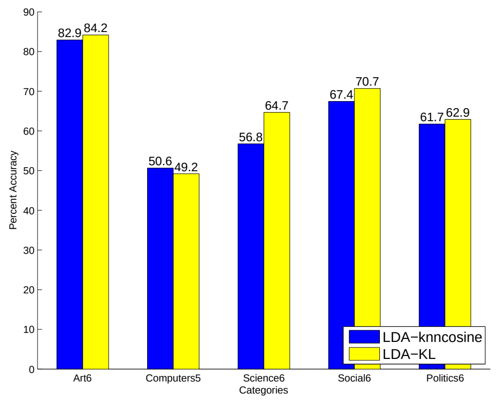


(f) NYT to Newsgroups Classification

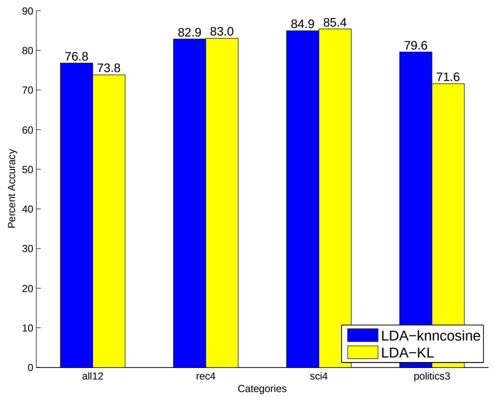
Figure 7.10: Comparison of cross-domain classification using KNN-Cosine (blue) and our algorithm, KNN-Beta-KL (green), with single domain classification (red). We see that our algorithm performs better than the conventional algorithm (green bar is higher than the blue bar) for cross-domain classification in most cases. The three rows of figure show three different pairs of domains used in our experiments.



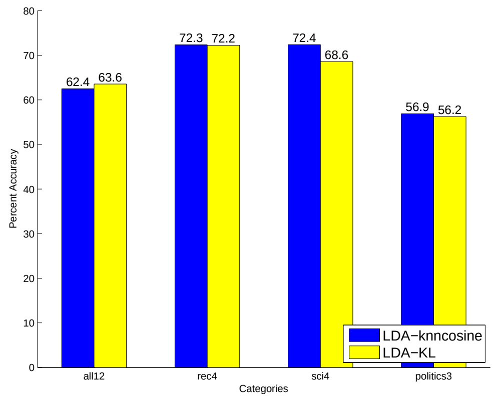
(a) NYT to Wiki Classification



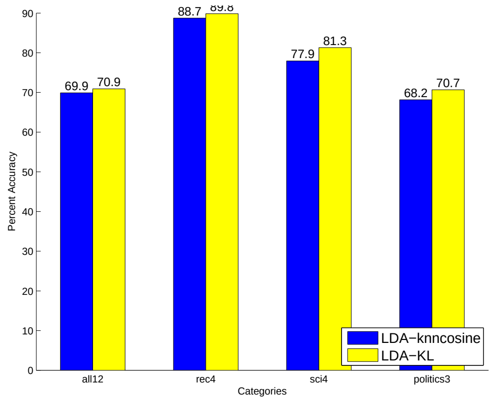
(b) Wiki to NYT Classification



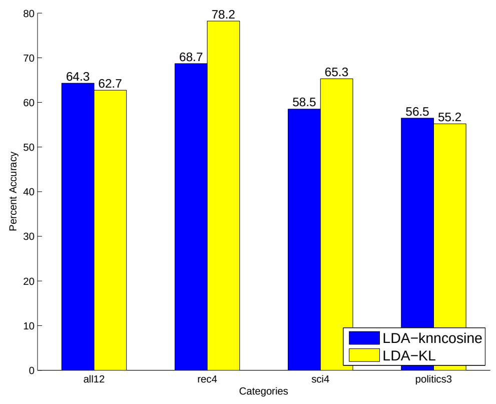
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification

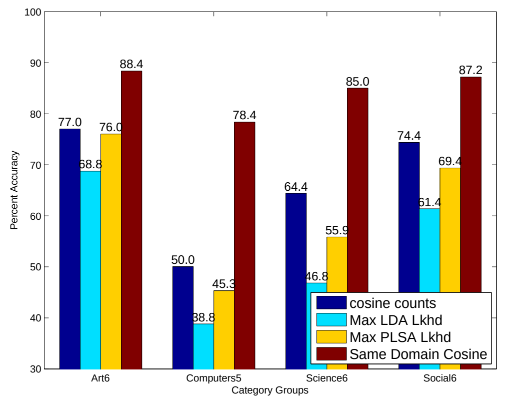


(e) Newsgroups to NYT Classification

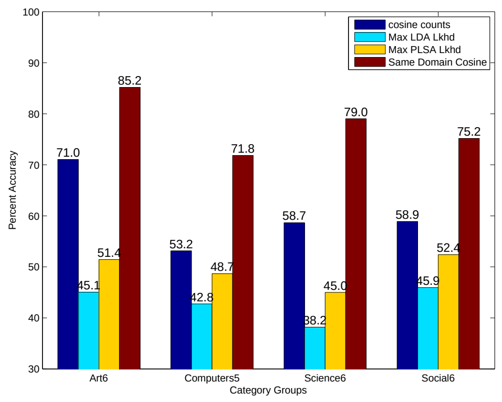


(f) NYT to Newsgroups Classification

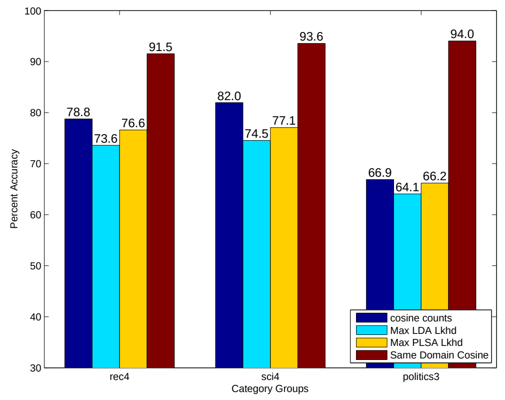
Figure 7.11: Comparison of two distance measures (1) KNN-Beta-Cosine (blue) and (2) KNN-Beta-KL (yellow) for comparing the documents to the topics obtained from LDA. KL distance performs better than the cosine distance measure in most of the cases. The accuracy shown are for the cross-domain classification. The three rows of figure show three different pairs of domains (Wikipedia, NYT and Newsgroups) used in our experiments.



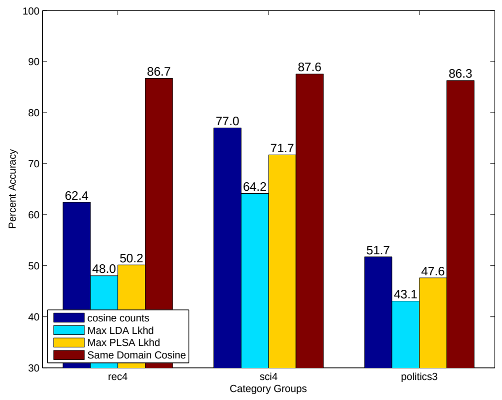
(a) NYT to Wiki Classification



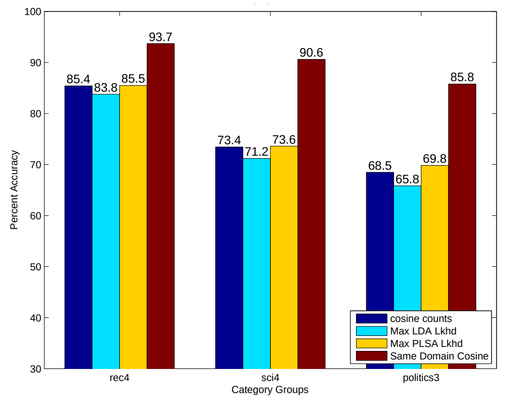
(b) Wiki to NYT Classification



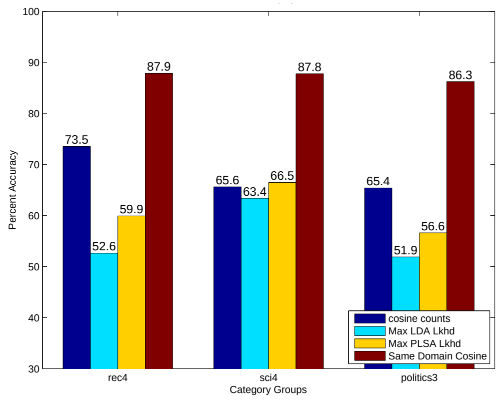
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification



(e) Newsgroups to NYT Classification



(f) NYT to Newsgroups Classification

Figure 7.12: Comparison of cross-domain classification using KNN-Cosine (blue) and one second formulation the documents are classified according to the likelihood given by an LDA (light blue) and PLSA (yellow) model. The LDA and PLSA models are trained on source domain categories. We see that the second formulation of our topic based algorithm does not perform better than the conventional algorithm (dark blue bar is higher than the light blue and the yellow bar) for cross-domain classification. This is not unexpected as fitting LDA and PLSA model on a new document gives it a lot flexibility in choosing the topics that generate the document. As all the categories are related, there is enough overlap between the topics to give the document high likelihood under each model. The three rows of figure show three different pairs of domains used in our experiments.

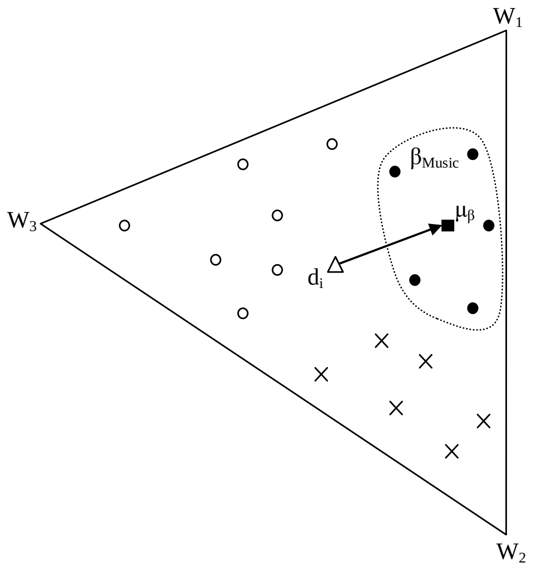
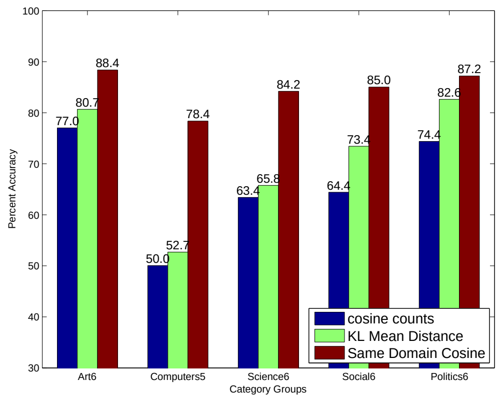
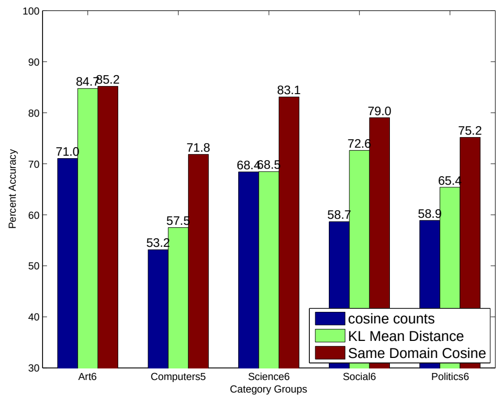


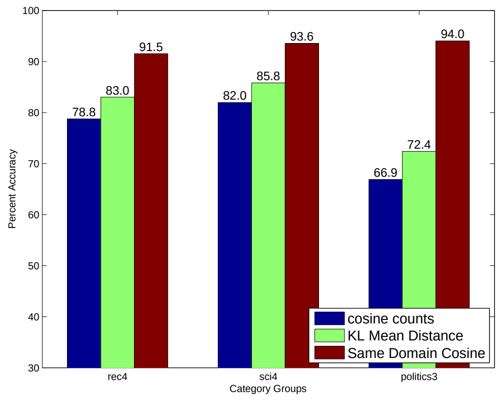
Figure 7.13: A figure showing the distance from the document to the set of topics in a cluster, where the cluster consists of all the topics obtained from one source domain category. The distance is computed using the KL-distance between the document and the mean of the cluster. This distance metric, that we call “topic centroid distance” always gives us a higher accuracy for cross-domain document classification when compared with the conventional word count distances and even when compared with our first formulations based on single topics.



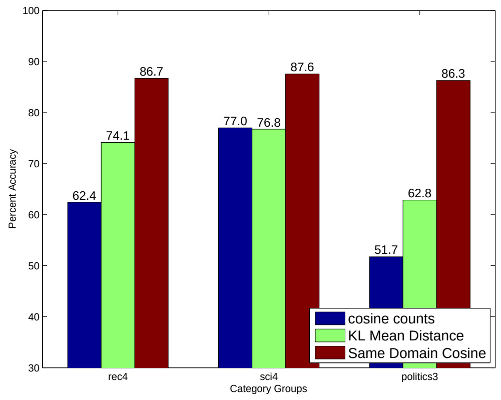
(a) NYT to Wiki Classification



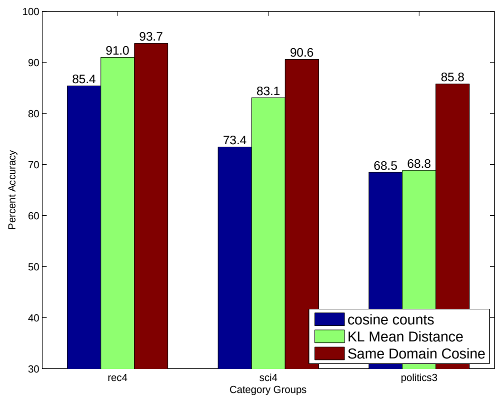
(b) Wiki to NYT Classification



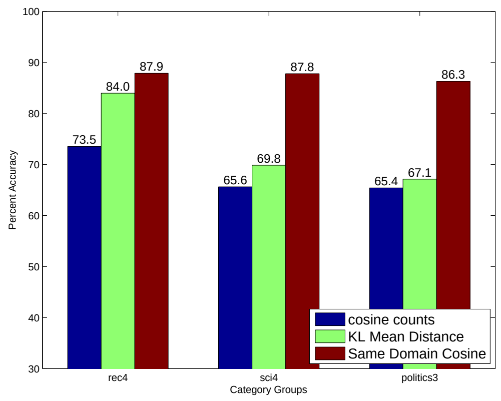
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification



(e) Newsgroups to NYT Classification



(f) NYT to Newsgroups Classification

Figure 7.14: Comparison of cross-domain classification using KNN-Cosine (blue) and our third formulation for topic based algorithm, KL-Mean-Distance (green), with single domain classification (red). We see that our algorithm performs better than the conventional algorithm (green bar is higher than the blue bar) for cross-domain classification in all cases. The third formulation is based on an assumption that the target domain models are generated using a mixture of multinomial model where all the components (topics) are equally weighed. In terms of distance measures, is the sum of the KL-distance between the document and all the topics.The three rows of figure show three different pairs of domains used in our experiments.

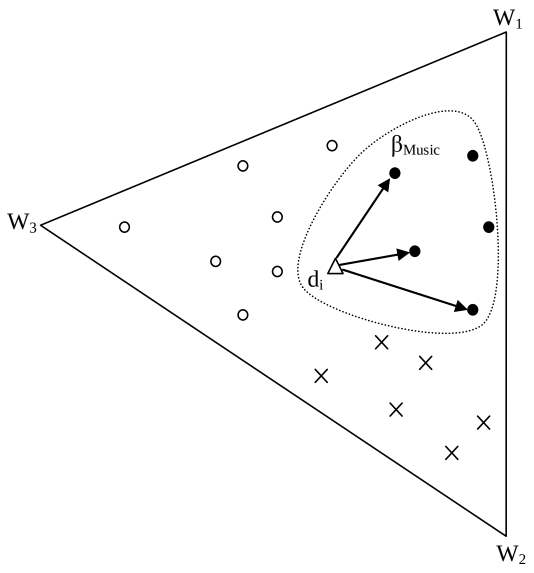
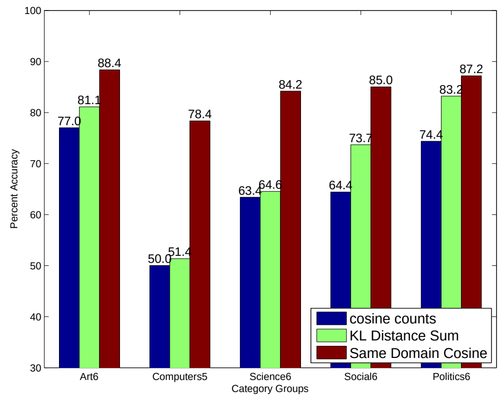
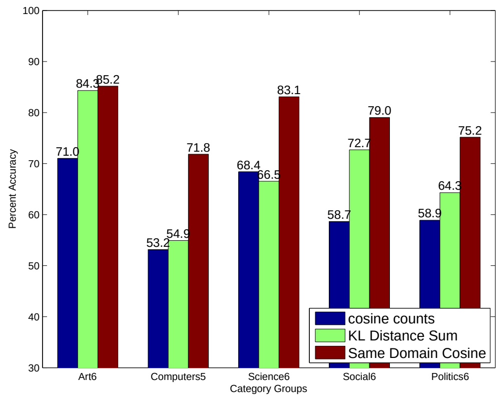


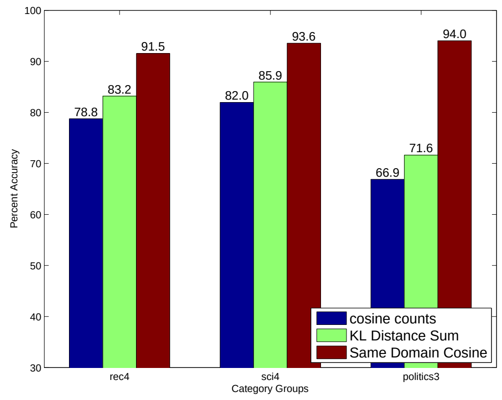
Figure 7.15: A figure showing the sum of all the distances from the document to the topics in a cluster, where the cluster consists of all the topics in one class. This distance metric, that we call “cluster sum distance”, gives us the accuracy that is always higher than when the document is classified using single topics, as discussed earlier.



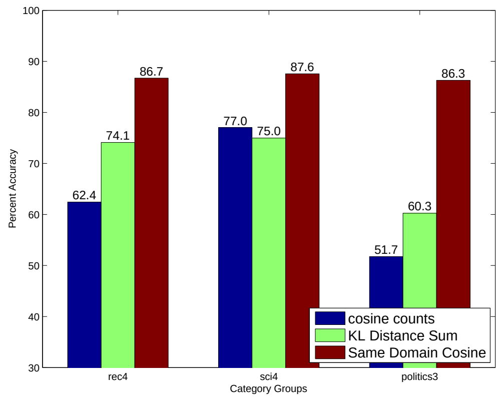
(a) NYT to Wiki Classification



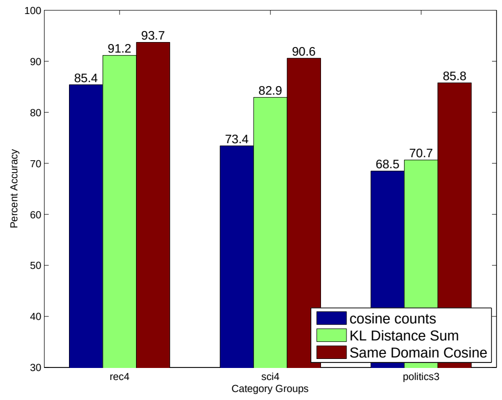
(b) Wiki to NYT Classification



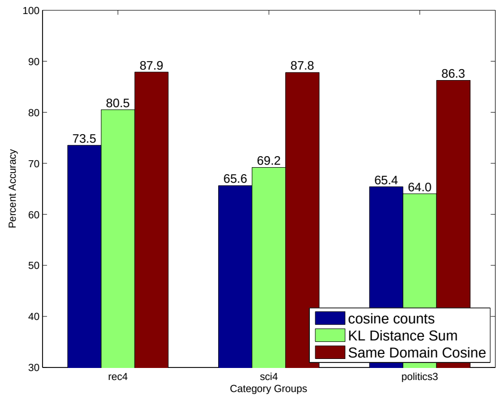
(c) Newsgroups to Wiki Classification



(d) Wiki to Newsgroups Classification

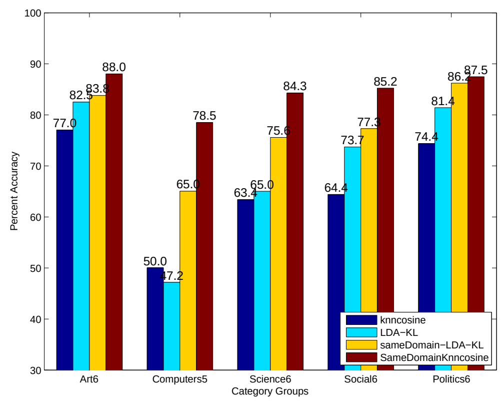


(e) Newsgroups to NYT Classification

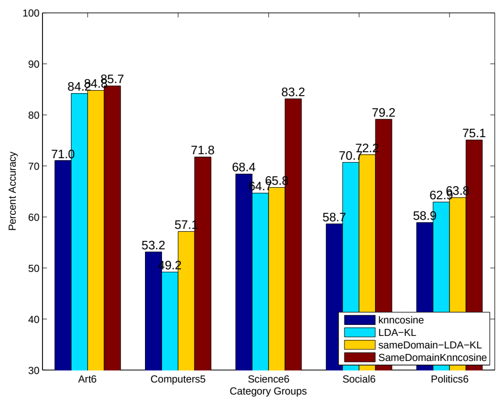


(f) NYT to Newsgroups Classification

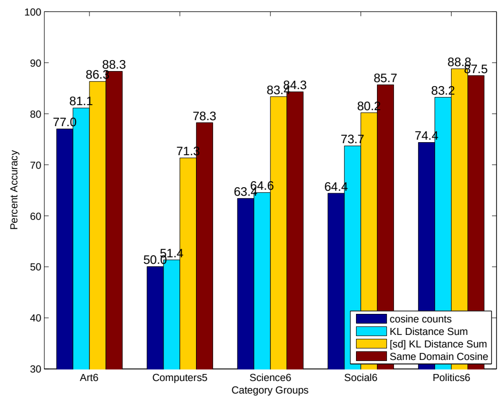
Figure 7.16: Comparison of cross-domain classification using KNN-Cosine (blue) and our fourth formulation for topic based algorithm, KL-Mean-Distance (green), with single domain classification (red). We see that our algorithm performs better than the conventional algorithm (green bar is higher than the blue bar) for cross-domain classification in all cases. The fourth formulation is based on an assumption that the target domain models are generated using an LDA model with a very large α value that generates uniform distribution over all the topics. In terms of distance measures, we show its equivalent to the KL-distance between the document and the mean of all the topics. The three rows of figure show three different pairs of domains used in our experiments.



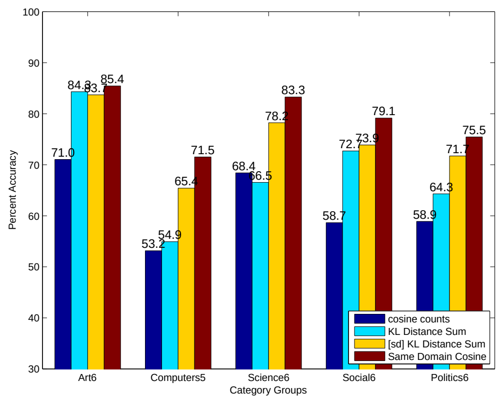
(a) NYT to Wiki - Formulation 1



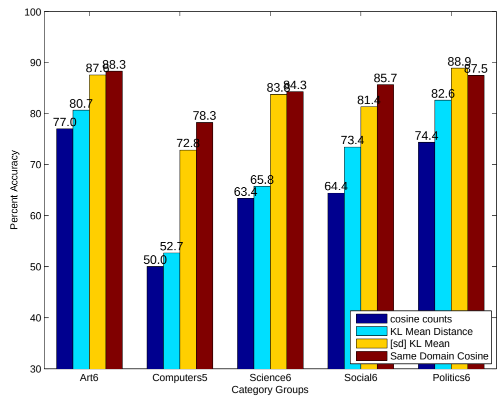
(b) Wiki to NYT - Formulation 1



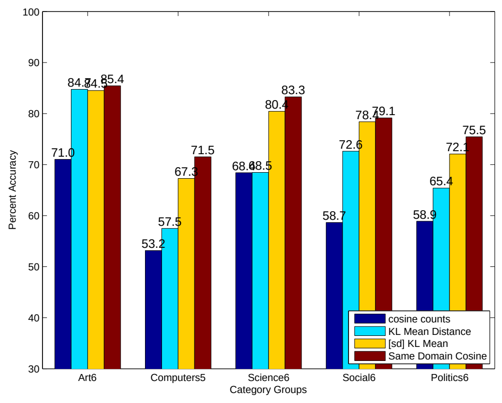
(c) NYT to Wiki - Formulation 3



(d) Wiki to NYT - Formulation 3

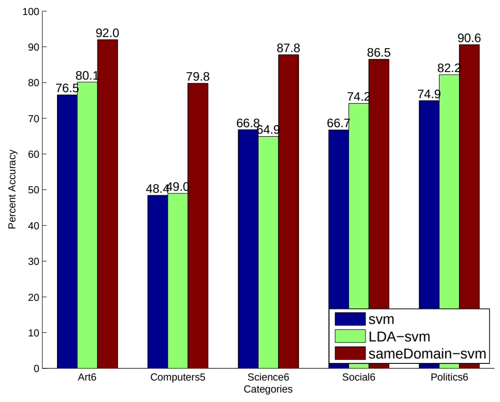


(e) NYT to Wiki - Formulation 4

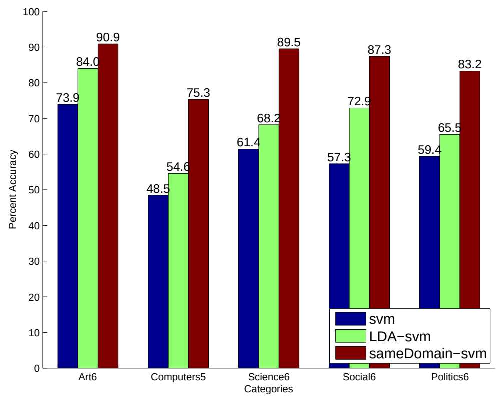


(f) Wiki to NYT - Formulation 4

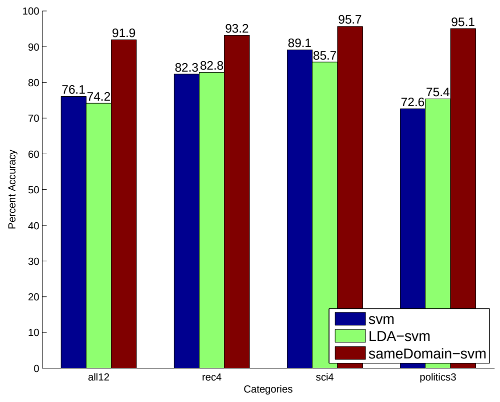
Figure 7.17: Figure shows the results of using the LDA K-nearest neighbor formulations on the same domain and compares it with the conventional K-nearest neighbor for the single domain classification task. The left most bar (dark blue) is the accuracy obtained by using the conventional KNN classifer for cross domain classification. The middle two bars (light blue and light orange) are the cross-domain and same domain accuracies obtained by our LDA based KNN classifier formulations. The right most bar (dark red) is the same domain accuracy obtained by the conventional KNN. We see that a conventional KNN, trained on word counts based feature vectors, does outperform the LDA based KNN formulations for the same domain classification (orange bar is lower than the red bar). The three rows show the graphs for three different formulations (1), (3) and (4) as described in this chapter.



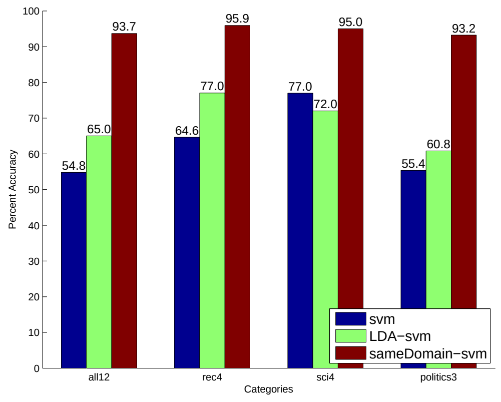
(a) LDA Beta Based SVM on NYT2Wiki



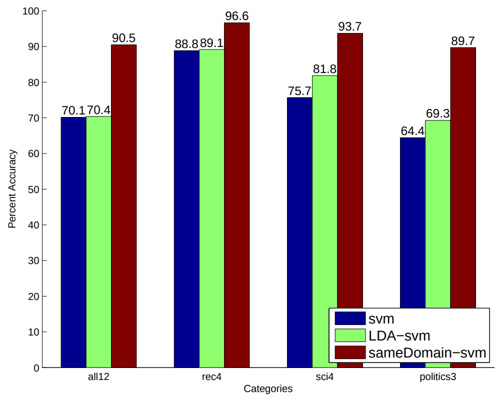
(b) LDA Beta Based SVM on Wiki2NYT



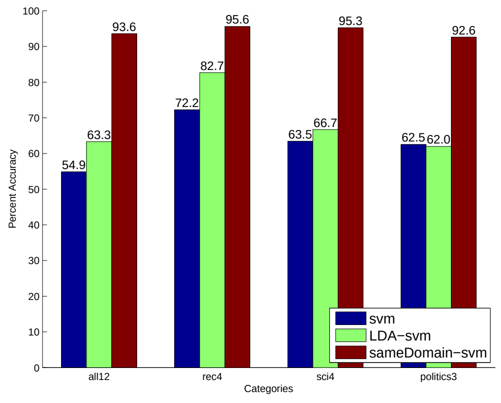
(c) LDA Beta Based SVM on News2Wiki



(d) LDA Beta Based SVM on Wiki2News



(e) LDA Beta Based SVM on News2NYT



(f) LDA Beta Based SVM on NYT2News

Figure 7.18: Comparison of cross-domain classification using conventional SVM on words counts (blue) and our approach of using SVM classification on the topics (green). We see that our method of training an SVM classifier using only the topics performs better than the conventional method of SVMs trained on word count vectors(green bar is higher than the blue bar) for cross-domain classification in most of the cases. The three rows of figures show three different pairs of domains (Wikipedia, NYT and Newsgroups) used in our experiments.

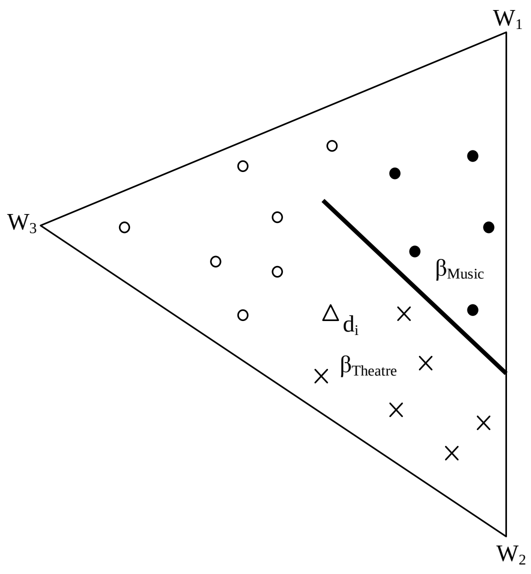
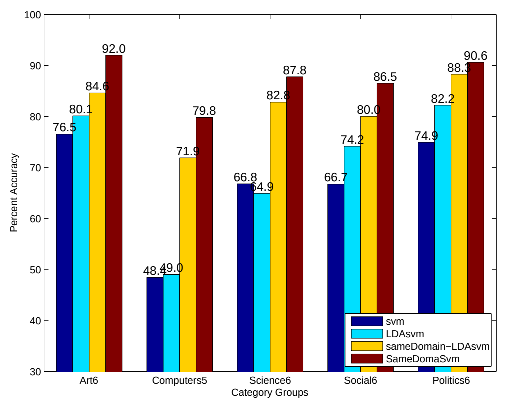
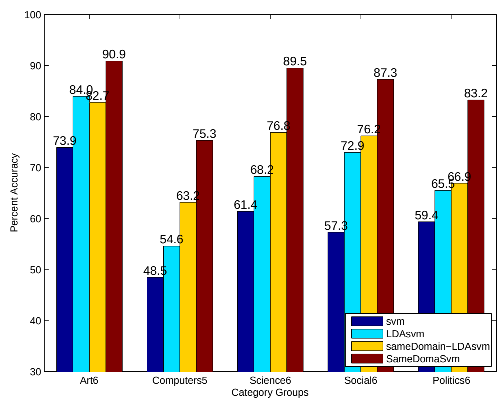


Figure 7.19: An illustration showing an output of the SVM classifier, when trained on topics, on a word simplex. The results obtained using the SVM on topics give us better cross-domain accuracy than the SVM classifier results obtained using the conventional word count feature vectors.



(a) NYT to Wiki Classification Using LDA-SVM



(b) Wiki to NYT Classification Using LDA-SVM

Figure 7.20: Figure shows the results of using the LDA SVM on the same domain and compares it with the conventional SVM on a single domain. The left most bar (dark blue) is the accuracy obtained by using the conventional SVM classifer for cross domain classification. The middle two bars (light blue and light orange) are the cross-domain and same domain accuracies obtained by our LDA based SVM classifier. The right most bar (dark red) is the same domain accuracy obtained by the conventional SVM. We see that a conventional SVM, trained on word counts based feature vectors, does outperform the LDA based SVM for the same domain classification (orange bar is lower than the red bar).

Group	Conventional Methods		Our Formulations			
	KNN	SVM	KNN-KL	Sum-KL	Mean-KL	LDA-SVM
	Wikipedia To NYT					
Arts	77.0	76.5	82.5	80.7	81.1	80.1
Computers	50.0	48.4	47.2	52.7	51.4	49.0
Science	63.4	66.8	65.0	65.8	64.6	64.9
Social	64.4	66.7	73.7	73.4	73.7	74.2
Politics	74.4	74.9	81.4	82.6	83.2	82.2
	NYT To Wikipedia					
Arts	71.0	73.9	84.2	84.7	84.3	84.0
Computers	53.2	48.5	49.2	57.5	54.9	54.6
Science	68.4	61.4	64.7	68.5	66.5	68.2
Social	58.7	57.3	70.7	72.6	72.7	72.9
Politics	58.9	59.4	62.9	65.4	64.3	65.5
	Newsgroups To Wikipedia					
rec4	78.8	82.3	83.0	83.0	83.2	82.8
sci4	82.0	89.1	85.4	85.8	85.9	85.7
poltics3	66.9	72.6	71.6	72.4	71.6	75.4
	Wikipedia To Newsgroups					
rec4	62.4	64.6	72.2	74.1	74.1	77.0
sci4	77.0	77.0	68.6	76.8	75.0	72.0
poltics3	51.7	55.4	56.2	62.8	60.3	60.8
	Newsgroups To NYT					
rec4	85.4	88.8	89.8	91.0	91.2	89.1
sci4	73.4	75.7	81.3	83.1	82.9	81.8
poltics3	68.5	64.4	70.7	68.8	70.7	69.3
	NYT To Newsgroups					
rec4	85.4	72.2	89.8	84.0	80.5	82.7
sci4	73.4	63.5	81.3	69.8	69.2	66.7
poltics3	68.5	62.5	70.7	67.1	64.0	62.0

Table 7.2: A summary of accuracy obtained from our methods and its comparison with the conventional methods. The table shows the results of four of our methods that show improvement over the conventional methods. The KNN-KL, Sum-KL and Mean-KL headings correspond to the formulations (1), (3) and (4) respectively as described in this chapter.

7.5 Conclusion

In this chapter, we provided various formulations for cross-domain document classification based on topic models. We analyze performance of different classifiers and the distribution of the topics obtained from the source domain using clustering algorithms. We develop four different distance metrics based on different assumptions for the relationship between the target domain documents and the source domain topics. We show that all four formulations perform better than the conventional algorithms when used for cross-domain document classification. We further show the use of a classifier such as SVMs on the topics from the source domain provides better target domain document classification than the SVMs used on the word counts feature vectors from the source domain documents. In the next chapter, we conclude our dissertation and provide some ideas for extending this research.

CHAPTER 8: Conclusions and Future Work

In this dissertation, we address the problem of cross-domain document classification. The cross-domain document classification is defined as the ability to classify unlabeled documents in any domain while using the labeled set of documents from a different domain. We first establish the fact that conventional classification algorithms such as Support Vector Machines (SVMs) and K-nearest neighbors (KNN) do not provide accurate classification when training and testing set of documents are used from two different domains of text. In this dissertation, we develop a new framework for cross-domain document classification based on topic models such as LDA model. We develop and describe two different ways of using LDA for classification, (1) using the vectors describing the distribution of topics in documents (γ vectors) and (2) using the topic as distribution of words (β vectors). We derive a few different formulations and distance metrics for our methods and give empirical results for each one of our formulations. We develop new classifiers and give theoretical and empirical justification for the effectiveness of our classifiers for the cross-domain classification.

8.1 Main Contributions

We perform experiments with 3 different domains, with each domain serving as a target and source domain. This combination of domains gives us 6 different pairs of domains for our experiments. Furthermore, we use different groups of categories from each of the domains that gives us wide variety of data to show consistency and robustness of our algorithms as well as any other experimental results. This thesis has four main contributions that we list below:

1. Establishing empirical evidence for the drop in accuracy using conventional classification algorithms when different domains of text documents are used for testing and training set.

2. Developing datasets using three different domains, namely (1) Wikipedia, (2) New York Times (NYT) and (3) 20-Newsgroups datasets. We document a method to gather large number of documents from similar categories from these different domains. This dataset can be used for other similar research.

3. Developing specialized algorithms for Wikipedia as the chosen source domain. Since Wikipedia offers a vast amount of text data, it is important to analyze and explore ways

of exploiting the information provided by Wikipedia for text classification. We use the unique structure of Wikipedia articles and develop algorithms based on the articles parsed into its different sections. We show that using different sections of the Wikipedia articles provide us better accuracy while using less data than using the entire Wikipedia article.

4. Analyzing different ways of utilizing the topic models for classification and developing four different classification algorithms that consistently outperform the conventional classifiers for cross-domain document classification. We derive distant metric for each one of our four metrics based on different assumptions and relationship between the source domain topics and target domain documents.

8.2 Future Work

Our research in the cross-domain document classification does provide classifiers, based on topic models, that consistently show a large improvement in the classification accuracy over conventional algorithms. In this section we briefly discuss some of the ideas that can be pursued to extend the work on cross-domain classification based on the topic models. We use LDA model as our main topic model. While using LDA model, we obtain topics from the categories of the source domain. These topics show a fair amount of separation among them and we use these topics to classify the documents in the target domain. In the future, we would like to devise ways to enhance this separation. We can develop metrics that measure which topics are show a large overlap with topics from other categories and eliminate them from the classification training set. Another direction of future work may be smoothing of the word counts data and the distributions obtained from them. Smoothing is often applied to Naive Bayes (NB) classifiers as NB classification also is sensitive to zero counts in the training data [77]. In our formulations, the distance metrics often involve KL-distance metrics that involve *digamma* function or the *log* function that show asymptotic behavior near zero and thus can be sensitive to small or zero counts. Some methods for smoothing the text data for Dirichlet distribution is the subject of Dr. Nallapati's doctoral thesis [78]. While extracting the topics from the source domain, we can also use the unlabeled target domain as the source of prior information. Since topic models use iterative EM algorithm to find topics, having a relevant prior or starting point that is influenced by the target domain may provide topics that are close to the target domain and thus improve the classification. In our framework, we do not make any assumptions about the underlying domains and the metrics we derive in our research can easily be adjusted to other topic models.

REFERENCES

[1] C. H. Gordon, G. A. Rendsburg, and N. H. Winter. Eblaitica: Essays on the ebla archives an deblaite language. 1987.

[2] R. Macleod. *The Library of Alexandria: Centre of Learning in the Ancient World*. London: I. B. Tauris. ISBN 1850435944.

[3] M. Dewey. *A Classification and Subject Index for Cataloguing and Arranging the Books and Pamphlets of a Library*. 1876. http://www.gutenberg.org/files/12513/12513-h/12513-h.htm.

[4] L. Mai Chan. *Cataloging and Classification: An Introduction*. New York: McGraw-Hill, 1994. ISBN 9780070105065.

[5] J. Lawrence. Introduction to neural networks. 1994. ISBN 1-883157-00-5.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(2):273–297, 1995. ISSN 1573-0565.

[7] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[8] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *IN AAAI-98 Workshop On Learning For Text Categorization*, pp. 41–48. AAAI Press, 1998.

[9] J. Hartigan. *Clustering Algorithms*. John Wiley and Sons, New York, 1975.

[10] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[11] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[12] G. Forman and I. Cohen. Learning from little: Comparison of classifiers given little training. In *in 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD*, pp. 161–172, 2004.

[13] G. M. Weiss and F. Provost. Learning when training data are costly: The effect of class distribution on tree induction. *JOURNAL OF ARTIFICIAL INTELLIGENCE RESEARCH*, 19:315–354, 2003.

[14] A. Kittur, Ed H. Chi, and B. Suh. What's in wikipedia?: mapping topics and conflict using socially annotated category structure. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pp. 1509–1512, 2009. ISBN 978-1-60558-246-7.

[15] W. Dakka and S. Cucerzan. Augmenting wikipedia with named entity tags. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pp. 545–552, 2008.

[16] A. Halavais and D. Lackaff. An analysis of topical coverage of wikipedia. *Journal of Computer-Mediated Communication*, 13(2):429–440, 2008.

[17] T. Holloway, M. Bozicevic, and K. Börner. Analyzing and visualizing the semantic coverage of wikipedia and its authors: Research articles. *Complex.*, 12(3):30–40, 2007. ISSN 1076-2787.

[18] R. B. Almeida, B. Mozafari, and J. Cho. On the evolution of wikipedia. In *International Conference on Weblogs and Social Media*, 2007.

[19] P. Wang, J. Hu, H. Zeng, L. Chen, and Z. Chen. Improving text classification by using encyclopedia knowledge. In *ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, pp. 332–341. IEEE Computer Society, Washington, DC, USA, 2007. ISBN 0-7695-3018-4.

[20] P. Wang, J. Hu, H. Zeng, and Z. Chen. Using wikipedia knowledge to improve text classification. *Knowl. Inf. Syst.*, 19(3):265–281, 2009. http://dblp.uni-trier.de/db/journals/kais/kais19.html#WangHZC09.

[21] E. Gabrilovich and S. Markovitch. Feature generation for text categorization using world knowledge. In *In IJCAI05*, pp. 1048–1053, 2005.

[22] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 331–339, 1995.

[23] P. Schonhofen. Identifying document topics using the wikipedia category network. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 456–462. IEEE Computer Society, Washington, DC, USA, 2006. ISBN 0-7695-2747-7.

[24] W. Dai, G. Xue, Q. Yang, and Y. Yu. Co-clustering based classification for out-of-domain documents. In *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 210–219. ACM, New York, NY, USA, 2007. ISBN 978-1-59593-609-7.

[25] P. Wang, C. Domeniconi, and J. Hu. Using wikipedia for co-clustering based cross-domain text classification. In *ICDM '08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pp. 1085–1090. IEEE Computer Society, Washington, DC, USA, 2008. ISBN 978-0-7695-3502-9.

[26] O. Medelyan, C. Legg, D. Milne, and I.H. Witten. Mining meaning from wikipedia. *Int. J. Hum.-Comput. Stud.*, 67(9):716–754, 2009.

[27] G. Heinrich. A generic approach to topic models. In *ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 517–532. Springer-Verlag, Berlin, Heidelberg, 2009. ISBN 978-3-642-04179-2.

[28] D. M. Blei and J. D. Mcauliffe. Supervised topic models. *Advances in Neural Information Processing Systems*, (21), 2007.

[29] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: a supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 248–256. Association for Computational Linguistics, Morristown, NJ, USA, 2009. ISBN 978-1-932432-59-6.

[30] H. Shan, A. Banerjee, and N. C. Oza. Discriminative mixed-membership models. In *ICDM '09: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pp. 466–475. IEEE Computer Society, Washington, DC, USA, 2009. ISBN 978-0-7695-3895-2.

[31] D. Andrzejewski and X. Zhu. Latent dirichlet allocation with topic-in-set knowledge. In *SemiSupLearn '09: Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, pp. 43–48. Association for Computational Linguistics, Morristown, NJ, USA, 2009. ISBN 978-1-932432-38-1.

[32] Y. Wang and G. Mori. Human action recognition by semilatent topic models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(10):1762–1774, 2009. ISSN 0162-8828.

[33] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes. In *IEEE 11th International Conference on Computer Vision*, pp. 1–8, 2007.

[34] D. Andrzejewski, A. Mulhern, B. Liblit, and X. Zhu. Statistical debugging using latent topic models. In *ECML '07: Proceedings of the 18th European conference on Machine Learning*, pp. 6–17. Springer-Verlag, Berlin, Heidelberg, 2007. ISBN 978-3-540-74957-8.

[35] D. D. Lewis. Reuters-21578 text categorization test collection, distribution 1.0. Downloaded from http://www.daviddlewis.com/resources/testcollections/reuters21578, May 2004.

[36] B. Pang and L. Lee. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 115–124. Association for Computational Linguistics, Morristown, NJ, USA, 2005.

[37] A. Banerjee and H. Shan. Latent dirichlet conditional naive-bayes models. In *ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, pp. 421–426. IEEE Computer Society, Washington, DC, USA, 2007. ISBN 0-7695-3018-4.

[38] L. Li and Y. Zhang. An empirical study of text classification using latent dirichlet allocation, 2006.

[39] A. K. McCallum. Simulated/real/aviation/auto usenet data. http://www.cs.umass.edu/~mccallum/code-data.html.

[40] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 89–98. ACM, New York, NY, USA, 2003. ISBN 1-58113-737-0.

[41] J. Hu, L. Fang, Y. Cao, H. Zeng, H. Li, Q. Yang, and Z. Chen. Enhancing text clustering by leveraging wikipedia semantics. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 179–186. ACM, New York, NY, USA, 2008. ISBN 978-1-60558-164-4.

[42] S. Swarup and S. R. Ray. Cross-domain knowledge transfer using structured representations. In *AAAI'06: Proceedings of the 21st national conference on Artificial intelligence*, pp. 506–511. AAAI Press, 2006. ISBN 978-1-57735-281-5.

[43] G. Xue, W. Dai, Q. Yang, and Y. Yu. Topic-bridged plsa for cross-domain text classification. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 627–634, 2008. ISBN 978-1-60558-164-4.

[44] C. Zhai, A. Velivelli, and B. Yu. A cross-collection mixture model for comparative text mining. In *In Proceedings of KDD 04*, pp. 743–748, 2004.

[45] Z. Marx, I. Dagan, Joachim M. Buhmann, and E. Shamir. Coupled clustering: a method for detecting structural correspondence. *J. Mach. Learn. Res.*, 3:747–780, 2003. ISSN 1532-4435.

[46] S. Sarawagi, S. Chakrabarti, and S. Godbole. Cross-training: learning probabilistic mappings between topics. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 177–186. ACM, New York, NY, USA, 2003. ISBN 1-58113-737-0.

[47] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger. Tackling the poor assumptions of naive bayes text classifiers. In *ICML*, pp. 616–623, 2003.

[48] I. Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, 2001. available online at http://www.intellektik.informatik.tu-darmstadt.de/~tom/IJCAI01/Rish.pdf.

[49] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[50] T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pp. 137–142. Springer-Verlag, London, UK, 1998. ISBN 3-540-64417-2.

[51] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152. ACM, New York, NY, USA, 1992. ISBN 0-89791-497-X.

[52] T. Hofmann. Probabilistic latent semantic analysis. In *In Proc. of Uncertainty in Artificial Intelligence, UAI99*, pp. 289–296, 1999.

[53] E. B. Andersen. Journal of the american statistical association. *Knowl. Inf. Syst.*, pp. 1248–1255, 1970.

[54] T.M. Cover and J.A. Thomas. *Elements of information theory*. Wiley-Interscience, 2006.

[55] J. Huang. Maximum likelihood estimation of dirichlet distribution parameters, 2006.

[56] T. Minka. Estimating a Dirichlet distribution. Web, 2000. http://www.stat.cmu.edu/~minka/papers/dirichlet/minka-dirichlet.pdf.

[57] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986.

[58] J. Lafferty, S. Della Pietra, and V. D. Pietra. Statistical learning algorithms based on bregman distances, 1997.

[59] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005. ISSN 1532-4435.

[60] R. T. Rockafellar. Convex analysis, 1970.

[61] Wikipedia. History of wikipedia — wikipedia, the free encyclopedia, 2010. http://en.wikipedia.org/w/index.php?title=History_of_Wikipedia&oldid=373757845. [Online; accessed 22-July-2010].

[62] Beautiful soup. http://www.crummy.com/software/BeautifulSoup/.

[63] C. Harrison. Wikiviz: Visualizing wikipedia. http://www.chrisharrison.net/projects/wikiviz/index.html.

[64] E. Sandhaus. *ew york times corpus: Corpus overview. Provided with the corpus*, 2008. LDC catalogue entry LDC2008T19.

[65] Home page for 20 newsgroups data set. http://people.csail.mit.edu/jrennie/20Newsgroups/.

[66] G. Bouma. Normalized (pointwise) mutual information in collocation. In *Proceedings of the Biennial GSCL Conference*, pp. 31–40, 2009.

[67] P.Srinivasan V. Ha-Thuc. A robust learning approach for text classification, 2008.

[68] G. Forman, I. Guyon, and A. Elisseeff. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

[69] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, 2004. ISSN 1931-0145.

[70] D. Mladenic and M. Grobelnik. Feature selection for unbalanced class distribution and naive bayes. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 258–267. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999. ISBN 1-55860-612-2.

[71] W. Penny. Kl-divergences of normal, gamma, dirichlet and wishart densities. *Technical report, Wellcome Department of Cognitive Neurology, University College London.*

[72] M. D. Hoffman, D. M. Blei, and P. R. Cook. Content-based musical similarity computation using the hierarchical dirichlet process. In Juan Pablo Bello, Elaine Chew, and Douglas Turnbull, editors, *ISMIR*, pp. 349–354, 2008. ISBN 978-0-615-24849-3. http://dblp.uni-trier.de/db/conf/ismir/ismir2008.html#HoffmanBC08.

[73] D. M. Blei, K. Franks, M. I. Jordan, and I. S. Mian. Statistical modeling of biomedical corpora: mining the caenorhabditis genetic center bibliography for genes related to life span. *BMC Bioinformatics*, 7:250, 2006. http://dblp.uni-trier.de/db/journals/bmcbi/bmcbi7.html#BleiFJM06.

[74] G.W. Corder and D.I. Foreman. *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach.* Wiley, 2009.

[75] J. Eguchi, S.and Copas. Interpreting kullback-leibler divergence with the neyman-pearson lemma. *J. Multivar. Anal.*, 97(9):2034–2040, 2006. ISSN 0047-259X.

[76] J. Shlens. Notes on kullback-leibler divergence and likelihood theory, 2007.

[77] F. He and X. Ding. Improving naive bayes text classifier using smoothing methods. In Giambattista Amati, Claudio Carpineto, and Giovanni Romano, editors, *ECIR*, volume 4425 of *Lecture Notes in Computer Science*, pp. 703–707. Springer, 2007. ISBN 978-3-540-71494-1. http://dblp.uni-trier.de/db/conf/ecir/ecir2007.html#HeD07.

[78] The smoothed-dirichlet distribution: Explaining kl-divergence based ranking in information retrieval. 2006.

[79] L. Cayton. Fast nearest neighbor retrieval for bregman divergences. In *ICML '08: Proceedings of the 25th International Conference on Machine Learning*, pp. 112–119. ACM, New York, NY, USA, 2008. ISBN 978-1-60558-205-4.

[80] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*. MIT Press, 2001.

[81] J.M. Gomez-Hidalgo, M. Buenaga-Rodrguez, L. Alfonso, U. Lpez, M.T. Martin-Valdivia, and M. Garcia-Vega. Integrating lexical knowledge in learning based text categorization. In *Proceedings of the 6th International Conference on the Statistical Analysis of Textual Data (JADT-02)*, pp. 313–322, 2002.

[82] E. Gabrilovich and S. Markovitch. Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorization with encyclopedic knowledge. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pp. 1301–1306, 2006.

[83] M. Girolami and A. Kabán. On an equivalence between plsi and lda. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 2003.

[84] A. Hotho, S. Staab, and G. Stumme. Wordnet improves text document clustering. In *In Proc. of the SIGIR 2003 Semantic Web Workshop*, pp. 541–544, 2003.

[85] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22:79–86, 1951.

[86] D. Lin. An information-theoretic definition of similarity. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 296–304. Morgan Kaufmann Publishers Inc., 1998. ISBN 1-55860-556-8.

[87] R. Nallapati and W. Cohen. Link-plsa-lda: A new unsupervised model for topics and influence of blogs. 2008.

[88] Q. Mei, C. Liu, H. Su, and C. Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *WWW*, pp. 533–542, 2006.

[89] P. Schonhofen. Annotating documents by wikipedia concepts. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, 1:461–467, 2008. ISBN 978-0-7695-3496-1.

[90] M. Steyvers and T. Griffiths. *Probabilistic Topic Models*. Lawrence Erlbaum Associates, 2007.

[91] Z. Syed, T. Finin, and A. Joshi. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press, March 2008.

[92] P. Viola and W. M. Wells, III. Alignment by maximization of mutual information. *Int. J. Comput. Vision*, 24(2):137–154, 1997. ISSN 0920-5691.

[93] T. Zesch and I. Gurevych. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, 2007.

[94] V. Zlatić, M. Božičević, H. štefančić, and M. Domazet. Wikipedias: Collaborative web-based encyclopedias as complex networks. *Physical Review E*, 74(1), 2006.

[95] H. M. Wallach, D. Mimno, and A. McCallum. Rethinking lda: Why priors matter. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, 2009.

[96] L. Rigouste, O. Cappe, and F. Yvon. Inference and evaluation of the multinomial mixture model for text clustering. *Information Processing and Management*, 5(43):1260–1280, 2007.

[97] John Willis Clark. *The Care Of Books: An Essay On The Development Of Libraries And Their Fittings, From The Earliest Times To The End Of The Eighteenth Century*. Cambridge University Press, Cambridge, 1901.

[98] C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[99] R. Fan, K. Chang, C.Hsieh, X. Wang, and C. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[100] A.Agarwal and Hal Daumé III. A geometric view of conjugate priors. *CoRR*, abs/1005.0047, 2010.

[101] T.W. Rauber, A. Conci, T. Braun, and K. Berns. Bhattacharyya probabilistic distance of the dirichlet density and its application to split-and-merge image segmentation. In *WSSIP08 15th International Conference on Systems, Signals, and Image Processing*, pp. 145–148, 2008.

[102] F. Nielsen and R. Nock. Sided and symmetrized bregman centroids. *IEEE Transactions on Information Theory*, 55(6):2882–2904, 2009. ISSN 0018-9448.

[103] F.Nielsen, J. Boissonnat, and R. Nock. Bregman voronoi diagrams: Properties, algorithms and applications. *CoRR*, abs/0709.2196, 2007.

[104] G. Druck, C. Pal, A. McCallum, and X. Zhu. Semi-supervised classification with hybrid generative/discriminative methods. In Pavel Berkhin, Rich Caruana, and Xindong Wu, editors, *KDD*, pp. 280–289. ACM, 2007. ISBN 978-1-59593-609-7. http://dblp.uni-trier.de/db/conf/kdd/kdd2007.html#DruckPMZ07.

APPENDIX A:
Wilcoxon Statistical Test Results

Following tables show the results of of the Wilcoxon rank sum test showing the statistical significance of the improvements by our method. Wilcoxon rank sum test was used as it does not assume a parametric distribution of the two samples [74]. The null hypothesis is that the accuracy samples obtained by the two methods, (1) conventional method and (2) our LDA based methods, belong to the same distribution. The results show that in most cases, we can reject this null hypothesis with 99% confidence. The tests are obtained by comparing the two accuracy samples of the two classifiers. The samples consisted of string of 1's and 0's signifying weather a particular document was accuracy classified or not.

The tables are laid out in the same order as the figures in Chapter 7. The p-values are listed in the table represents the probability that null hypothesis is true.

A.1 Comparing KNN-Cosine and KNN-LDA Using KL Distance

Tables A.1 to A.6 show the statistical test results for sub-figures (a) through (f) in figure 7.10.

Domains: NYT to Wiki				
Category	**knncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
Art6	77.03	82.51	1.143140E-017	1
Computers5	50.04	47.19	1.769157E-003	1
Science6	63.41	65.04	8.838840E-003	1
Social6	64.42	73.71	2.386436E-034	1
Politics6	74.38	81.42	7.675699E-018	1

Table A.1: P-values for cross domain NYT to Wiki. Between: knncosine and LDAKL

Domains: Wikipedia to NYT				
Category	**knncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
Art6	71.03	84.18	2.353846E-113	1
Computers5	53.16	49.22	8.356035E-006	1
Science6	68.41	64.67	1.287715E-007	1
Social6	58.65	70.70	1.435695E-048	1
Politics6	58.90	62.92	3.460969E-008	1

Table A.2: P-values for cross domain Wikipedia to NYT. Between: knncosine and LDAKL

Domains: Newsgroups to Wikipedia				
Category	**knncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	64.53	73.81	3.043221E-047	1
rec4	78.76	83.02	1.309288E-003	1
sci4	81.97	85.39	3.532474E-006	1
politics3	66.88	71.55	2.409783E-004	1

Table A.3: P-values for cross domain Newsgroups to Wikipedia. Between: knncosine and LDAKL

Domains: Wikipedia to Newsgroups				
Category	**knncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	55.94	63.55	5.543781E-032	1
rec4	62.42	72.25	1.167039E-020	1
sci4	77.01	68.58	4.336410E-017	1
politics3	51.74	56.24	1.071875E-003	1

Table A.4: P-values for cross domain Wikipedia to Newsgroups. Between: knncosine and LDAKL

Domains: Newsgroups to NYT				
Category	**knncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	57.92	70.90	2.871597E-143	1
rec4	85.41	89.84	1.505948E-013	1
sci4	73.44	81.29	2.925052E-025	1
politics3	68.48	70.67	3.060532E-002	1

Table A.5: P-values for cross domain Newsgroups to NYT. Between: knncosine and LDAKL

Domains: NYT to Newsgroups				
Category	**knncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	58.89	62.73	2.430838E-009	1
rec4	73.54	78.21	1.184613E-006	1
sci4	65.63	65.30	7.580971E-001	0
politics3	65.41	55.21	4.713171E-014	1

Table A.6: P-values for cross domain NYT to Newsgroups. Between: knncosine and LDAKL

Domains: NYT to Wiki				
Category	**LDAknncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
Art6	82.23	82.51	6.451224E-001	0
Computers5	44.76	47.19	7.843933E-003	1
Science6	64.40	65.04	3.035260E-001	0
Social6	72.41	73.71	7.518420E-002	0
Politics6	80.64	81.42	3.143091E-001	0

Table A.7: P-values for cross domain NYT to Wiki. Between: LDAknncosine and LDAKL

Domains: Wikipedia to NYT				
Category	**LDAknncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
Art6	82.91	84.18	1.376694E-002	1
Computers5	50.64	49.22	1.077030E-001	0
Science6	56.75	64.67	3.344569E-027	1
Social6	67.43	70.70	3.928332E-005	1
Politics6	61.75	62.92	1.051785E-001	0

Table A.8: P-values for cross domain Wikipedia to NYT. Between: LDAknncosine and LDAKL

Domains: Newsgroups to Wikipedia				
Category	**LDAknncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	76.78	73.81	7.287502E-007	1
rec4	82.85	83.02	8.931426E-001	0
sci4	84.89	85.39	4.837795E-001	0
politics3	79.57	71.55	1.327821E-011	1

Table A.9: P-values for cross domain Newsgroups to Wikipedia. Between: LDAknncosine and LDAKL

A.2 Comparing KNN-LDA Using Cosine and KNN-LDA Using KL Distance

Tables A.7 to A.12 show the statistical test results for sub-figures (a) through (f) in figure 7.11. These results show the comparison of using KL distance and Cosine distance measure, defined as (1 - cosine similarity), with the LDA topics. We see that in general the difference between the two measures is not statistical significant, however, the KL distance does provide a slightly better accuracy.

Domains: Wikipedia to Newsgroups				
Category	**LDAknncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	62.45	63.55	8.304009E-002	0
rec4	72.32	72.25	9.399733E-001	0
sci4	72.39	68.58	2.118888E-004	1
politics3	56.89	56.24	6.356201E-001	0

Table A.10: P-values for cross domain Wikipedia to Newsgroups. Between: LDAknncosine and LDAKL

Domains: Newsgroups to NYT				
Category	**LDAknncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	69.89	70.90	3.701209E-002	1
rec4	88.73	89.84	4.845328E-002	1
sci4	77.93	81.29	3.852562E-006	1
politics3	68.16	70.67	1.347853E-002	1

Table A.11: P-values for cross domain Newsgroups to NYT. Between: LDAknncosine and LDAKL

Domains: NYT to Newsgroups				
Category	**LDAknncosine**	**LDAKL**	**p-value**	**Reject Null (99%)**
all12	64.30	62.73	1.322907E-002	1
rec4	68.69	78.21	8.643282E-022	1
sci4	58.50	65.30	5.084705E-010	1
politics3	56.47	55.21	3.585560E-001	0

Table A.12: P-values for cross domain NYT to Newsgroups. Between: LDAknncosine and LDAKL

Domains: NYT to Wiki				
Category	**svm**	**LDAsvm**	**p-value**	**Reject Null (99%)**
Art6	76.53	80.12	4.771552E-008	1
Computers5	48.44	48.99	5.462959E-001	0
Science6	66.80	64.91	2.144264E-003	1
Social6	66.72	74.15	3.733569E-023	1
Politics6	74.94	82.22	2.459978E-019	1

Table A.13: P-values for cross domain NYT to Wiki. Between: svm and LDAsvm

Domains: Wikipedia to NYT				
Category	**svm**	**LDAsvm**	**p-value**	**Reject Null (99%)**
Art6	73.91	83.96	6.201274E-070	1
Computers5	48.45	54.56	4.683887E-012	1
Science6	61.38	68.22	1.450411E-021	1
Social6	57.29	72.91	6.188038E-081	1
Politics6	59.35	65.50	2.043576E-017	1

Table A.14: P-values for cross domain Wikipedia to NYT. Between: svm and LDAsvm

Domains: Newsgroups to Wikipedia				
Category	**svm**	**LDAsvm**	**p-value**	**Reject Null (99%)**
all12	76.08	74.16	1.358309E-003	1
rec4	82.34	82.79	7.224166E-001	0
sci4	89.10	85.68	2.434933E-007	1
politics3	72.62	75.39	2.183317E-002	1

Table A.15: P-values for cross domain Newsgroups to Wikipedia. Between: svm and LDAsvm

A.3 Comparing SVM Using Counts Features and SVM Using LDA Topics

Tables A.13 to A.18 show the statistical test results for sub-figures (a) through (f) in figure 7.18.

Domains: Wikipedia to Newsgroups				
Category	**svm**	**LDAsvm**	**p-value**	**Reject Null (99%)**
all12	54.81	65.04	1.660464E-056	1
rec4	64.65	77.02	9.038501E-034	1
sci4	76.98	72.01	4.085371E-007	1
politics3	55.36	60.82	6.224672E-005	1

Table A.16: P-values for cross domain Wikipedia to Newsgroups. Between: svm and LDAsvm

Domains: Newsgroups to NYT				
Category	**svm**	**LDAsvm**	**p-value**	**Reject Null (99%)**
all12	70.11	70.38	5.765521E-001	0
rec4	88.84	89.14	6.003025E-001	0
sci4	75.68	81.83	8.345365E-017	1
politics3	64.44	69.26	3.735993E-006	1

Table A.17: P-values for cross domain Newsgroups to NYT for svm and LDAsvm

Domains: NYT to Newsgroups				
Category	**svm**	**LDAsvm**	**p-value**	**Reject Null (99%)**
all12	54.88	63.33	7.173769E-039	1
rec4	72.25	82.65	1.623395E-028	1
sci4	63.45	66.70	2.490855E-003	1
politics3	62.54	61.97	6.690104E-001	0

Table A.18: P-values for cross domain NYT to Newsgroups for svm and LDAsvm

Domains: NYT to Wiki				
Category	**Cosine counts**	**KL Mean Distance**	**p-value**	**Reject Null (99%)**
Art6	77.03	80.67	2.318729E-008	1
Computers5	50.04	52.70	3.648596E-003	1
Science6	63.41	65.76	1.522082E-004	1
Social6	64.42	73.44	2.114631E-032	1
Politics6	74.38	82.63	2.458951E-024	1

Table A.19: P-values for cross domain NYT to Wiki. Between: Cosine counts and KL Mean Distance

Domains: Wikipedia to NYT				
Category	**Cosine counts**	**KL Mean Distance**	**p-value**	**Reject Null (99%)**
Art6	71.03	84.75	3.291039E-124	1
Computers5	53.16	57.50	7.719083E-007	1
Science6	68.41	68.46	9.485027E-001	0
Social6	58.65	72.62	1.643607E-065	1
Politics6	58.90	65.40	2.906596E-019	1

Table A.20: P-values for cross domain Wikipedia to NYT. Between: Cosine counts and KL Mean Distance

Domains: Newsgroups to Wikipedia				
Category	**Cosine counts**	**KL Mean Distance**	**p-value**	**Reject Null (99%)**
rec4	78.76	83.02	1.309288E-003	1
sci4	81.97	85.80	1.691840E-007	1
politics3	66.88	72.39	1.392973E-005	1

Table A.21: P-values for cross domain Newsgroups to Wikipedia. Between: Cosine counts and KL Mean Distance

A.4 Comparing KNN-Cosine and KNN-LDA KL Mean Distance

Tables A.19 to A.24 show the statistical test results for sub-figures (a) through (f) in figure 7.14.

Domains: Wikipedia to Newsgroups				
Category	**Cosine counts**	**KL Mean Distance**	**p-value**	**Reject Null (99%)**
rec4	62.42	74.14	3.952567E-029	1
sci4	77.01	76.75	7.893392E-001	0
politics3	51.74	62.85	4.375089E-016	1

Table A.22: P-values for cross domain Wikipedia to Newsgroups. Between: Cosine counts and KL Mean Distance

Domains: Newsgroups to NYT				
Category	**Cosine counts**	**KL Mean Distance**	**p-value**	**Reject Null (99%)**
rec4	85.41	91.00	1.755831E-021	1
sci4	73.44	83.09	2.508768E-038	1
politics3	68.48	68.79	7.567972E-001	0

Table A.23: P-values for cross domain Newsgroups to NYT. Between: Cosine counts and KL Mean Distance

Domains: NYT to Newsgroups				
Category	**Cosine counts**	**KL Mean Distance**	**p-value**	**Reject Null (99%)**
rec4	73.54	83.96	7.969488E-030	1
sci4	65.63	69.85	6.332788E-005	1
politics3	65.41	67.12	1.885315E-001	0

Table A.24: P-values for cross domain NYT to Newsgroups. Between: Cosine counts and KL Mean Distance

Domains: NYT to Wiki				
Category	**Cosine counts**	**KL Distance Sum**	**p-value**	**Reject Null (99%)**
Art6	77.03	81.13	2.706618E-010	1
Computers5	50.04	51.36	1.487446E-001	0
Science6	63.41	64.56	6.485582E-002	0
Social6	64.42	73.69	2.996782E-034	1
Politics6	74.38	83.23	4.787461E-028	1

Table A.25: P-values for cross domain NYT to Wiki. Between: Cosine counts and KL Distance Sum

Domains: Wikipedia to NYT				
Category	**Cosine counts**	**KL Distance Sum**	**p-value**	**Reject Null (99%)**
Art6	71.03	84.31	9.824783E-116	1
Computers5	53.16	54.92	4.506520E-002	0
Science6	68.41	66.54	7.828963E-003	1
Social6	58.65	72.71	2.312902E-066	1
Politics6	58.90	64.30	1.125382E-013	1

Table A.26: P-values for cross domain Wikipedia to NYT. Between: Cosine counts and KL Distance Sum

Domains: Newsgroups to Wikipedia				
Category	**Cosine counts**	**KL Distance Sum**	**p-value**	**Reject Null (99%)**
rec4	78.76	83.19	8.133625E-004	1
sci4	81.97	85.94	5.650083E-008	1
politics3	66.88	71.63	1.894803E-004	1

Table A.27: P-values for cross domain Newsgroups to Wikipedia. Between: Cosine counts and KL Distance Sum

A.5 Comparing KNN-Cosine and KNN-LDA KL Sum Distance

Tables A.25 to A.30 show the statistical test results for sub-figures (a) through (f) in figure 7.16.

Domains: Wikipedia to Newsgroups				
Category	**Cosine counts**	**KL Distance Sum**	**p-value**	**Reject Null (99%)**
rec4	62.42	74.12	5.259800E-029	1
sci4	77.01	74.97	3.488337E-002	0
politics3	51.74	60.25	5.414242E-010	1

Table A.28: P-values for cross domain Wikipedia to Newsgroups. Between: Cosine counts and KL Distance Sum

Domains: Newsgroups to NYT				
Category	**Cosine counts**	**KL Distance Sum**	**p-value**	**Reject Null (99%)**
rec4	85.41	91.15	1.128994E-022	1
sci4	73.44	82.93	5.288515E-037	1
politics3	68.48	70.65	3.252476E-002	0

Table A.29: P-values for cross domain Newsgroups to NYT. Between: Cosine counts and KL Distance Sum

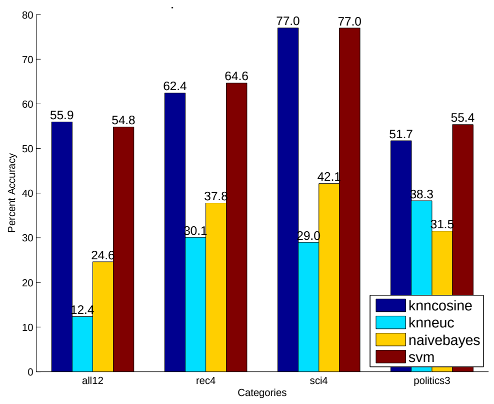
Domains: NYT to Newsgroups				
Category	**Cosine counts**	**KL Distance Sum**	**p-value**	**Reject Null (99%)**
rec4	73.54	80.51	1.689902E-013	1
sci4	65.63	69.19	7.664864E-004	1
politics3	65.41	64.03	2.979449E-001	0

Table A.30: P-values for cross domain NYT to Newsgroups. Between: Cosine counts and KL Distance Sum

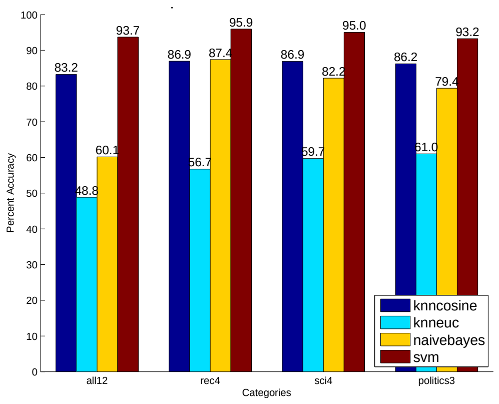
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: Accuracy For Cross-Domain Classification Using Conventional Algorithms

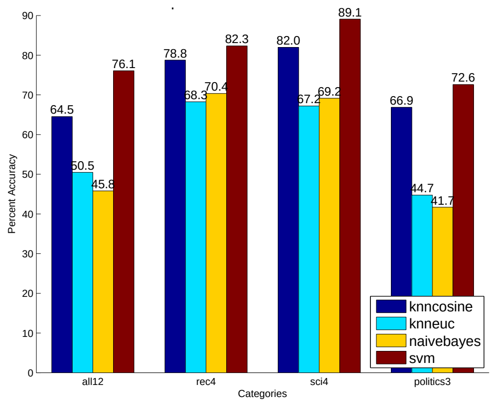
Some additional graphs for the cross domain accuracy drop are shown in this appendix.



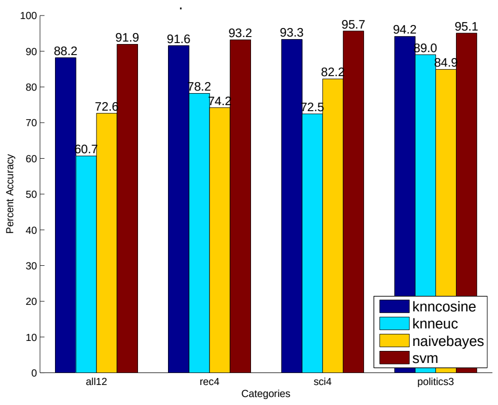
(a) caption: Wikipedia to Newsgroups



(b) caption: News2News

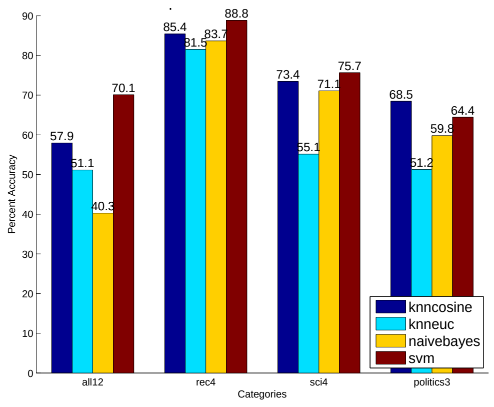


(c) caption: Newsgroups to Wikipedia

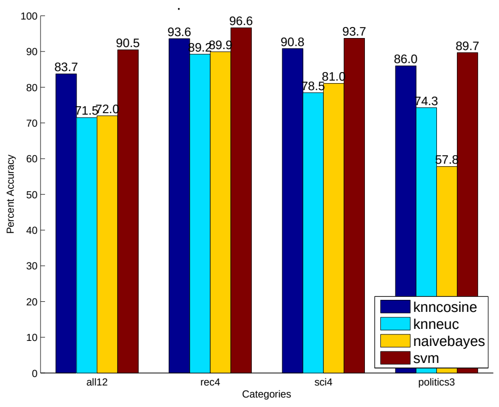


(d) caption: Wiki2Wiki

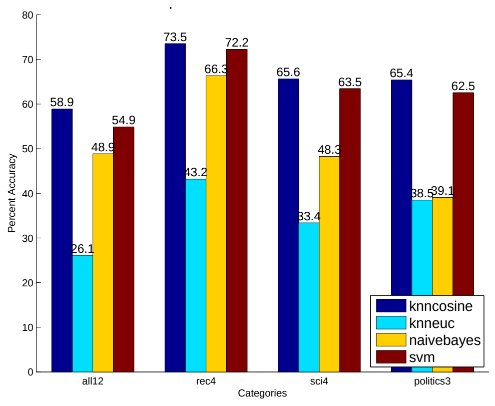
Figure B.1: Classification accuracy drop between domains Wikipedia and Newsgroups using conventional classifiers



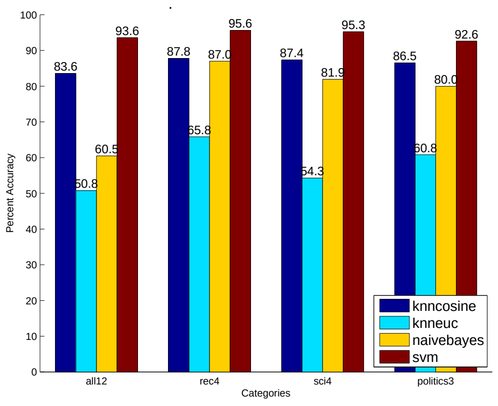
(a) caption: Newsgroups to NYT



(b) caption: NYT2NYT



(c) caption: NYT to Newsgroups



(d) caption: News2News

Figure B.2: Classification accuracy drop between domains New York Times and Newsgroups using conventional classifiers

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Mr. Ralph Glenn
   Spawar Systems Center, Pacific
   San Diego, California

4. Ms. Joan Kaina
   Spawar Systems Center, Pacific
   San Diego, California